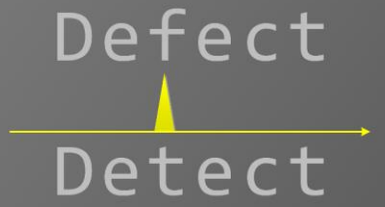


Linux
Core Dump Analysis
Accelerated

Third Edition

Dmitry Vostokov
Software Diagnostics Services

Accelerated Linux Core Dump Analysis: Training Course Transcript with GDB and WinDbg Practice Exercises, Third Edition

Published by OpenTask, Republic of Ireland

OpenTask books and magazines are available through booksellers and distributors worldwide. For further information or comments, send requests to press@opentask.com.

A CIP catalog record for this book is available from the British Library.

ISBN-l3: 978-1-912636-59-4 (Paperback)

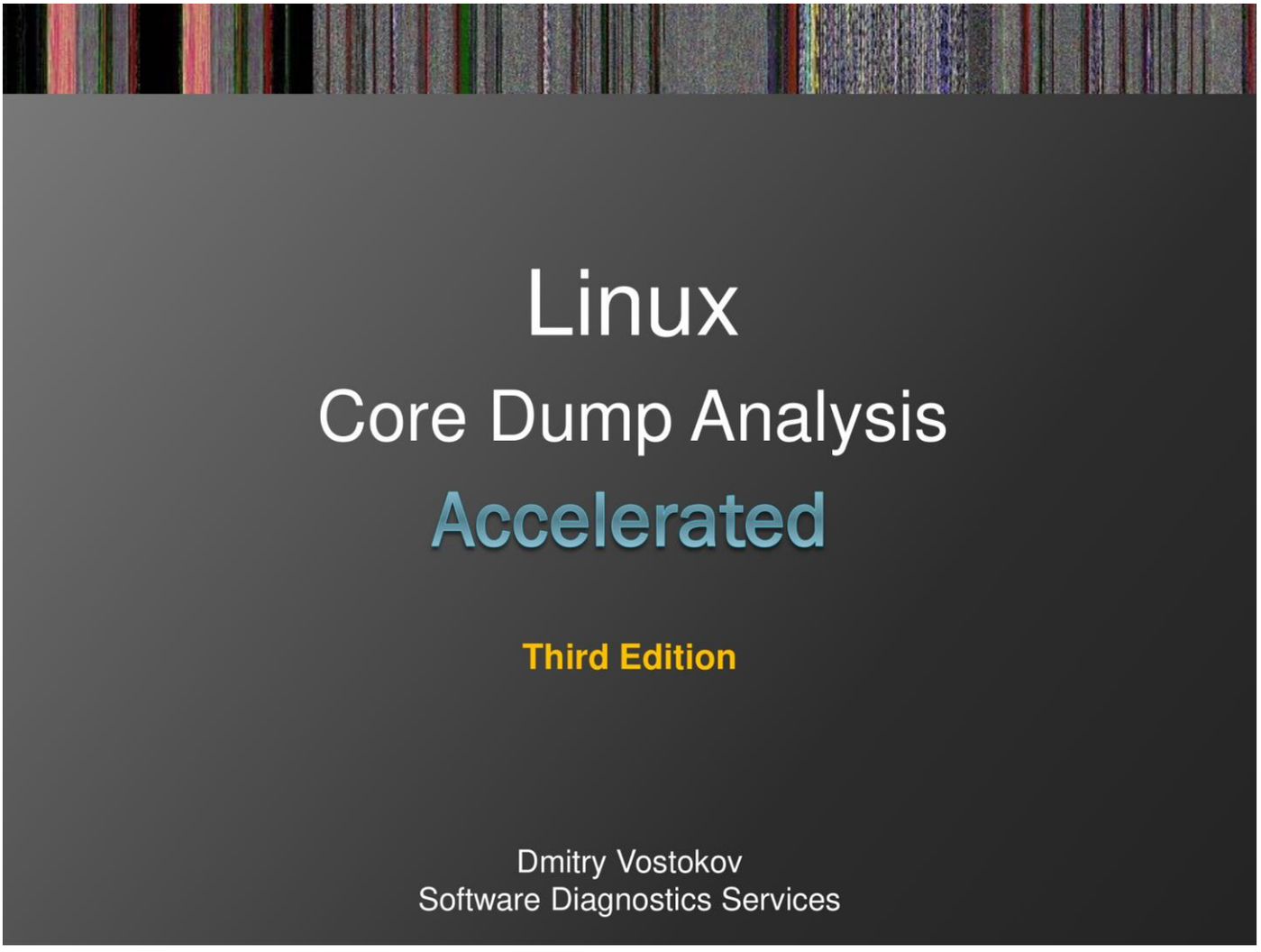
Revision 3.02 (January 2023)

Contents

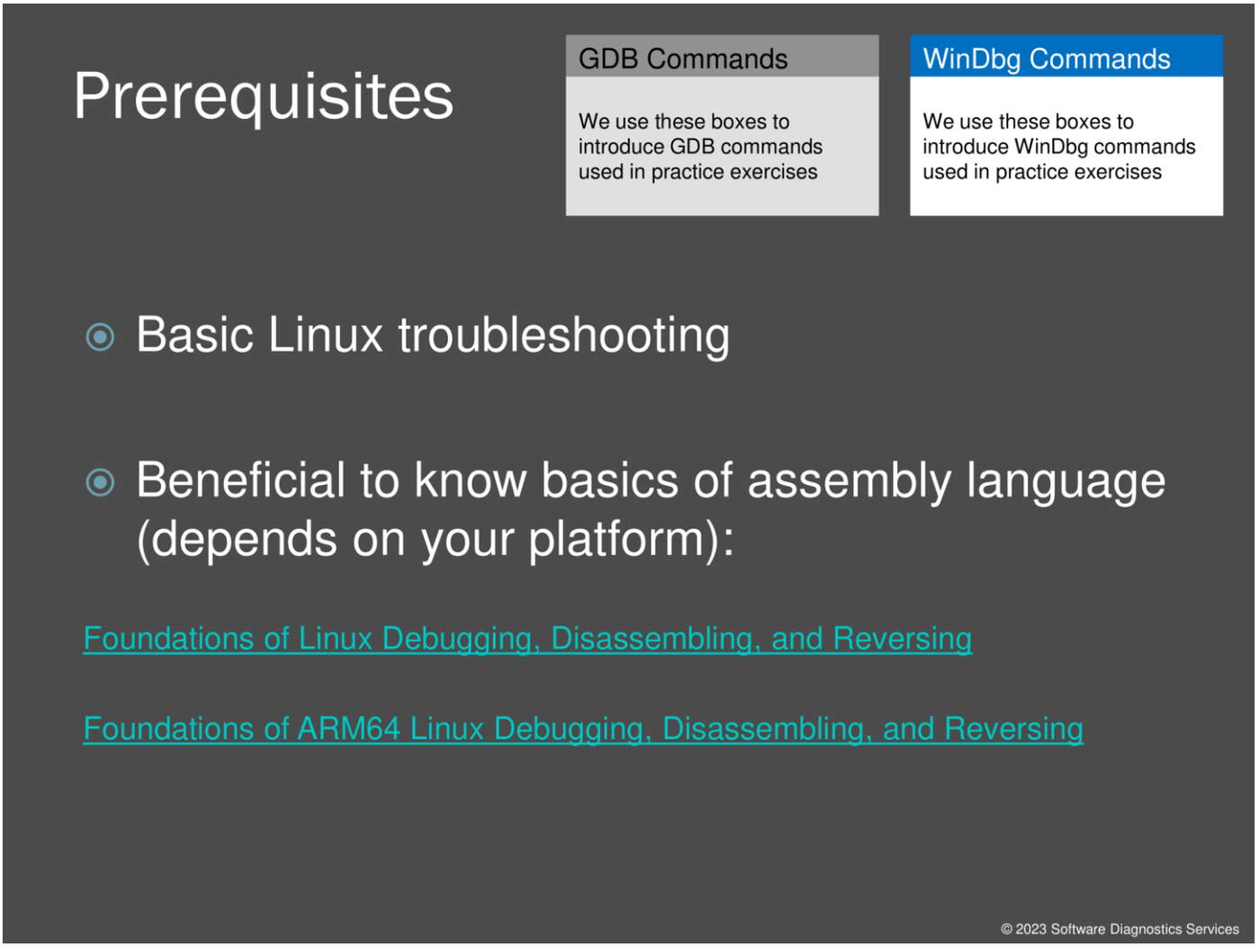
About the Author

Dmitry Vostokov is an internationally recognized expert, speaker, educator, scientist, inventor, and author. He is the founder of the pattern-oriented software diagnostics, forensics, and prognostics discipline (Systematic Software Diagnostics), and Software Diagnostics Institute (DA+TA: DumpAnalysis.org + TraceAnalysis.org). Vostokov has also authored more than 50 books on software diagnostics, anomaly detection and analysis, software and memory forensics, root cause analysis and problem solving, memory dump analysis, debugging, software trace and log analysis, reverse engineering, and malware analysis. He has over 25 years of experience in software architecture, design, development, and maintenance in various industries, including leadership, technical, and people management roles. Dmitry also founded Syndromatix, Anolog.io, BriteTrace, DiaThings, Logtellect, OpenTask Iterative and Incremental Publishing (OpenTask.com), Software Diagnostics Technology and Services (former Memory Dump Analysis Services) PatternDiagnostics.com, and Software Prognostics. In his spare time, he presents various topics on Debugging.TV and explores Software Narratology, its further development as Narratology of Things and Diagnostics of Things (DoT), Software Pathology, and Quantum Software Diagnostics. His current interest areas are theoretical software diagnostics and its mathematical and computer science foundations, application of formal logic, artificial intelligence, machine learning and data mining to diagnostics and anomaly detection, software diagnostics engineering and diagnostics-driven development, diagnostics workflow and interaction. Recent interest areas also include cloud native computing, security, automation, functional programming, and applications of category theory to software development and big data.

Presentation Slides and Transcript

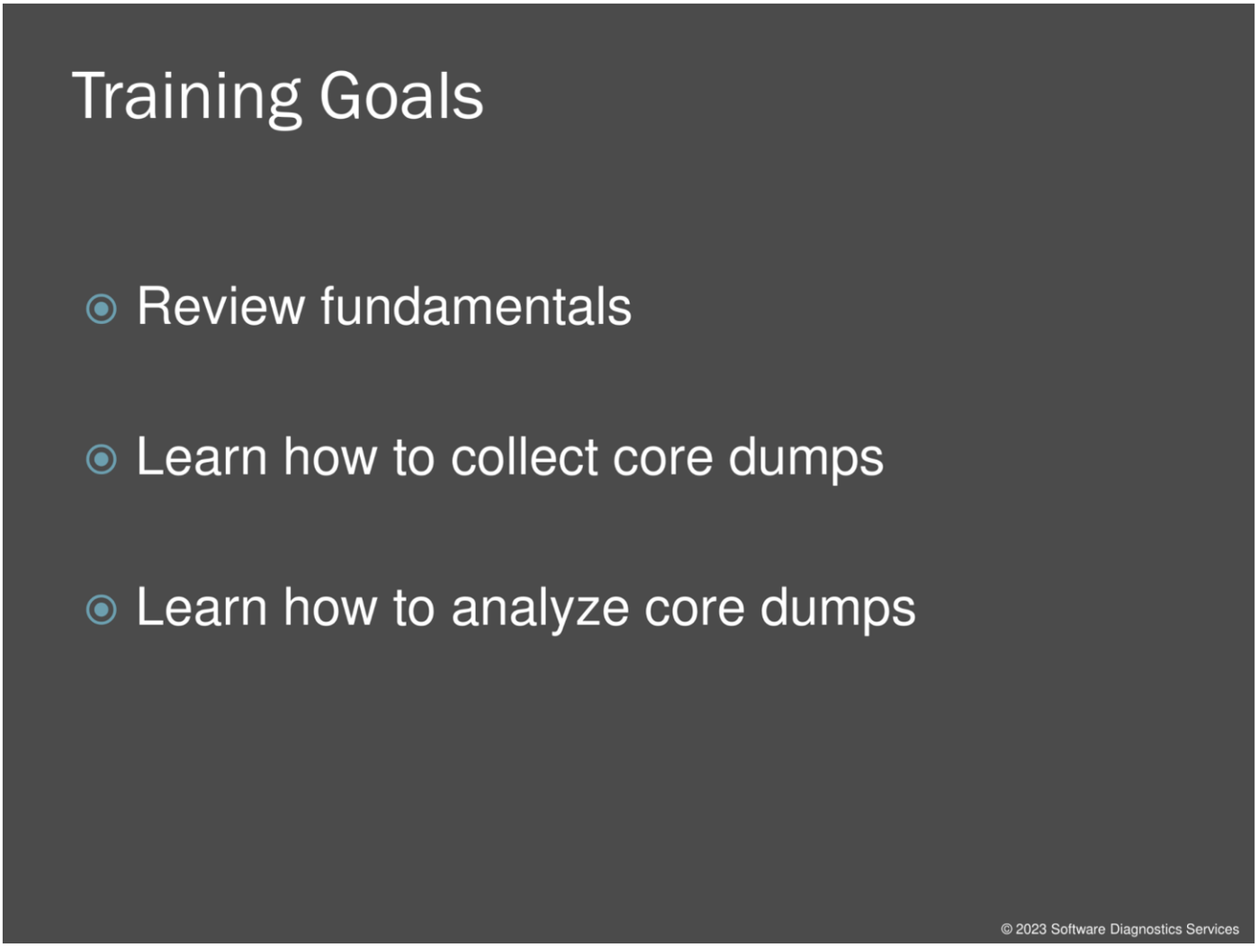


Hello, everyone, my name is Dmitry Vostokov, and I teach this training course.

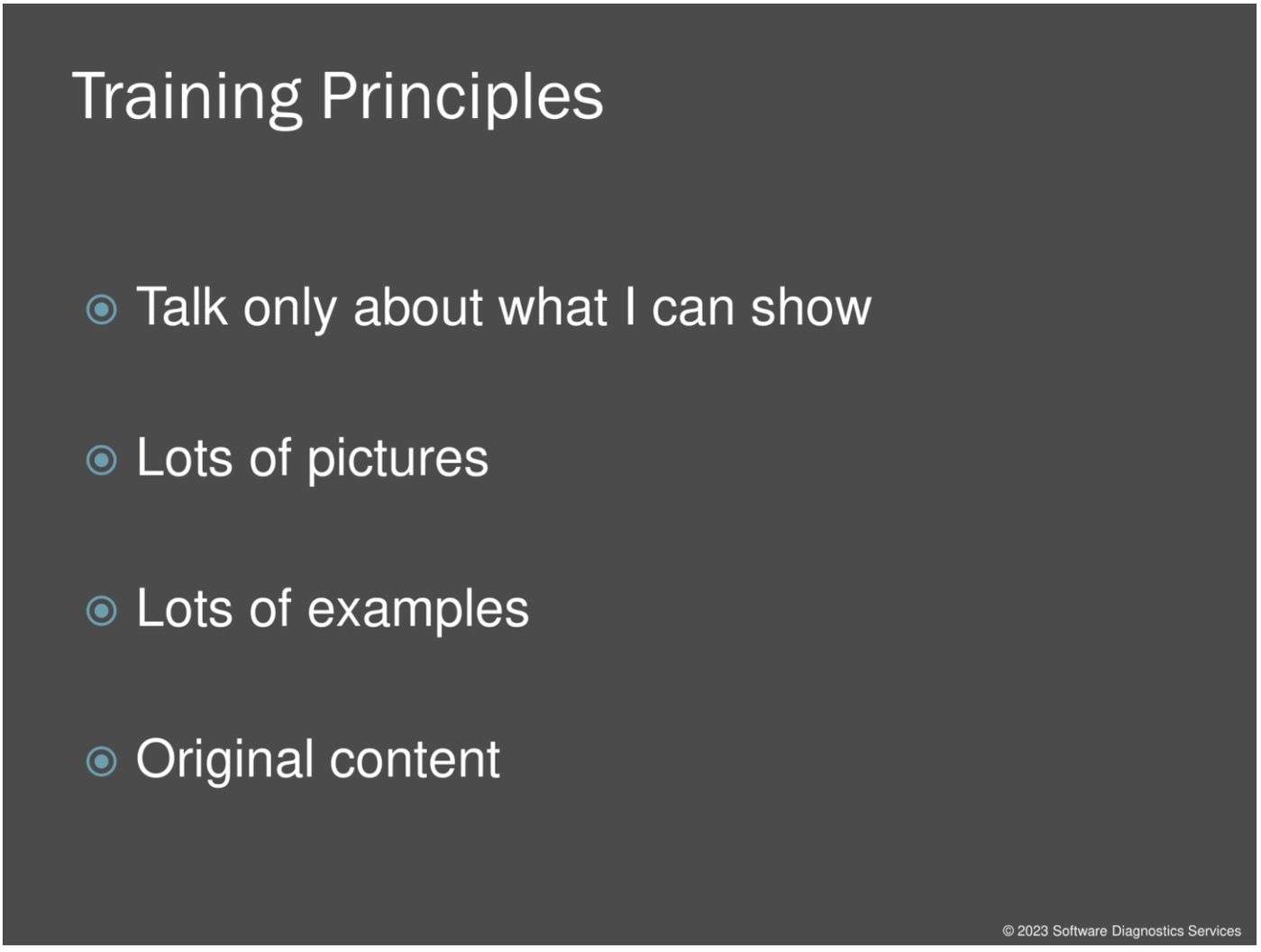


The prerequisites are hard to define. Some of you have software development experience, and some do not. However, one thing is certain that to get most of this training, you are expected to have basic troubleshooting experience. Another thing I expect you to be familiar with is hexadecimal notation and that you have seen or can read programming source code in some language. The ability to read assembly language has some advantages but is not really necessary for this training. Windows memory dump analysis experience may help ease the transition but is not absolutely necessary. If you have read either **Accelerated macOS Core Dump Analysis** or **Accelerated Windows Memory Dump Analysis** book or both, you may find a similar approach here. You may also find the additional Linux assembly language books useful:

Foundations of Linux Debugging, Disassembling, and Reversing
https://www.patterndiagnostics.com/practical-foundations-linux-debugging-disassembling-reversing
Foundations of ARM64 Linux Debugging, Disassembling, and Reversing
https://www.patterndiagnostics.com/practical-foundations-arm64-linux-debugging-disassembling-reversing



Our primary goal is to learn core dump analysis in an accelerated fashion. So first, we review absolutely essential fundamentals necessary for core dump analysis. Also, this training is mostly about user process core dump analysis with an accelerated transition to kernel core dump analysis, a topic fully explored in the follow-up course and book **Advanced Linux Core Dump Analysis with Data Structures**. An additional goal is to leverage Windows or macOS debugging and memory dump analysis experience you may have.



For me, there were many training formats to consider, and I decided that the best way is to concentrate on hands-on exercises. Specifically, for this training, I developed more than 40 of them, and they utilize the same pattern-oriented approach I used in **Accelerated Windows Memory Dump Analysis** and **Accelerated macOS Core Dump Analysis** training.

Schedule Summary

Day 1

- Analysis fundamentals (25 minutes)
- Process core dump collection (5 minutes)
- Basic x64 assembly language review (30 minutes)
- Process GDB core dump analysis (1 hour)

Day 2

- Process GDB core dump analysis (2 hours)

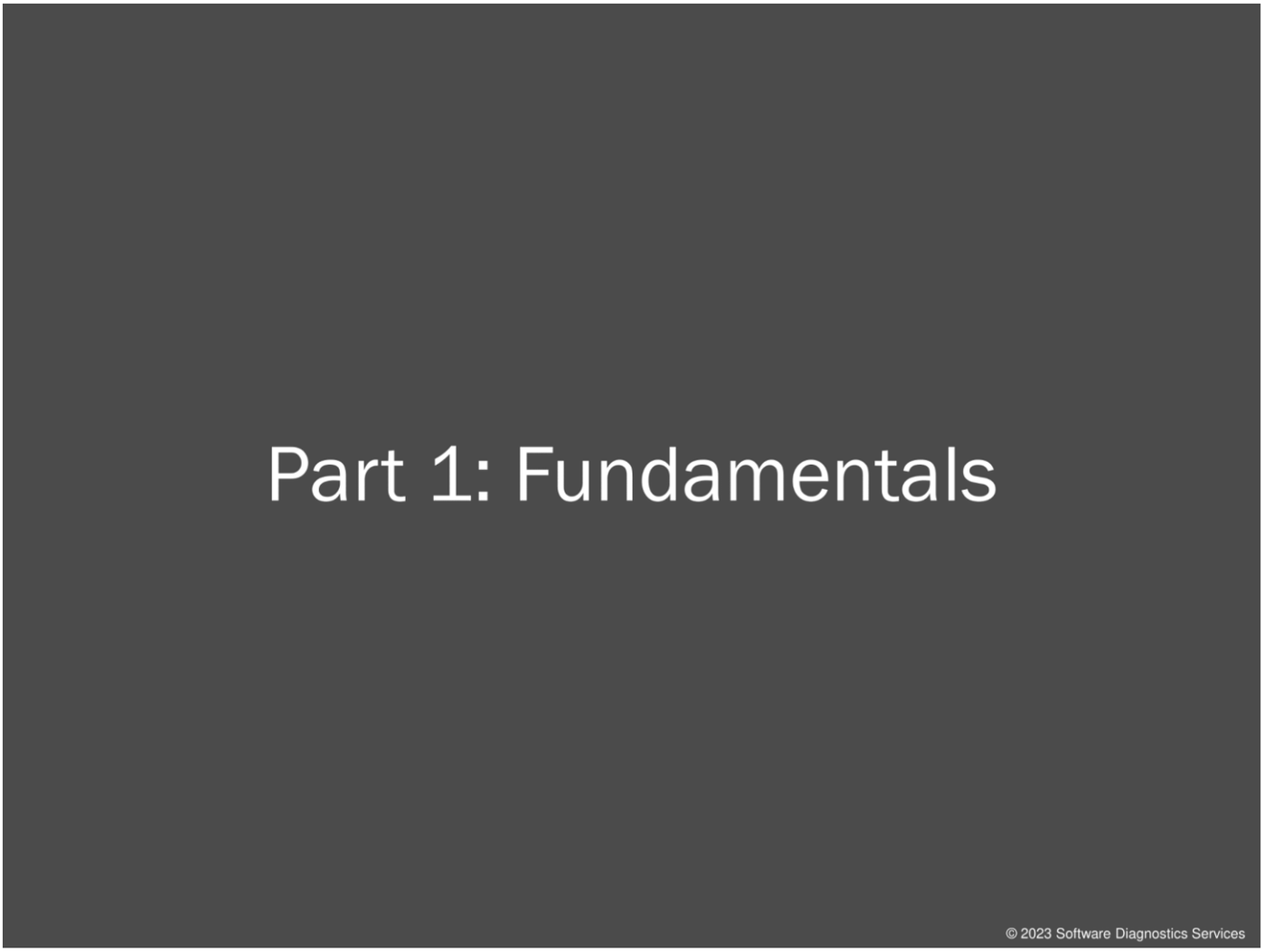
Day 3

- Kernel core dump collection (5 minutes)
- Kernel core dump analysis (1 hour 55 minutes)

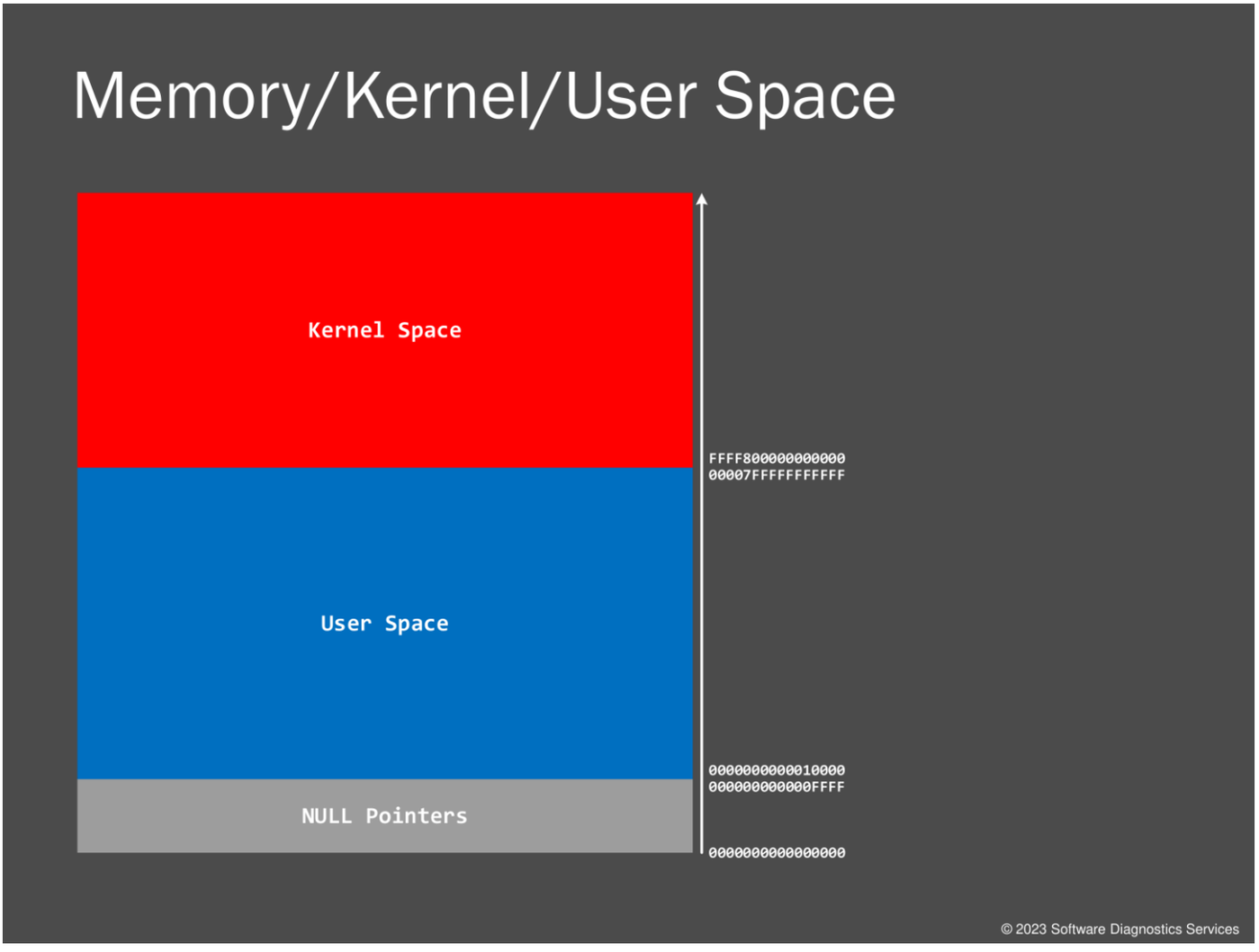
Day 4

- Basic ARM64 assembly language review (30 minutes)
- Process GDB core dump analysis (1 hour 30 minutes)
- [Optional] Process WinDbg core dump analysis

This slide shows a roughly planned schedule subject to changes as we go. Changes from the previous edition are also highlighted. If we finish a particular topic earlier, we start the next one to make more room for the ARM64 section.

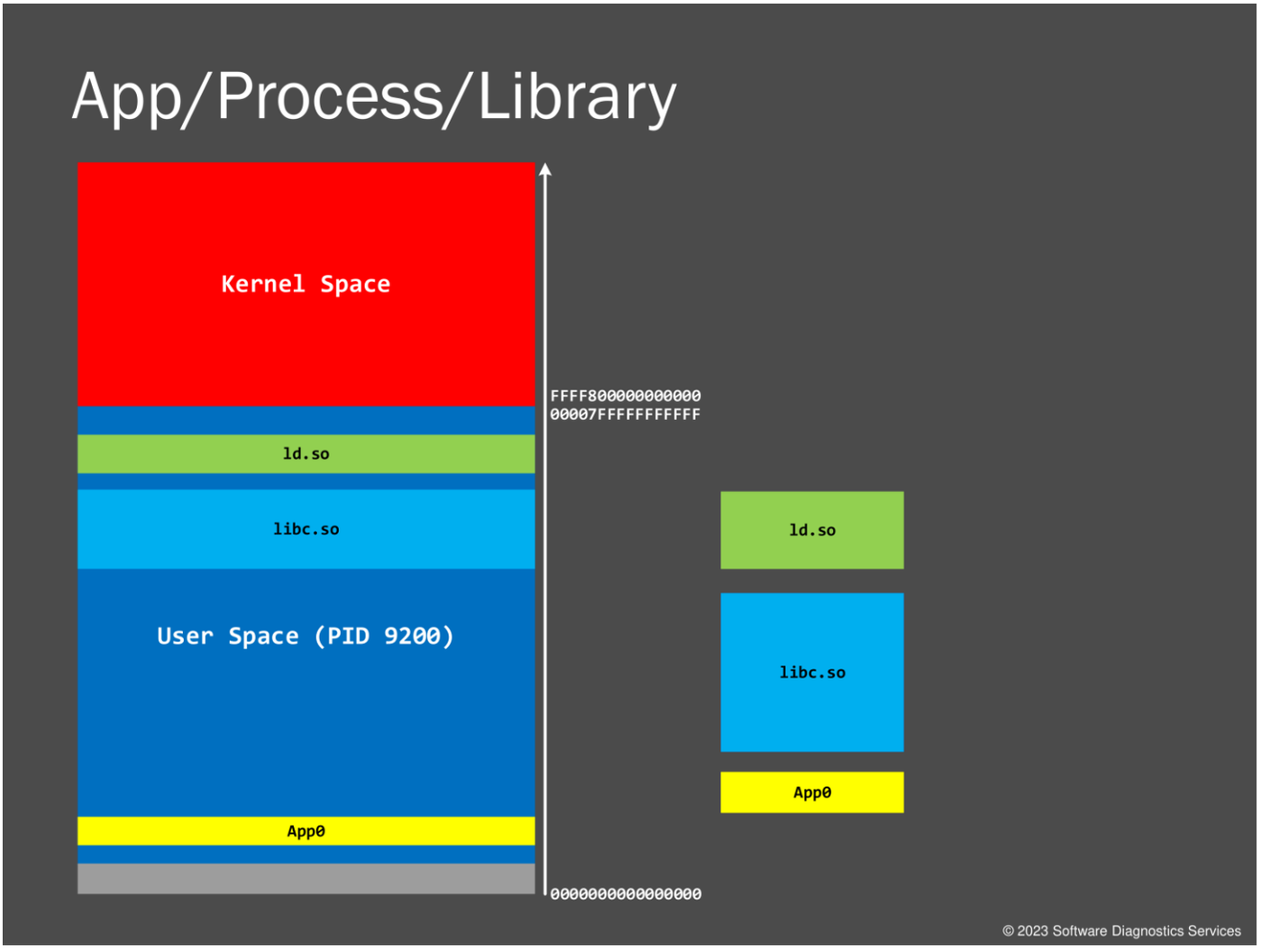


Now, I show you some pictures. I use 64-bit examples. Most of the time, fundamentals do not change when we move to 32-bit Linux, and the analysis process is mostly the same.

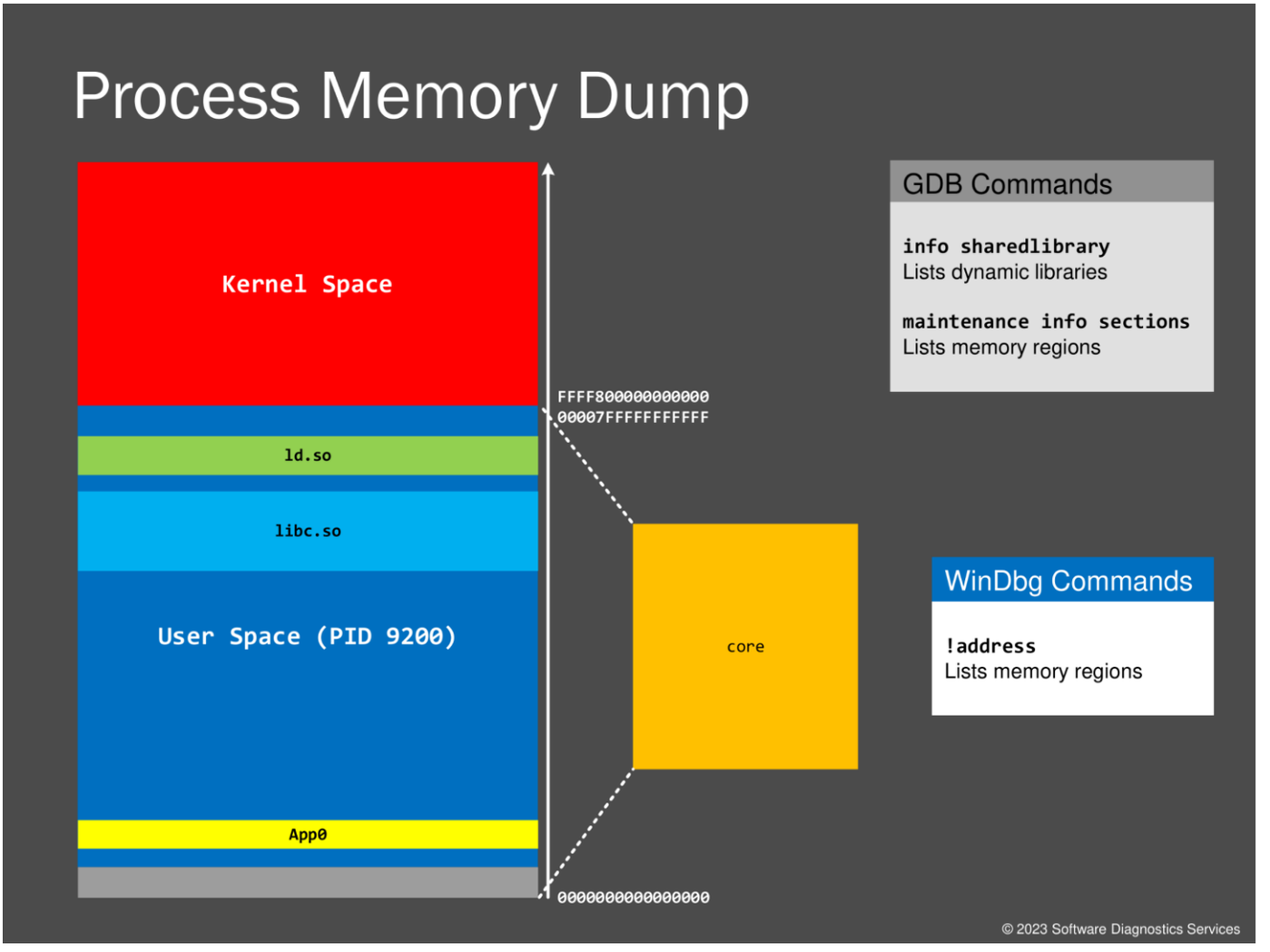


If you come from Windows or macOS background, you find fundamentals almost the same. For every process, the Linux memory range is divided into kernel and user space parts and an inaccessible part for catching null pointers[1]. This non-accessible region is different from macOS, where it is 1 GB. I follow the long tradition of using red for the kernel and blue for the user part. Please note that there is a difference between space and mode. The mode is the execution privilege attribute; for example, code running in kernel space has a higher execution privilege than code running in user space. However, kernel code can access user space and access data there. We say that such code is running in kernel mode. On the contrary, the application code from user space is running in user mode, and because of its lower privilege, it cannot access kernel space. This division prevents accidental kernel modifications. Otherwise, you could easily crash your system. I put addresses on the right. This uniform memory space is called virtual process space because it is an abstraction that allows us to analyze core dumps without thinking about how it is all organized in physical memory. When we look at process dumps, we are concerned with virtual space only.

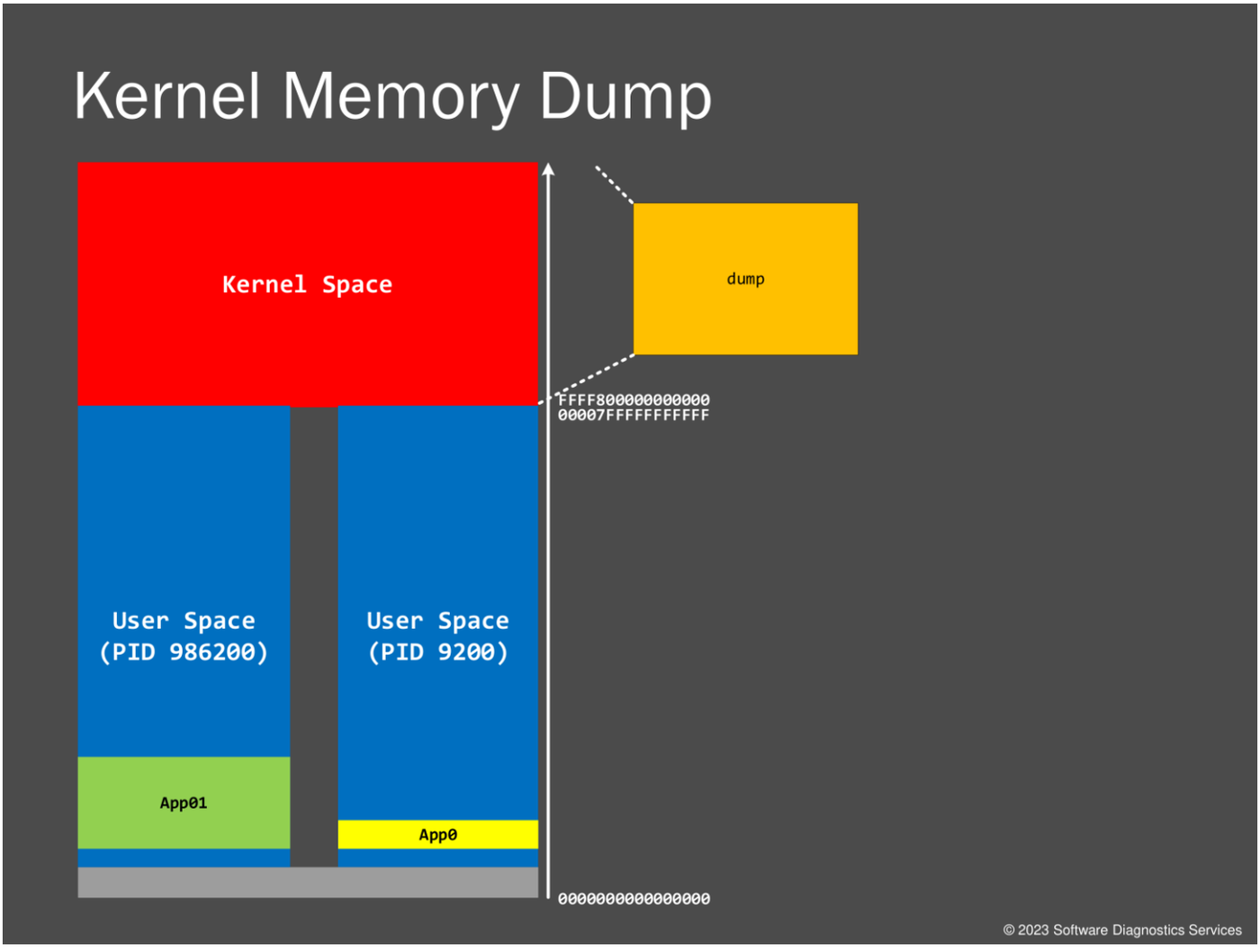
[1] On my Debian system it is 0xFFFF, as seen from /proc/sys/vm/mmap_min_addr value.



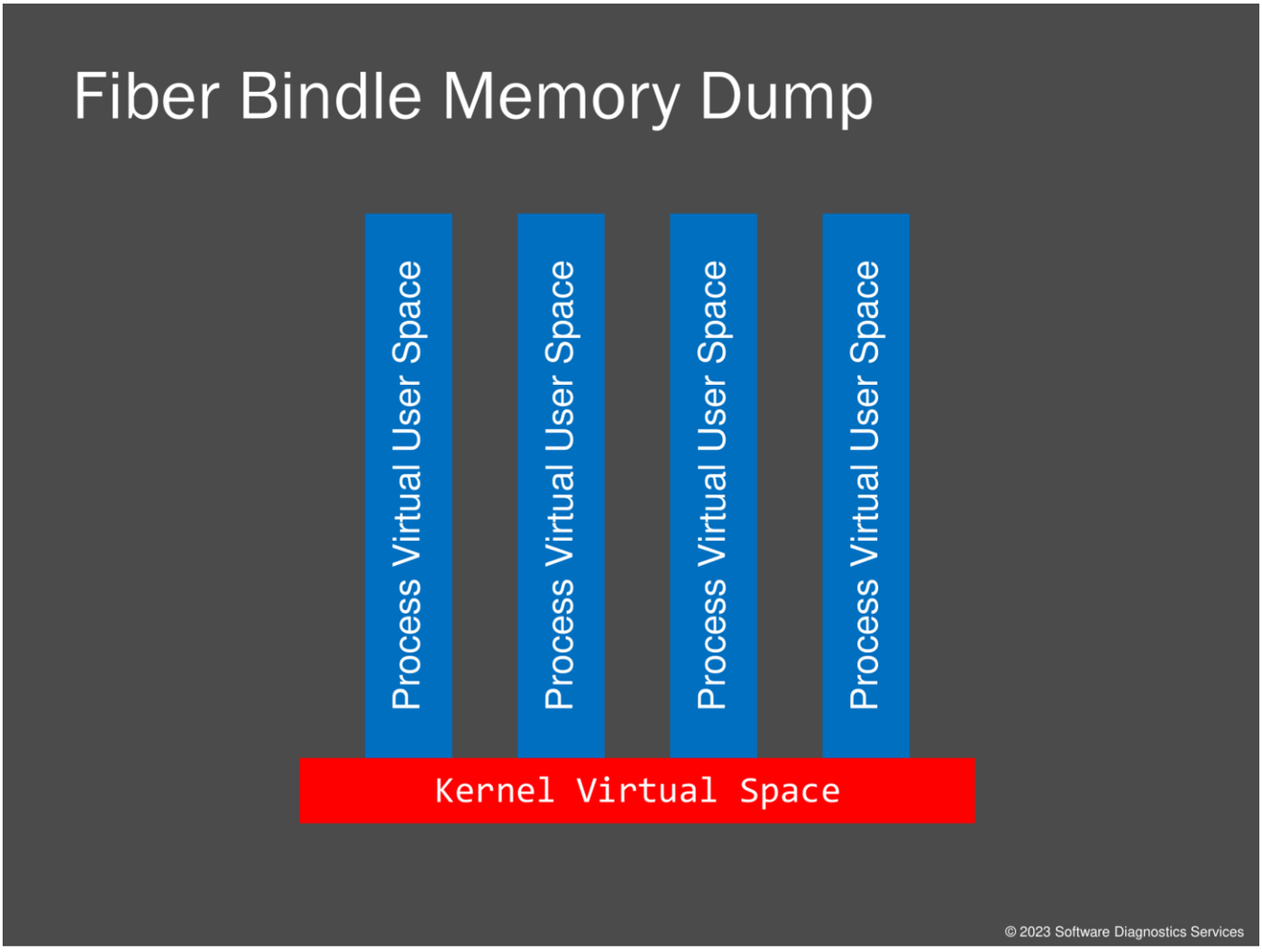
When an app is loaded, all its referenced dynamic libraries are mapped to virtual memory space. Different sections of the same file (like code and data) may be mapped into a different portion of memory. In contrast, modules in Windows are organized sequentially in virtual memory space. A process is then set up for running, and a process ID is assigned to it. If you run another such app, it has a different virtual memory space.



When we save a process core memory dump, a user space portion of the process space is saved without any kernel space stuff. However, we never see such large core dumps unless we have memory leaks. This is because process space has gaps unfilled with code and data. These unallocated parts are not saved in a core dump. However, if some parts were paged out and reside in a page file, they are usually brought back before saving a core dump.

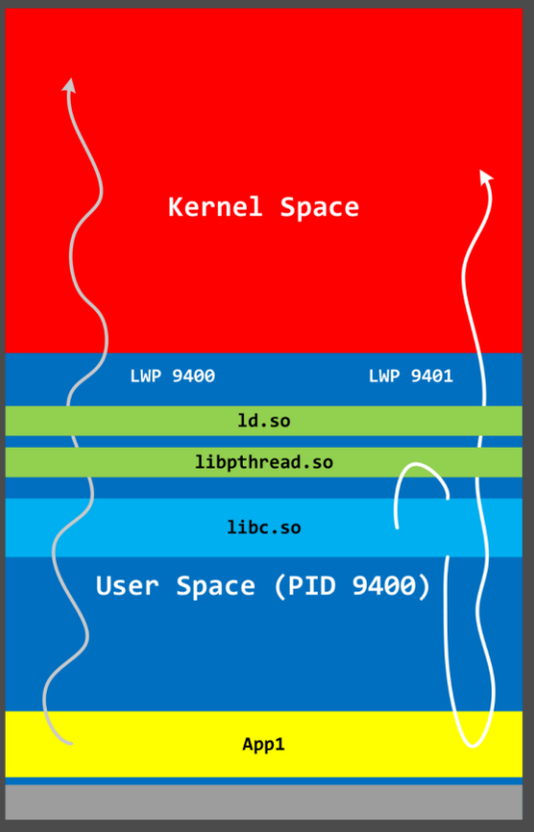


In case of a kernel panic, a kernel memory dump is saved if the appropriate mechanism is configured (mostly by default for recent distributions, such as Ubuntu). Virtual memories of running processes are not saved, however. For that, you need various physical memory acquisition methods and tools that are outside the scope of this course.



The lack of complete memory dumps may be circumvented by dumping individual processes and then forcing a kernel memory dump to analyze together. We call the resulting dump type **Fiber Bundle**.

Lightweight Processes (Threads)



GDB Commands

`info threads`
Lists threads

`thread <n>`
Switches between threads

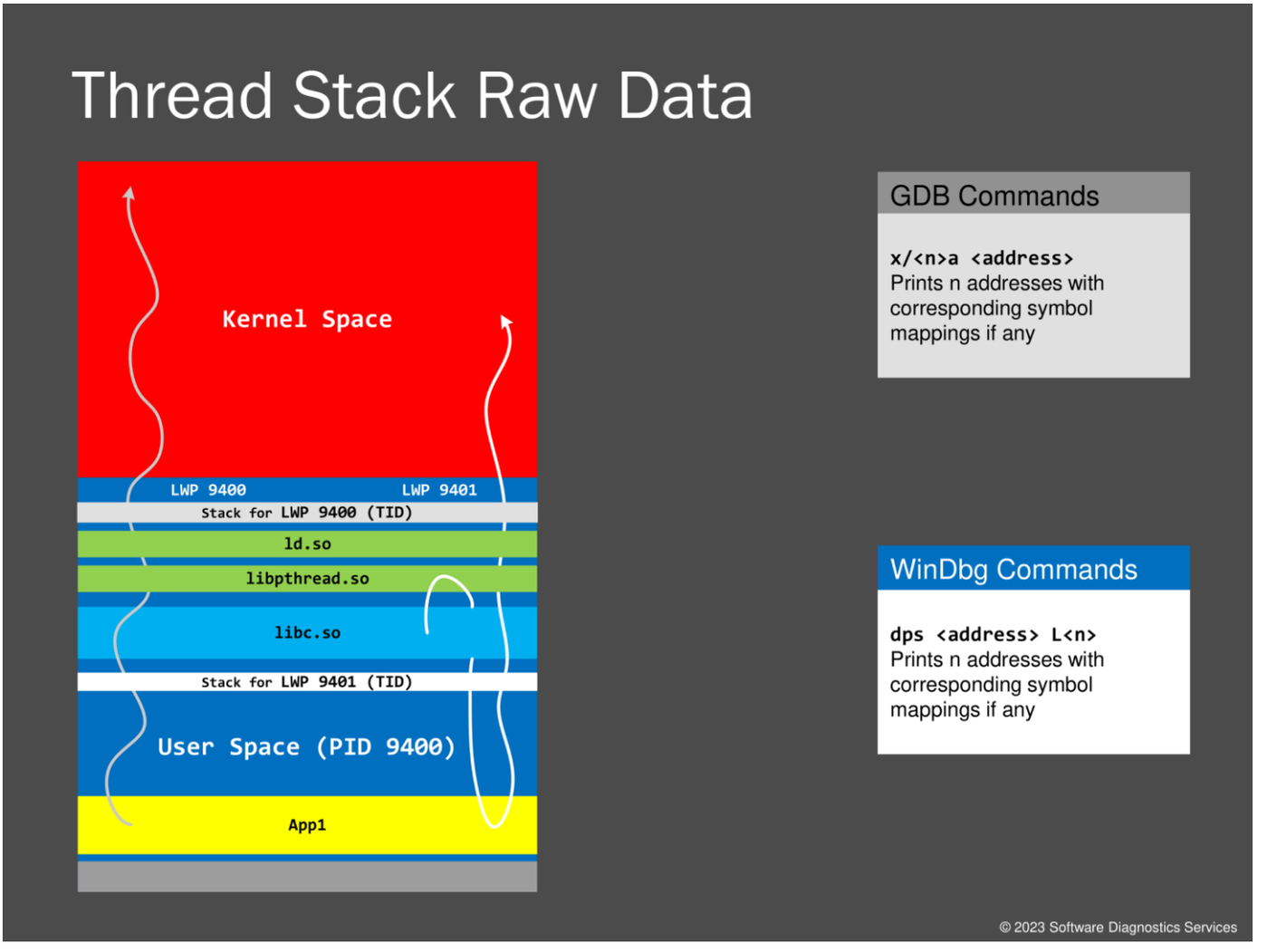
`thread apply all bt`
Lists stack traces from all threads

WinDbg Commands

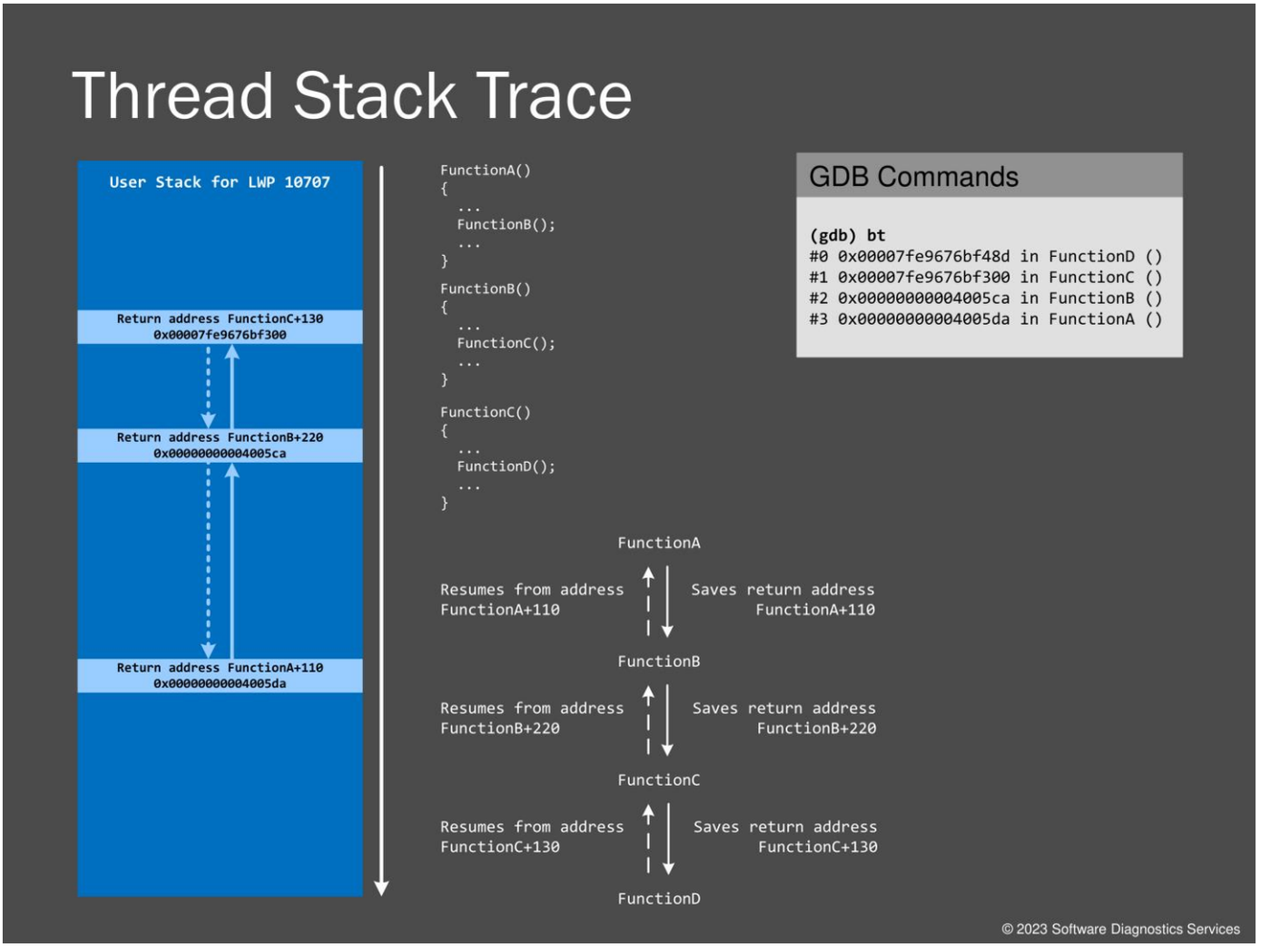
`~*k`
Lists stack traces from all threads

`~<n>s`
Switches between threads

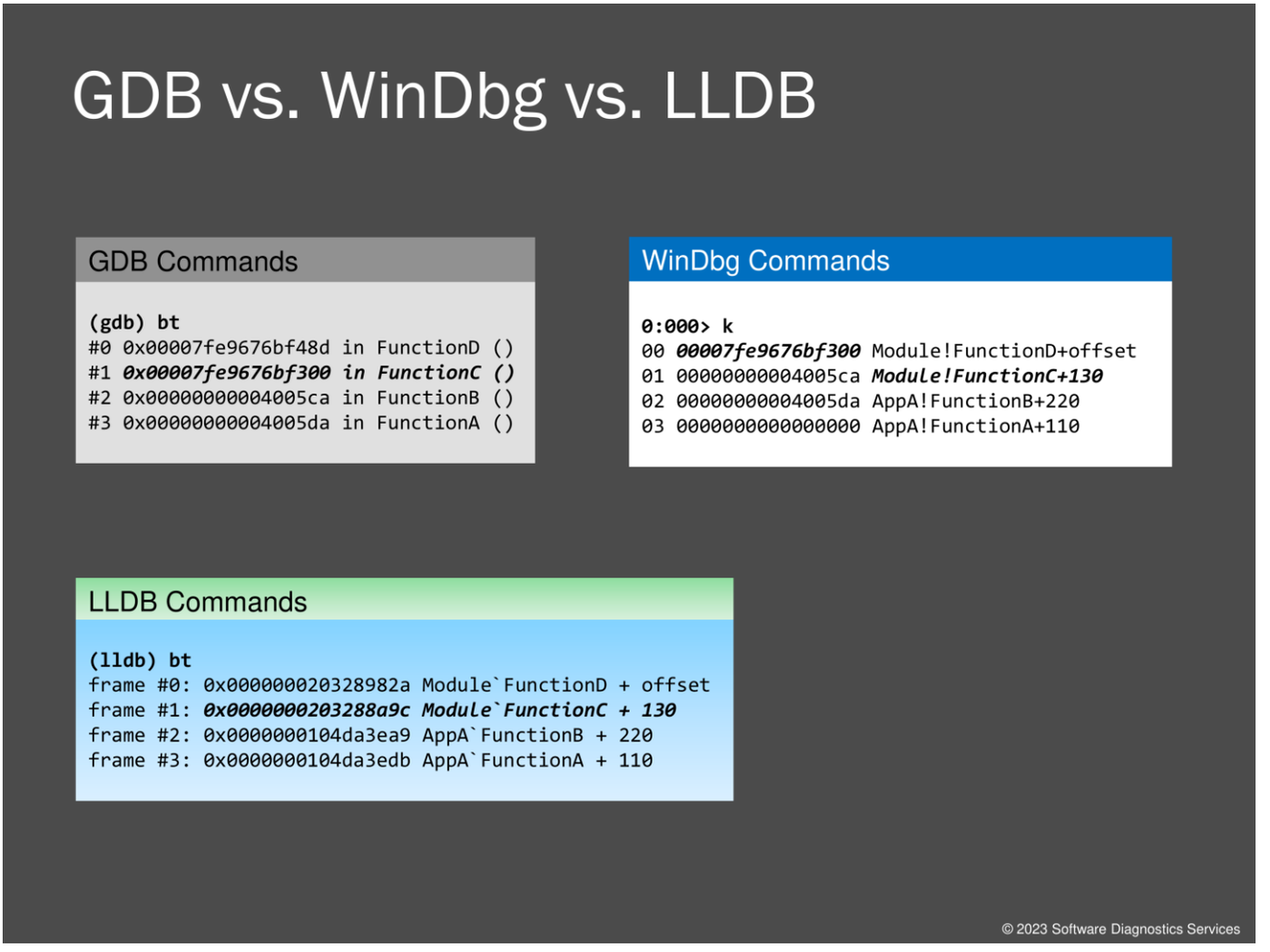
Now, we come to another important fundamental concept in Linux core dump analysis: a thread or lightweight process (LWP). It is basically a unit of execution, and there can be many threads (LWPs) for a given process (all of them share the same process space). Every thread just executes some code and performs various tasks. Every thread has its ID (LWP ID). In this training, we also learn how to navigate between process threads. Note that threads transition to kernel space via *libc* dynamic library similar to *ntdll* on Windows and *libsystem_kernel* in macOS. Threads additional to the main thread (POSIX Threads) originate from *libc* and *libpthread* dynamic libraries similar to *libsystem_c* in macOS.



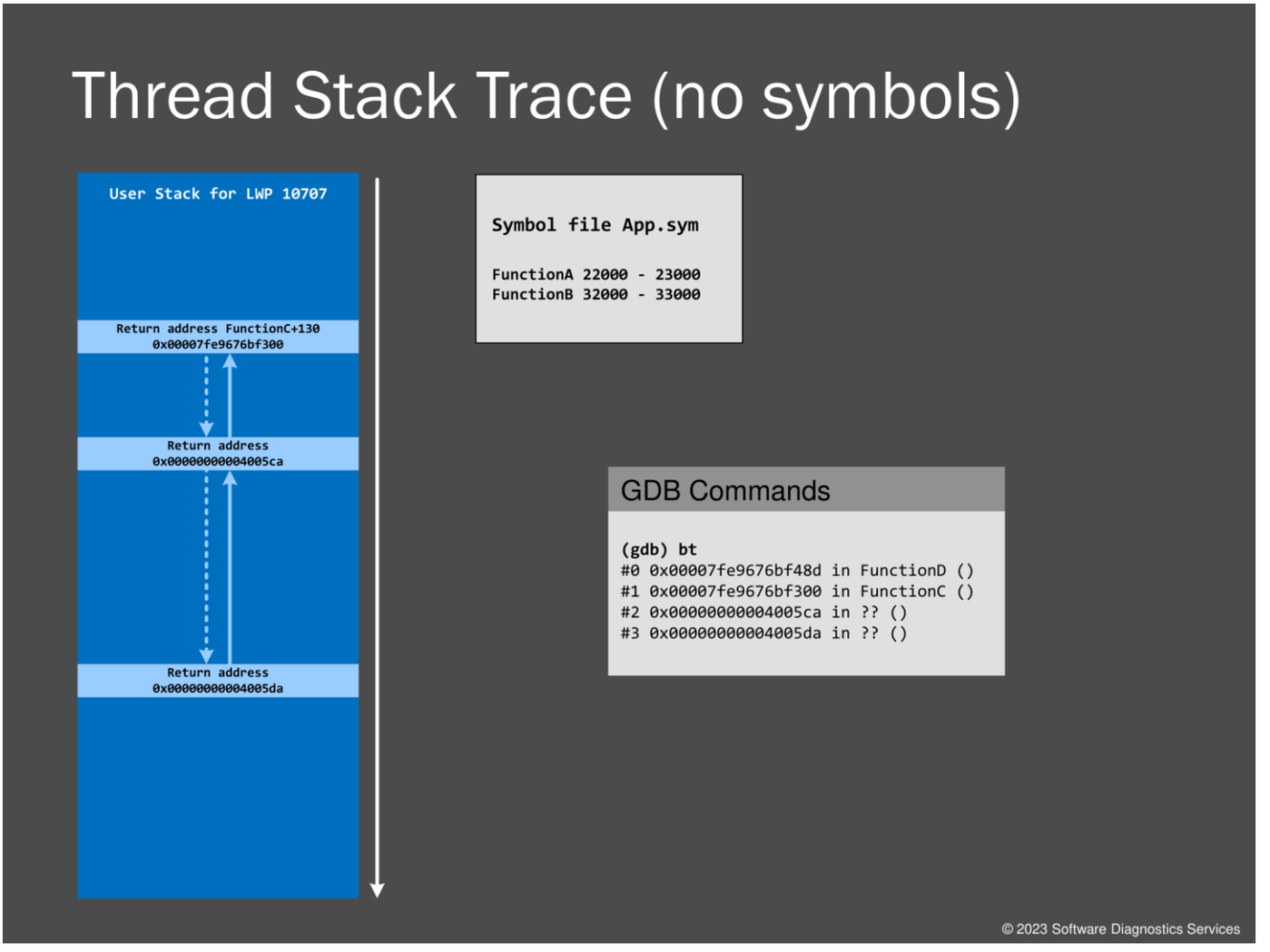
Every thread needs a temporary memory region to store its execution history and temporary data. This region is called a thread stack. Please note that the stack region is just any other memory region, and you can use any GDB data dumping commands there. We also learn how to get the address range of a thread stack region. Examining raw stack data can hint at the past process and kernel behavior: the so-called **Execution Residue** pattern.



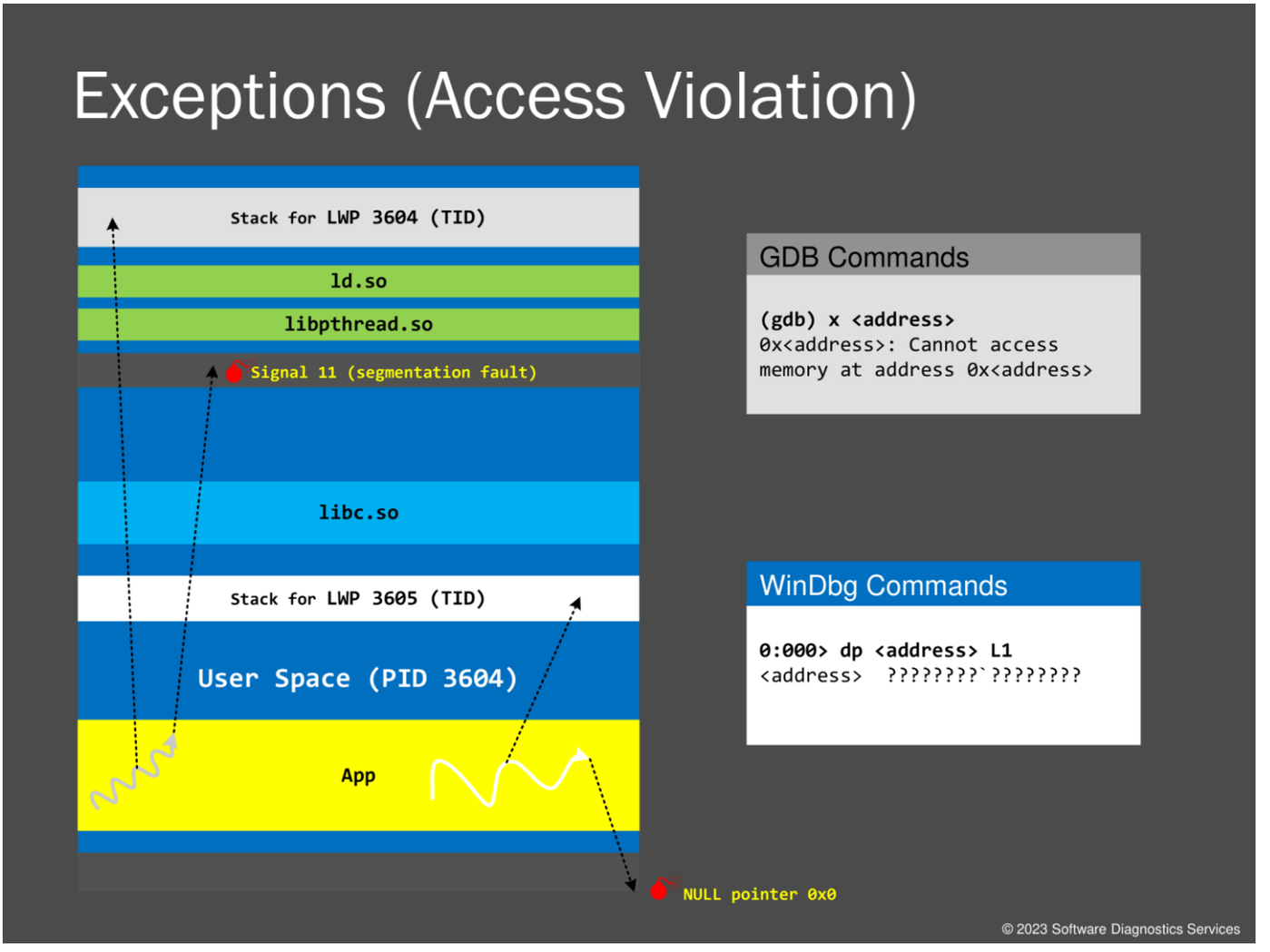
Now we explain thread stack traces. Suppose we have source code where *FunctionA* calls *FunctionB* at some point, *FunctionB* calls *FunctionC*, and so on. This sequence is called a thread of execution. If *FunctionA* calls *FunctionB*, you expect the execution thread to return to the same place where it left, and to resume from there. This goal is achieved by saving a return address in the thread stack region. So every return address is saved and then restored during the course of thread execution. Although the memory addresses grow from top to bottom in this picture, return addresses are saved from bottom to top: the stack grows from higher to lower addresses. This picture might seem counter-intuitive to all previous pictures, but this is how you see the output from GDB commands. What GDB does when you instruct it to dump a backtrace from a given thread is to analyze the thread raw stack data and figure out return addresses, map them to a symbolic form according to symbol files and show them from top to bottom. Note that *FunctionD* is not present in the raw stack data on the left because it is a currently executing function called from *FunctionC*. However, *FunctionC* called *FunctionD*, and the return address of *FunctionC* was saved. In the box on the right, we see the result of the GDB **bt** command.



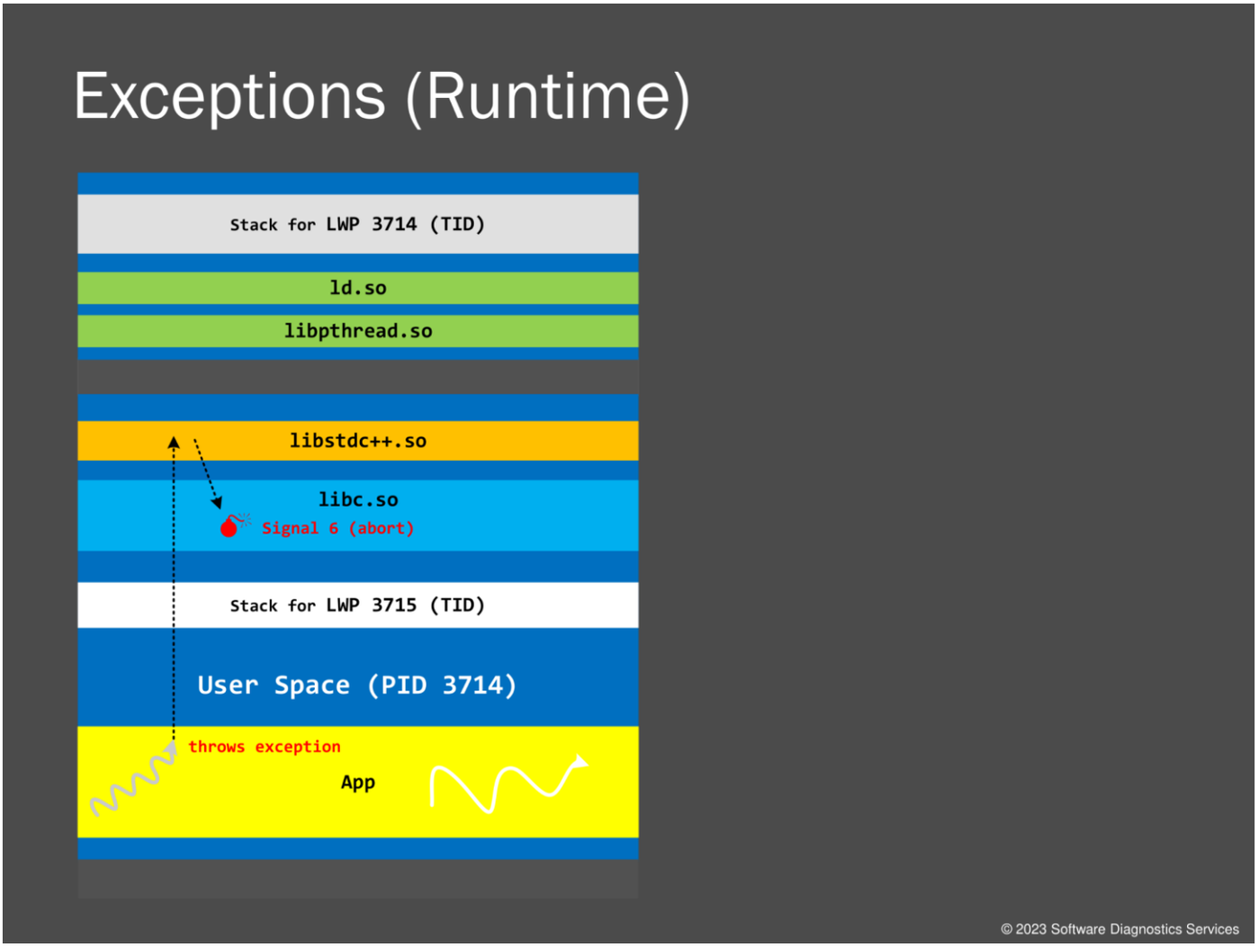
The difference from WinDbg (from Debugging Tools for Windows) here is that the return address is on the same line for the function to return (except for *FunctionD*, where the address is the next instruction to execute), whereas in WinDbg, it is for the function on the next line.



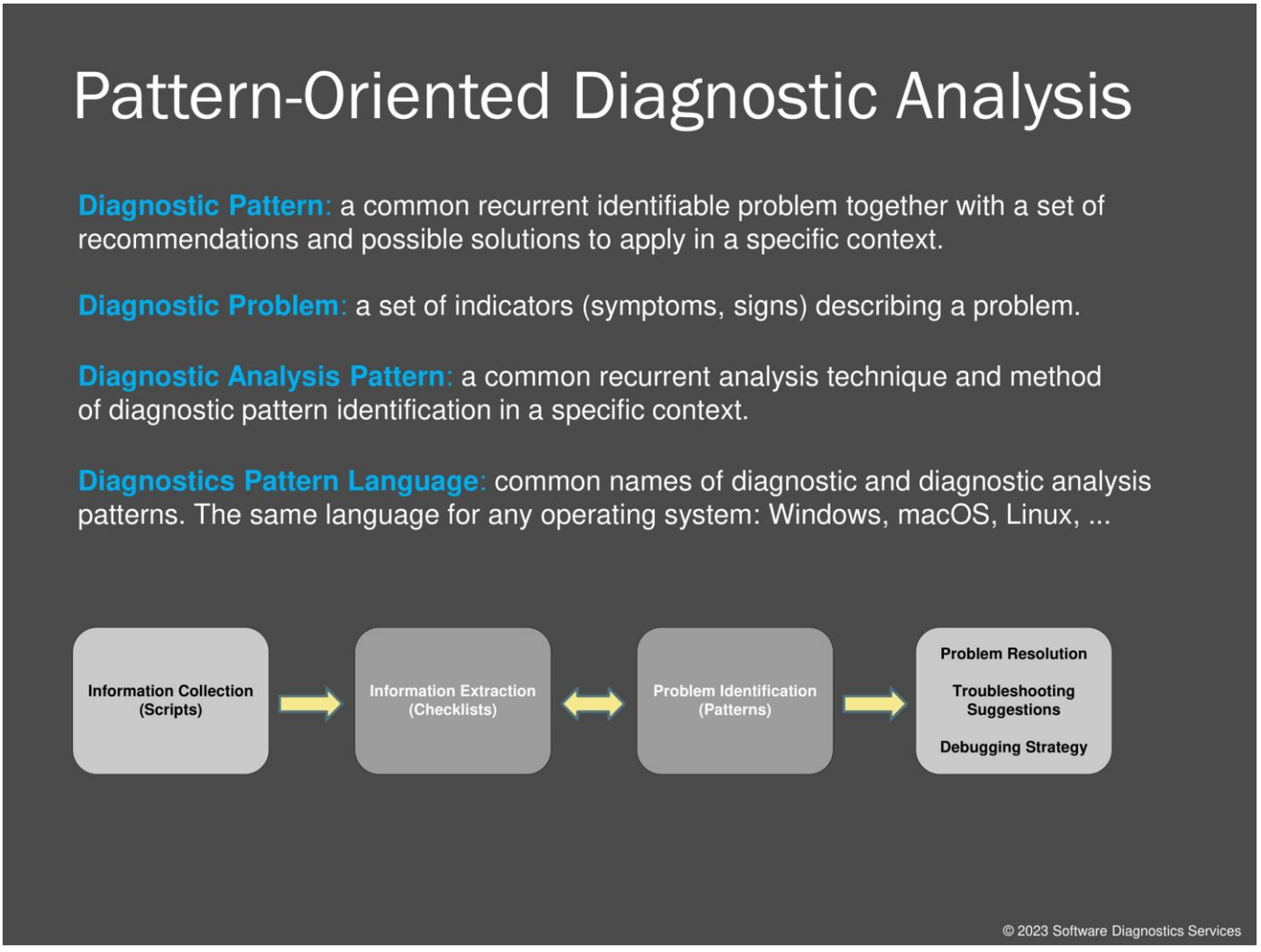
Here I'd like to show you why symbol files are important and what stack traces you get without them. Symbol files just provide mappings between memory address ranges and associated symbol names like the table of contents in a book. So in the absence of symbols, we are left with bare addresses that are saved in a dump. For example, without App symbols, we have the output shown in the box on the right.



Now we talk about access violation exceptions. During the thread execution, it accesses various memory addresses doing reads and writes. Sometimes memory is not present due to gaps in virtual address space or different protection levels like read-only or no-execute memory regions. If a thread tries to violate that, we get an exception that is also translated to a traditional UNIX signal. Certain regions are forbidden to read and write, such as the first 64KB. If we have such an access violation there, then it is called NULL pointer access. Note that any thread can have an exception (a victim thread in macOS). It is also sometimes the case that code can catch these exceptions preventing a user from seeing error messages. Such exceptions can contribute to corruption, and we call them hidden.

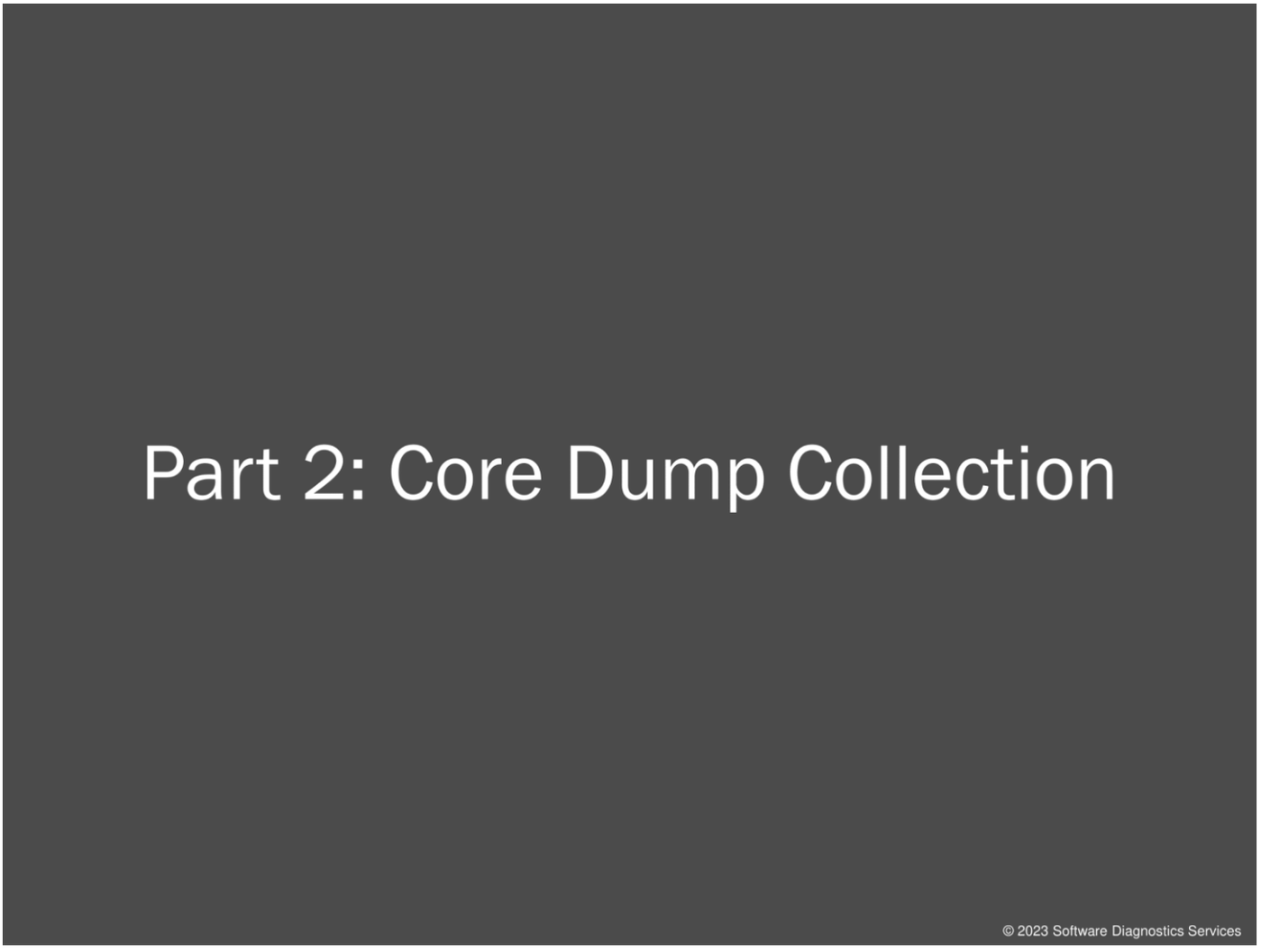


However, not all exceptions happen from invalid access. Many exceptions are generated by the code itself when it checks for some condition, and it is not satisfied, for example, when the code checks a buffer or an array to verify whether it is full before trying to add more data. If it finds it is already full, the code throws an exception translated to SIGABRT. We would see that in one of our practice examples when C++ code throws a C++ exception. Such exceptions are usually called runtime exceptions.



A few words about logs, checklists, and patterns. Core memory dump analysis is usually an analysis of a text for the presence of diagnostic patterns. We run commands, they output text, and then we look at that textual output, and when we find suspicious diagnostic indicators, we execute more commands. Here pattern and command checklists can be very useful.

Core Dump Collection



Here I'd like to show you how to collect core dumps because this option is switched off on Linux by default.

Enabling Collection (Processes)

- Temporary for the current user

```
$ ulimit -c unlimited
```

- Permanent for every user except root

Edit the file: `/etc/security/limits.conf`

Add or uncomment the line:

```
*      soft   core   unlimited
```

To limit root to 1GB, add or uncomment this line:

```
*      hard   core   1000000
```


On some systems, a process core dump is stored in the process's working directory. On other systems, you need to verify what is a configured path. We see that in the following slides.

Generation Methods (Processes)

- kill (requires ulimit)

```
$ kill -s SIGQUIT PID
$ kill -s SIGABRT PID
```

- gcore

```
$ gcore [-o filename] PID
```

- procdump

https://github.com/Sysinternals/ProcDump-for-Linux

Procdump
https://github.com/Sysinternals/ProcDump-for-Linux

Finding Core Dumps (Processes)

- Check the current core dump directory and naming pattern

```
$ cat /proc/sys/kernel/core_pattern
```

- Search

```
$ sudo find / -name core.*
```

- Further information

https://man7.org/linux/man-pages/man5/core.5.html

Core man page
https://man7.org/linux/man-pages/man5/core.5.html

Enabling Collection (Kernel)

- Uncompressed kernel image with symbols:

Debian: `$ sudo apt install linux-image-$(uname -r)-dbg`

Ubuntu: https://wiki.ubuntu.com/Kernel/Systemtap (Where to get debug symbols for kernel X?)

- Kdump (and kexec):

```
$ sudo apt install kdump-tools kexec-tools
```


Ubuntu
https://wiki.ubuntu.com/Kernel/Systemtap

Generation Methods (Kernel)

- Manual

```
$ sudo echo 1 > /proc/sys/kernel/sysrq
$ sudo echo c > /proc/sysrq-trigger
```

- Kernel modules

Finding Core Dumps (Kernel)

- Core dumps

```
/var/crash
```

- vmlinux

```
/usr/lib/debug
```

Enabling Analysis (Kernel)

- Install crash tool (depends on distribution)

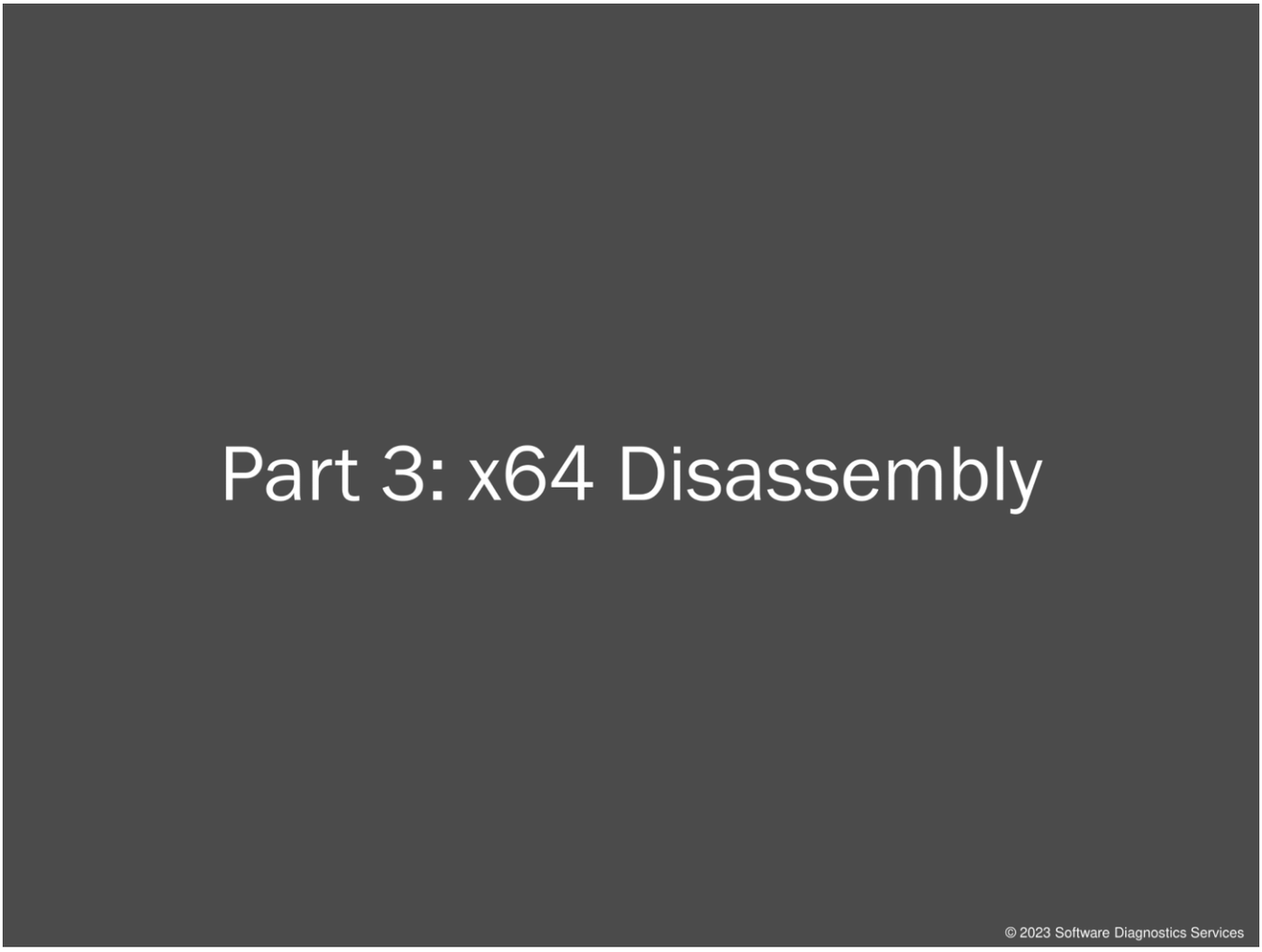
```
$ sudo apt install crash
```

- Compile crash tool from source

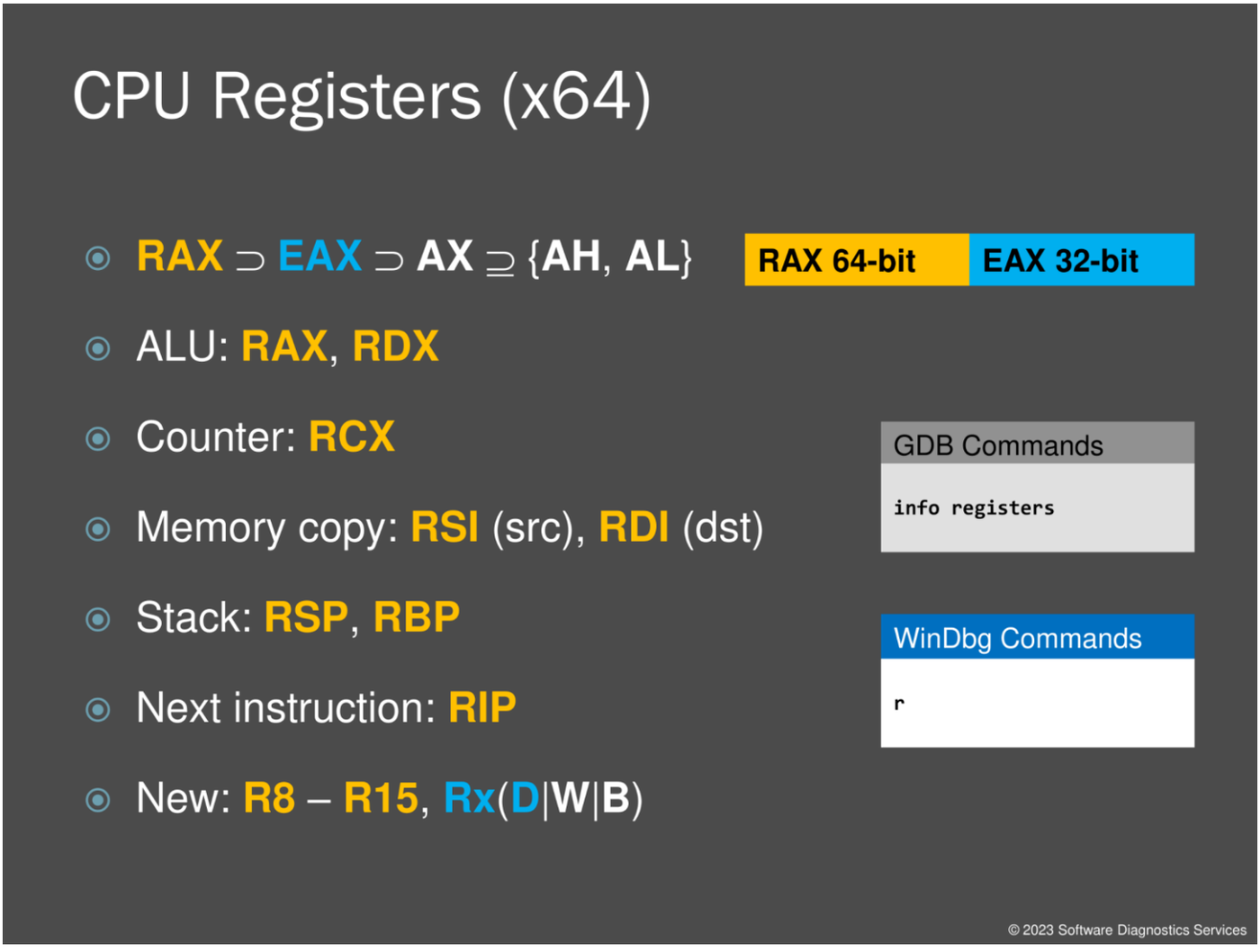
```
$ git clone https://github.com/crash-utility/crash.git
$ sudo apt install bison
$ cd crash
$ make
$ sudo make install
```


Crash tool
https://github.com/crash-utility/crash.git

x64 Disassembly



Now we come to a brief overview of relevant x64 disassembly. We only cover what we would see in the exercises.



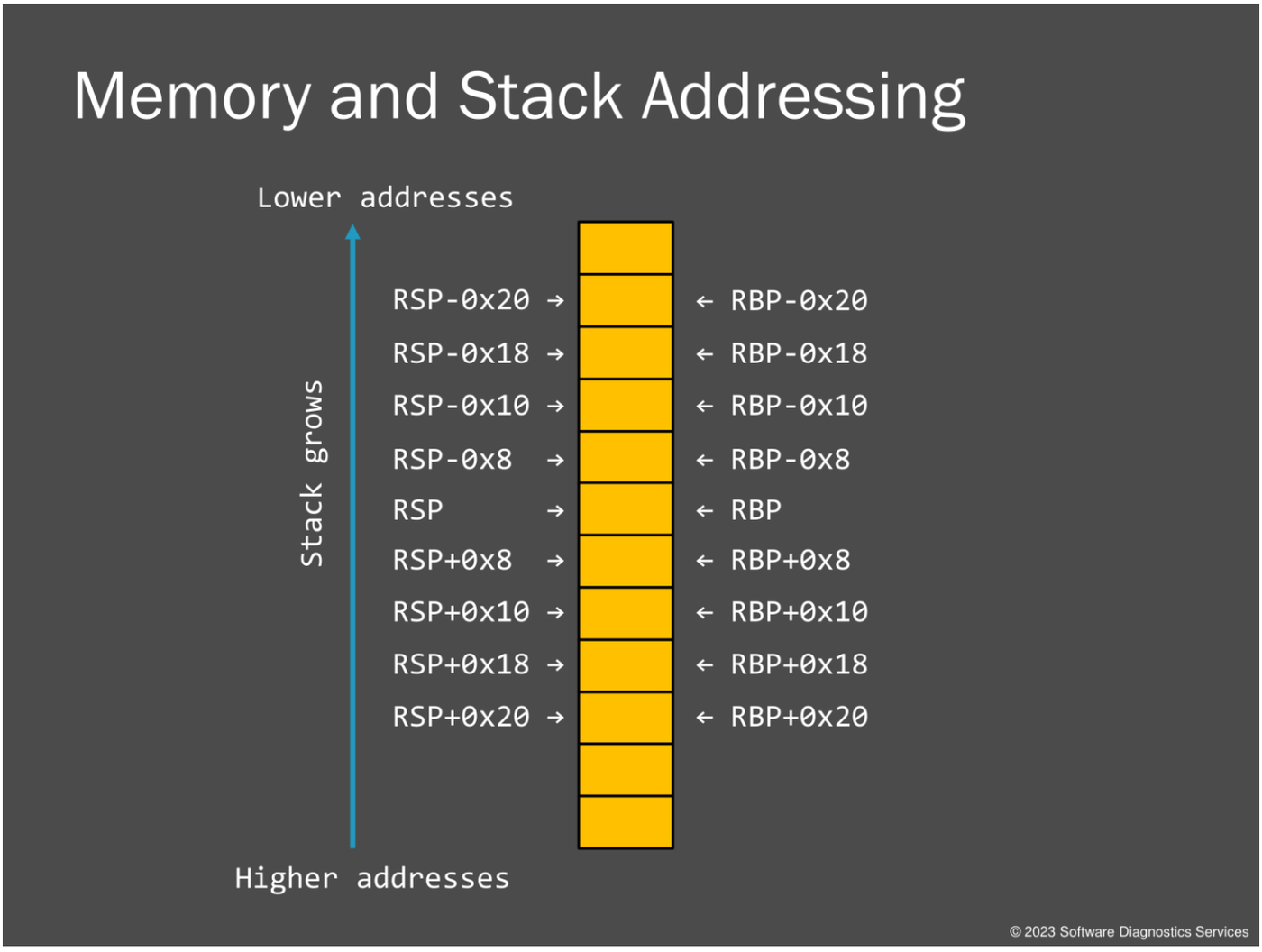
There are usual 32-bit CPU register names, such as **EAX,** that are extended to 64-bit names, such as **RAX**. Most of them are traditionally specialized, such as ALU, counter, and memory copy registers. Although, now they all can be used as general-purpose registers. There is, of course, a stack pointer, **RSP**, and, additionally, a frame pointer, **RBP**, that is used to address local variables and saved parameters. It can be used for backtrace reconstruction. In some compiler code generation implementations, **RBP** is also used as a general-purpose register, with **RSP** taking the role of a frame pointer. An instruction pointer RIP is saved in the stack memory region with every function call, then restored on return from the called function. In addition, the x64 platform features another eight general-purpose registers, from **R8** to **R15**.

Instructions: registers (x64)

- Opcode SRC, DST # default AT&T flavour
- Examples:

```
mov   $0x10, %rax          # 0x10 → RAX
mov   %rsp, %rbp           # RSP → RBP
add   $0x10, %r10          # R10 + 0x10 → R10
imul  %ecx, %edx           # ECX * EDX → EDX
callq *%rdx                # RDX already contains
                           #    the address of func (&func)
                           # PUSH RIP; &func → RIP
sub   $0x30, %rsp          # RSP-0x30 → RSP
                           # make a room for local variables
```


This slide shows a few examples of CPU instructions involving operations with registers, such as moving a value and doing arithmetic. The direction of operands is opposite to the Intel x64 disassembly flavor if you are accustomed to WinDbg on Windows. It is possible to use the Intel disassembly flavor in GDB, but we opted for the default AT&T flavor in line with our book **Foundations of Linux Debugging, Disassembly, and Reversing**.



Before we look at operations with memory, let's look at a graphical representation of memory addressing. A thread stack is just any other memory region, so instead of **RSP** and **RBP,** any other register can be used. Please note that stack grows towards lower addresses, so to access the previously pushed values, you need to use positive offsets from **RSP**.

Instructions: memory load (x64)

- Opcode Offset(SRC), DST
- Opcode DST
- Examples:

```
mov   0x10(%rsp), %rax      # value at address RSP+0x10 → RAX
mov   -0x10(%rbp), %rcx     # value at address RBP-0x10 → RCX
add   (%rax), %rdx          # RDX + value at address RAX → RDX
pop   %rdi                  # value at address RSP → RDI
                            # RSP + 8 → RSP
lea   0x20(%rbp), %r8       # address RBP+0x20 → R8
```


Constants are encoded in instructions, but if we need arbitrary values, we must get them from memory. Round brackets show memory access relative to an address stored in some register.

Instructions: memory store (x64)

- Opcode SRC, Offset(DST)
- Opcode SRC|DST
- Examples:

```
mov   %rcx, -0x20(%rbp)      # RCX → value at address RBP-0x20
addl  $1, (%rax)             # 1 + 32-bit value at address RAX →
                             #     32-bit value at address RAX
push  %rsi                   # RSP - 8 → RSP
                             # RSI → value at address RSP
inc   (%rcx)                 # 1 + value at address RCX →
                             #     value at address RCX
```

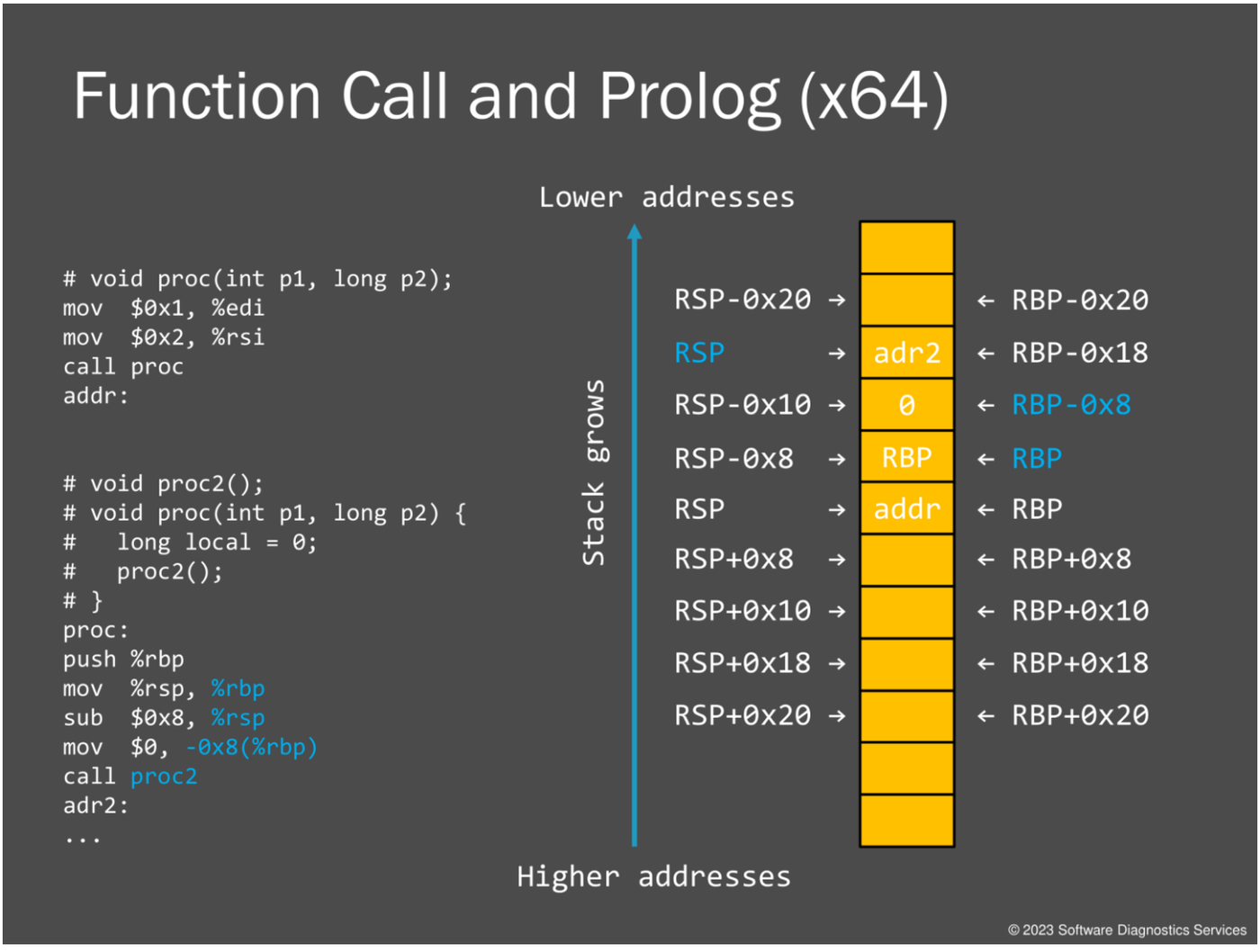

Storing is similar to loading.

Instructions: flow (x64)

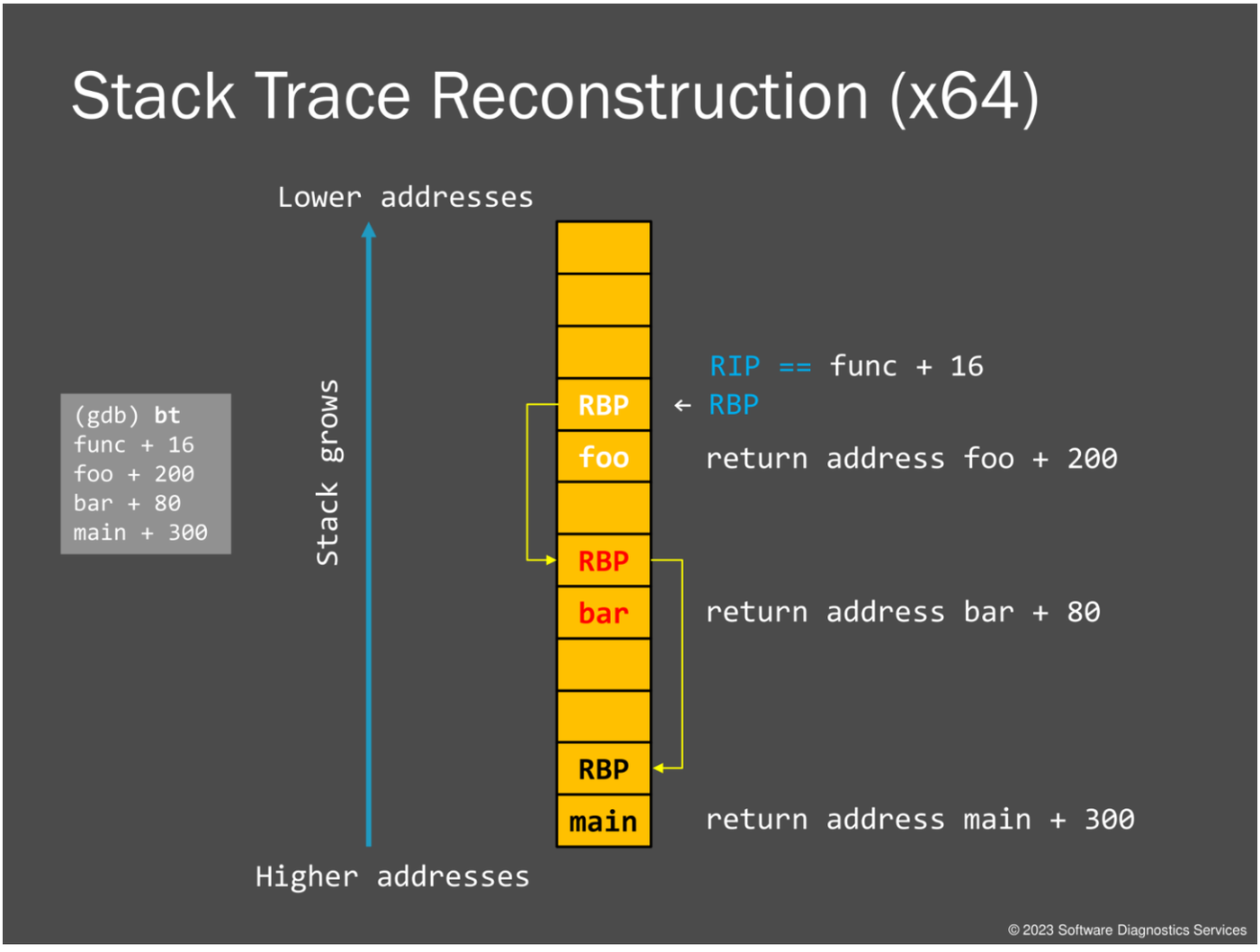
- Opcode DST
- Examples:

```
jmp     0x10493fc1c       # 0x10493fc1c → RIP
                          # (goto 0x10493fc1c)

call    0x10493ff74       # RSP – 8 → RSP
0x10493fc14:              # 0x10493fc14 → value at address RSP
                          # 0x10493ff74 → RIP
                          # (goto 0x10493ff74)
```


Goto (an unconditional jump) is implemented via the **JMP** instruction. Function calls are implemented via **CALL** instruction. For conditional branches, please look at the official documentation provided in the References slide. We don't use these instructions in our exercises.

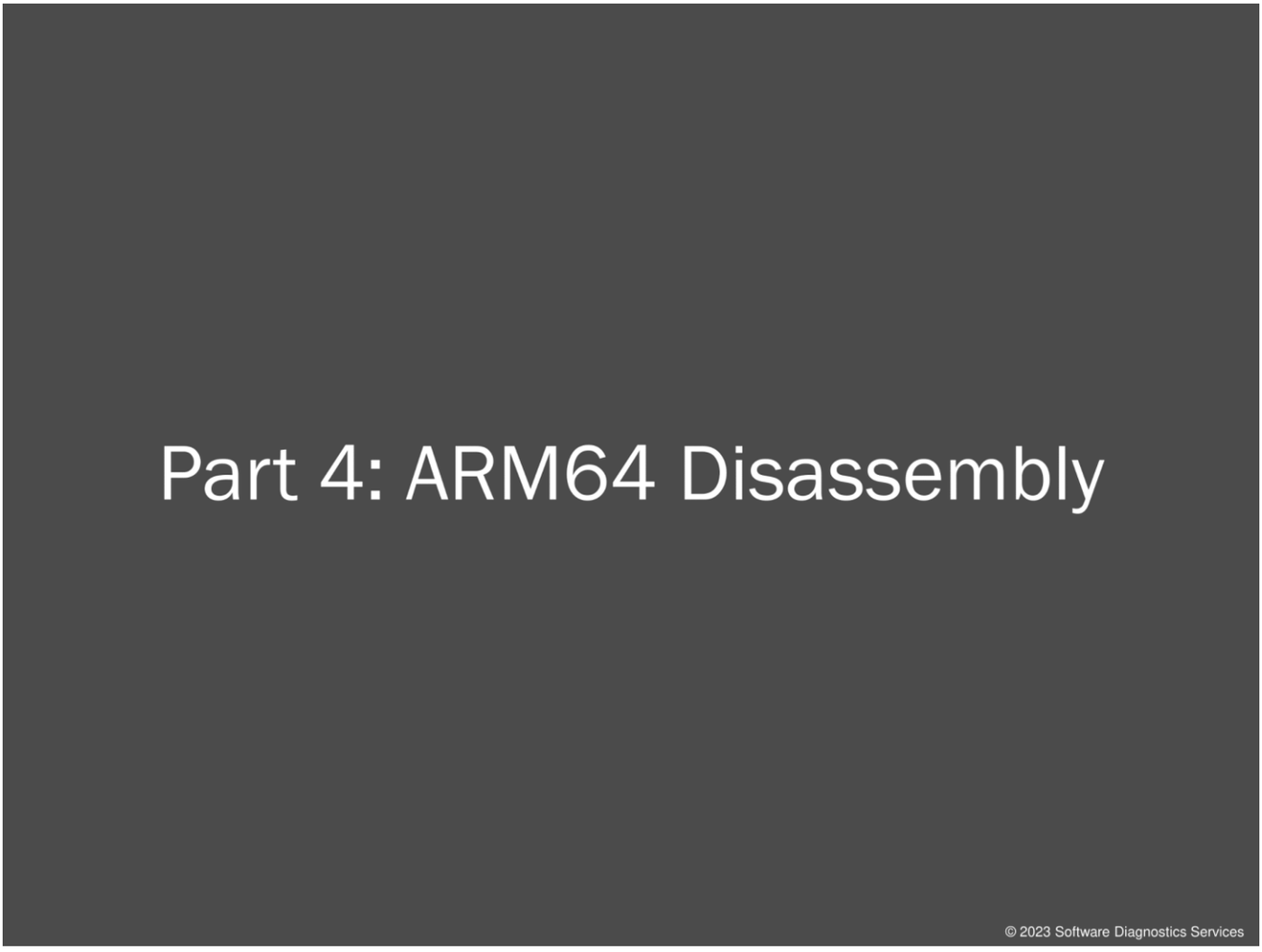


When a function is called from the caller, a callee needs to do certain operations to make room for local variables on the thread stack. There are different ways to do that, and the assembly language code on the left is one of them. I use a different color in the diagram on the right to highlight the updated **RSP** and **RBP** values before *proc2* is called. For simplicity of illustration, I only use 64-bit values.

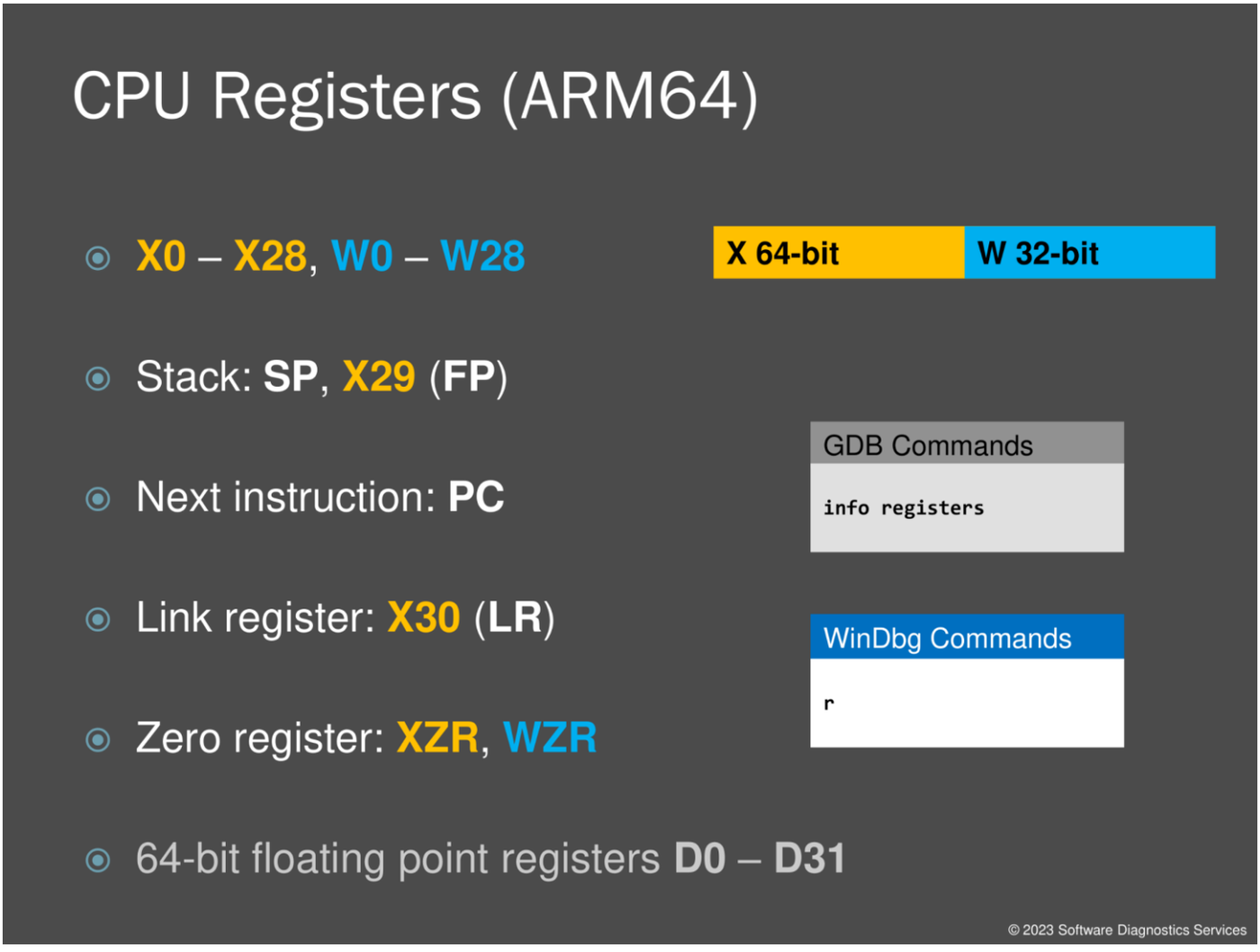


You may have noticed on the previous diagram that the new **RBP** points to the **RBP** of the caller, and below the previous **RBP** is the return address of the caller. So, if you know the **RBP** value, you can reconstruct the stack trace if the compiler follows the preceding function prolog convention.

ARM64 Disassembly



Now we come to a brief overview of relevant ARM64 disassembly. We only cover what we would see in the exercises.



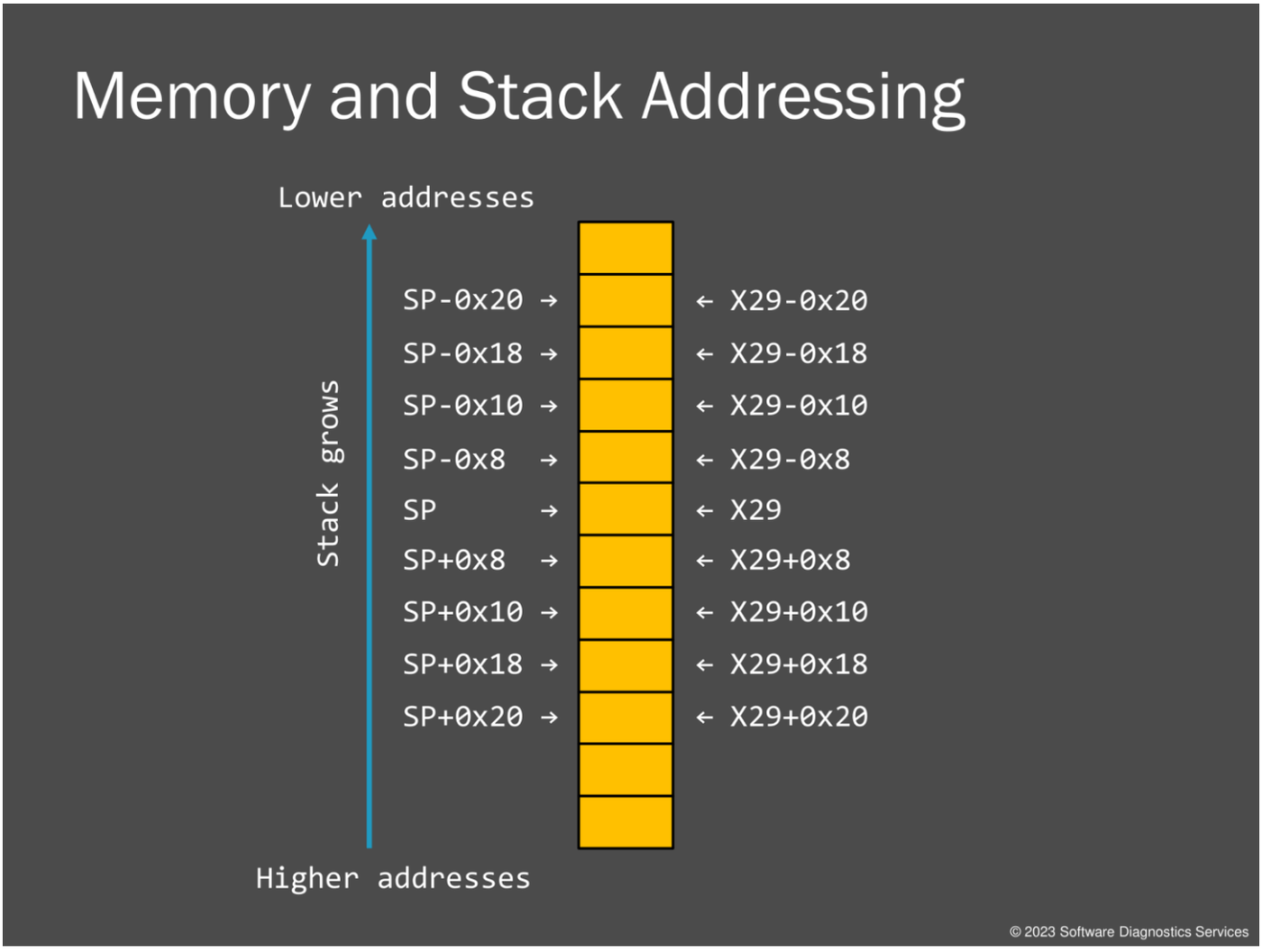
There are 31 general registers from **X0** and **X30**, with some delegated to specific tasks such as addressing stack frames (Frame Pointer, **FP**, **X29**) and return addresses, the so-called Link Register (**LR**, **X30**). When you call a function, the return address of a caller is saved in **LR**, not on the stack as in Intel/AMD x64. The return instruction in a callee will use the address in **LR** to assign it to **PC** and resume execution. But if a callee calls other functions, the current **LR** needs to be manually saved somewhere, usually on the stack. There's Stack Pointer, **SP**, of course. To get zero values, there's the so-called Zero Register, **XZR**. All **X** registers are 64-bit, and 32-bit lower parts are addressed via the **W** prefix. The References slide provides links to the ARM64 instruction set architecture. Next, we briefly look at some aspects related to our exercises.

Instructions: registers (ARM64)

- Opcode DST, SRC, SRC_2
- Examples:

```
mov   x0, #16              // X0 ← 16 (0x10)
mov   x29, sp              // X29 ← SP
add   x1, x2, #16          // X1 ← X2+16 (0x10)
mul   x1, x2, x3           // X1 ← X2*X3
blr   x8                   // X8 already contains
                           //    the address of func (&func)
                           // LR ← PC+4; PC ← &func
sub   sp, sp, #48          // SP ← SP-48 (-0x30)
                           // make a room for local variables
```


This slide shows a few examples of CPU instructions that involve operations with registers, for example, moving a value and doing arithmetic. The direction of operands is the same as in the Intel x64 disassembly flavor if you are accustomed to WinDbg on Windows. It is equivalent to an assignment. **BLR** is a call of some function whose address is in the register. **BL** means Branch and Link.



Before we look at operations with memory, let's look at a graphical representation of memory addressing. A thread stack is just any other memory region, so instead of **SP** and **X29** (**FP**), any other register can be used. Please note that the stack grows towards lower addresses, so to access the previously pushed values, you need to use positive offsets from **SP**.

Instructions: memory load (ARM64)

- Opcode DST, DST_2, [SRC, Offset]
- Opcode DST, DST_2, [SRC], Offset // Postincrement
- Examples:

```
ldr   x0, [sp]              // X0 ← value at address SP+0
ldr   x0, [x29, #-8]        // X0 ← value at address X29-0x8
ldp   x29, x30, [sp, #32]   // X29 ← value at address SP+32 (0x20)
                            // X30 ← value at address SP+40 (0x28)
ldp   x29, x30, [sp], #16   // X29 ← value at address SP+0
                            // X30 ← value at address SP+8
                            // SP ← SP+16 (0x10)
```


Constants are encoded in instructions, but if we need arbitrary values, we must get them from memory. Square brackets are used to show memory access relative to an address stored in some register. There's also an option to adjust the value of the register after load, the so-called **Postincrement**, which can be negative. As we see later, loading pairs of registers can be useful.

Instructions: memory store (ARM64)

- `Opcode SRC, SRC2, [DST, Offset]`
- `Opcode SRC, SRC2, [DST, Offset]! // Preincrement`
- Examples:

```
str   x0, [sp, #16]          // x0 → value at address SP+16 (0x10)
str   x0, [x29, #-8]         // x0 → value at address X29-8
stp   x29, x30, [sp, #32]    // x29 → value at address SP+32 (0x20)
                             // x30 → value at address SP+40 (0x28)
stp   x29, x30, [sp, #-16]! // SP ← SP-16 (-0x10)
                             // x29 → set value at address SP
                             // x30 → set value at address SP+8
```

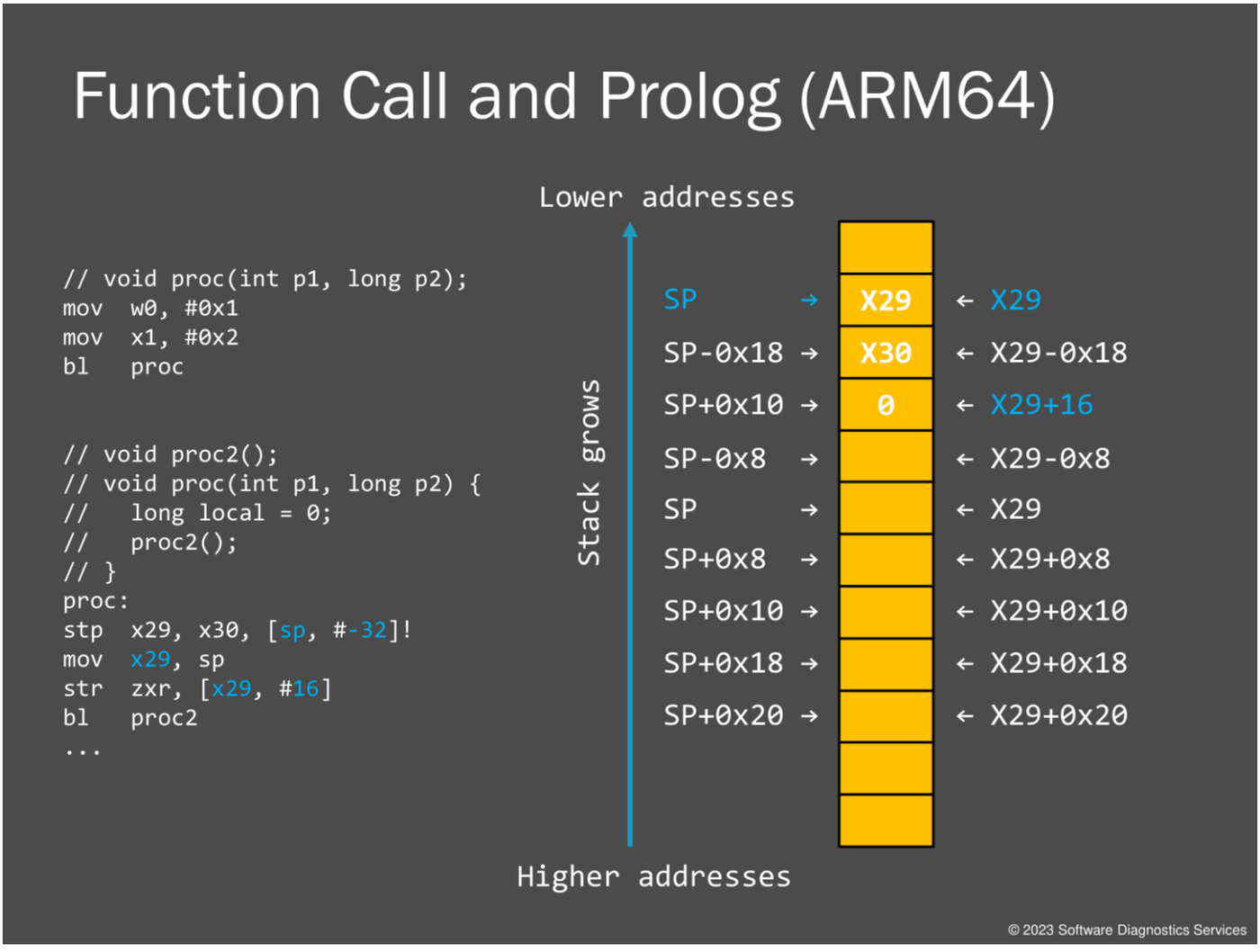

Storing operand order goes in the other direction compared to other instructions. There's a possibility to **Preincrement** the destination register before storing values.

Instructions: flow (ARM64)

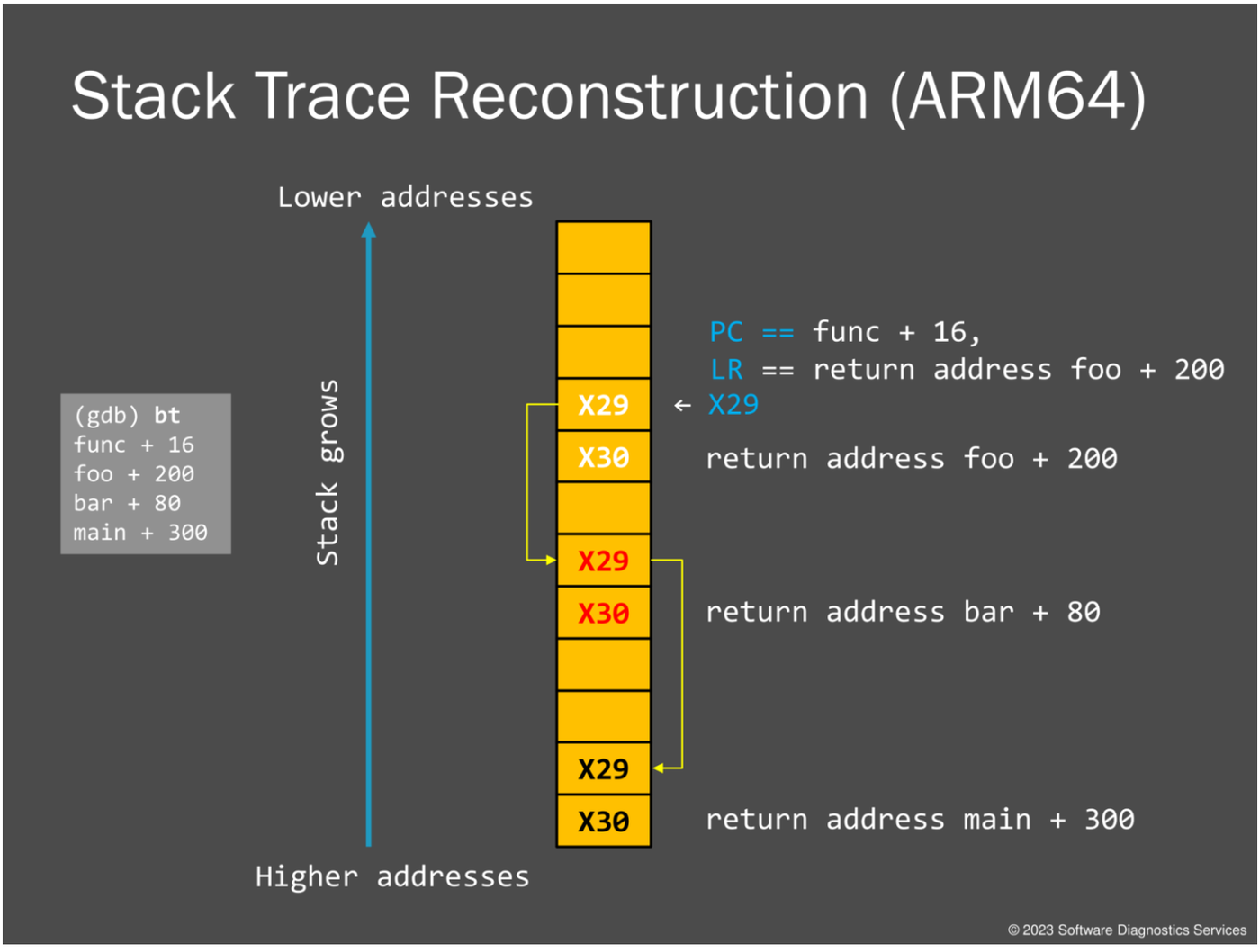
- Opcode DST, SRC
- Examples:

```
adrp  x0, 0x420000      // x0 ← 0x420000

b     0x10493fc1c       // PC ← 0x10493fc1c
                        // (goto 0x10493fc1c)
br    x17               // PC ← the value of X17

0x10493fc14:            // PC == 0x10493fc14
bl    0x10493ff74       // LR ← PC+4 (0x10493fc18)
                        // PC ← 0x10493ff74
                        // (goto 0x10493ff74)
```


Because the size of every instruction is 4 bytes (32 bits), it is only possible to encode a part of a large 4GB address range, either as a relative offset to the current **PC** or via **ADRP** instruction. Goto (an unconditional branch) is implemented via the **B** instruction. Function calls are implemented via the **BL** (Branch and Link) instruction. For conditional branches, please look at the official documentation provided in the References slide. We don't use these instructions in our exercises.

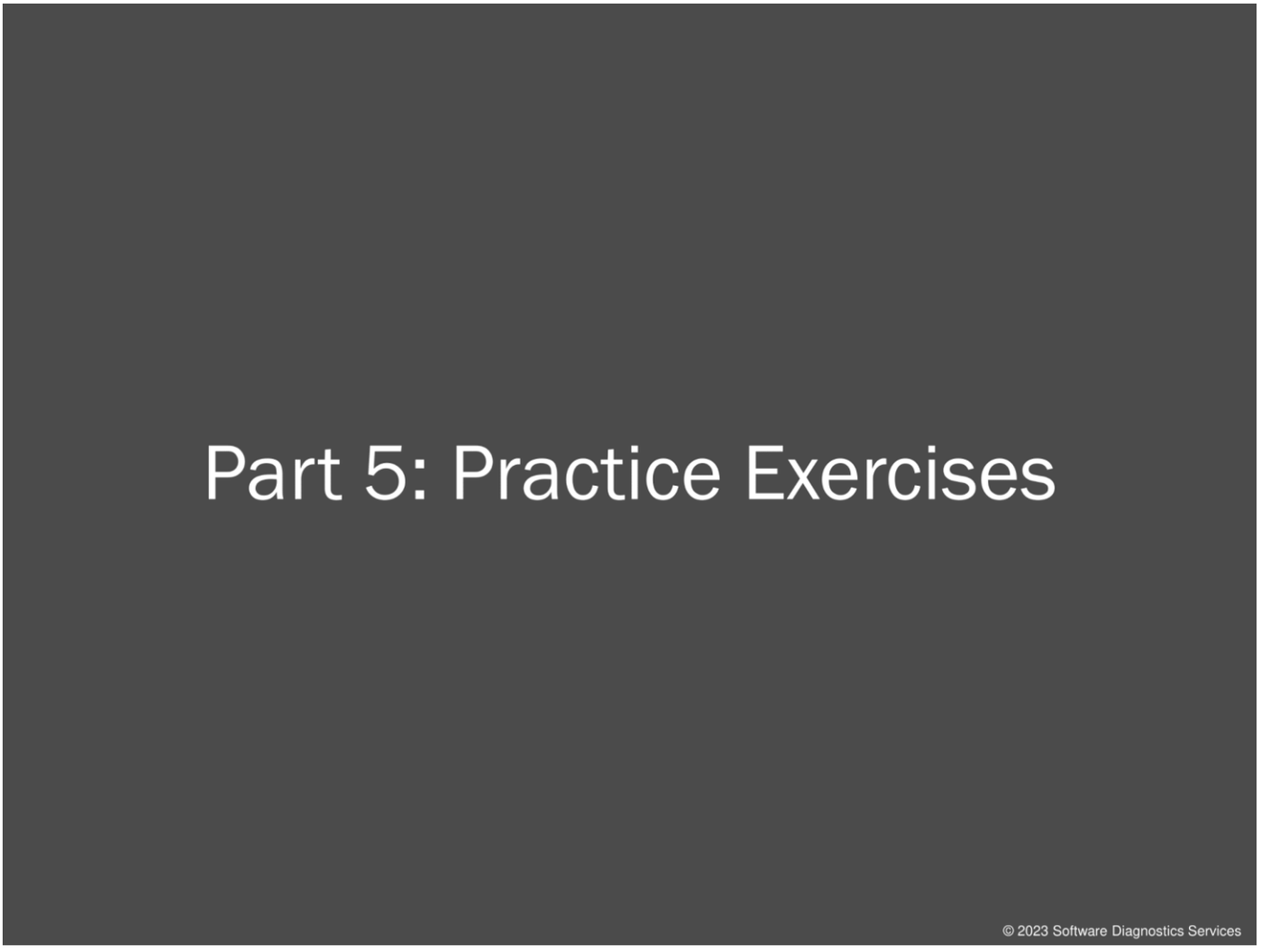


When a function is called from the caller, a callee needs to do certain operations to make room for local variables on the thread stack and save **LR** if there are further calls in the function body. There are different ways to do that, and the assembly language code on the left is one of them. I use a different color in the diagram on the right to highlight the updated **SP** and **X29** (**FP**) values before *proc2* is called. Please also note an example of zero register usage. For simplicity of illustration, I only use 64-bit values.



You may have noticed on the previous diagram that the new **X29** (**FP**) points to the **X29** of the caller, and below the previous **X29** is the return address of the caller. So, if you know either the return address in **LR** or **X29** you can reconstruct the stack trace if the compiler follows the preceding function prolog convention.

Practice Exercises



Now we come to practice. The goal is to show you important commands and how their output helps recognize patterns of abnormal software behavior.

Links

- Memory Dumps:

Included in Exercise 0

- Exercise Transcripts:

Included in this book

Exercise 0

- **Goal:** Install GDB and check if GDB loads a core dump correctly
- **Goal:** Install WinDbg Preview or Debugging Tools for Windows, or pull Docker image, and check that symbols are set up correctly
- **Patterns:** Stack Trace; Incorrect Stack Trace
- \ALCDA-Dumps\Exercise-A0-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A0-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A0-A64-WinDbg.pdf

We have three similar exercise sets: x64 Linux core dumps/GDB, ARM64 Linux core dumps/GDB, and ARM64 Linux core dumps/WinDbg.

Exercise 0 (x64, GDB)

Goal: Install GDB and check if GDB loads a core dump correctly.

Patterns: Stack Trace; Incorrect Stack Trace.

1. Download core dump files if you haven't done that already and unpack the archives:

https://www.patterndiagnostics.com/Training/ALCDA/ALCDA-V2-Dumps.tar.gz

2. Download and install the latest version of GDB. For WSL2 Debian, we used the following commands:

```
$ sudo apt install build-essential
$ sudo apt install gdb
```

On our RHEL-type system, we installed the tools and GDB via:

```
$ sudo yum group install "Development Tools"
$ sudo yum install gdb
```

3. Verify that GDB is accessible and then exit it (**q** command):

```
$ gdb
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word".

(gdb) q
$
```

4. Load *core.App0* dump file and *App0* executable from the x64/App0 directory:

```
$ cd ALCDA2/x64/App0

~/ALCDA2/x64/App0$ gdb -c core.App0 -se App0

GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
```

```
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App0...(no debugging symbols found)...done.
[New LWP 4561]
Core was generated by `./App0'.
Program terminated with signal SIGABRT, Aborted.
#0  0x00000000004075cb in raise ()
```

5. Verify that the stack trace (back trace) is shown correctly with symbols:

```
(gdb) bt
#0  0x00000000004075cb in raise ()
#1  0x0000000000401205 in abort ()
#2  0x0000000000401b56 in bar ()
#3  0x0000000000401b64 in foo ()
#4  0x0000000000401b80 in main ()
```

6. We exit GDB.

```
(gdb) q

~/ALCDA2/x64/App0$
```

Exercise 0 (A64, GDB)

Goal: Install GDB and check if GDB loads a core dump correctly.

Patterns: Stack Trace; Incorrect Stack Trace.

1. Download core dump files if you haven't done that already and unpack the archives:

https://www.patterndiagnostics.com/Training/ALCDA/ALCDA-V2-Dumps.zip
https://www.patterndiagnostics.com/Training/ALCDA/ALCDA-V3-Dumps.tar.gz

2. Download and install the latest version of GDB. For Ubuntu, we used the following commands:

```
$ sudo apt install build-essential
$ sudo apt install gdb
```

3. Verify that GDB is accessible and then exit it (**q** command):

```
$ gdb
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word".
(gdb)

(gdb) q
$
```

4. Load *core.31918* dump file and *App0* executable from the A64/App0 directory:

```
$ cd ALCDA2/A64/App0

~/ALCDA2/A64/App0$ gdb -c core.31918 -se App0

GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
```

```
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App0...
(No debugging symbols found in App0)

warning: Can't open file /home/opc/ALCDA2/App0/App0 during file-backed mapping note processing
[New LWP 31918]
Core was generated by `./App0'.
Program terminated with signal SIGABRT, Aborted.
#0  0x0000000000415000 in raise ()
```

5. Verify that the stack trace (back trace) is shown correctly with symbols:

```
(gdb) bt
#0  0x0000000000415000 in raise ()
#1  0x0000000000402808 in abort ()
#2  0x0000000000401d24 in bar ()
#3  0x0000000000401d30 in foo ()
#4  0x0000000000401d4c in main ()
```

6. We exit GDB.

```
(gdb) q

~/ALCDA2/A64/App0$
```

Exercise 0 (A64, WinDbg Preview, WinDbg, Docker)

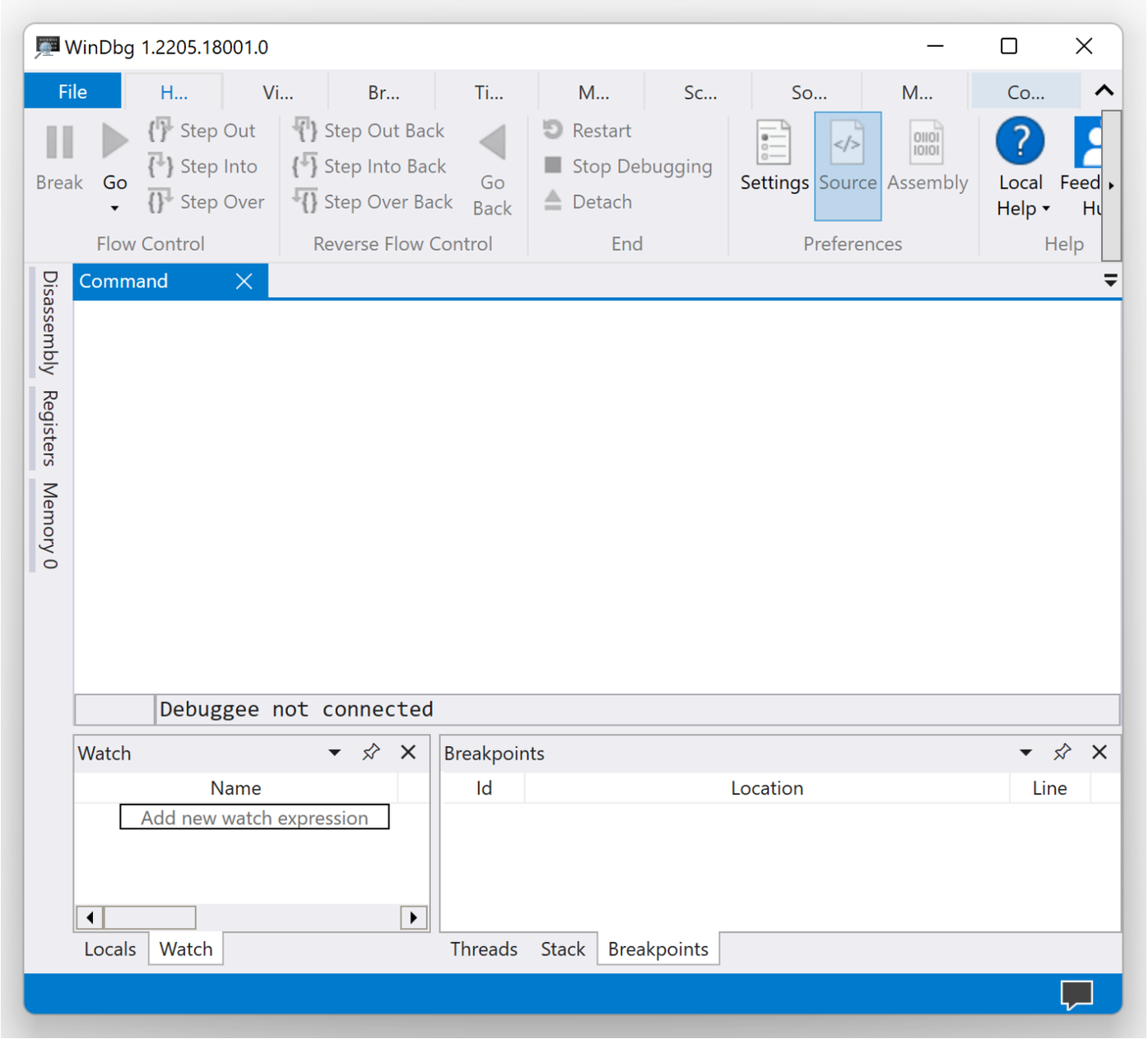
Goal: Install WinDbg Preview or Debugging Tools for Windows, or pull Docker image, and check that symbols are set up correctly.

Patterns: Stack Trace; Incorrect Stack Trace.

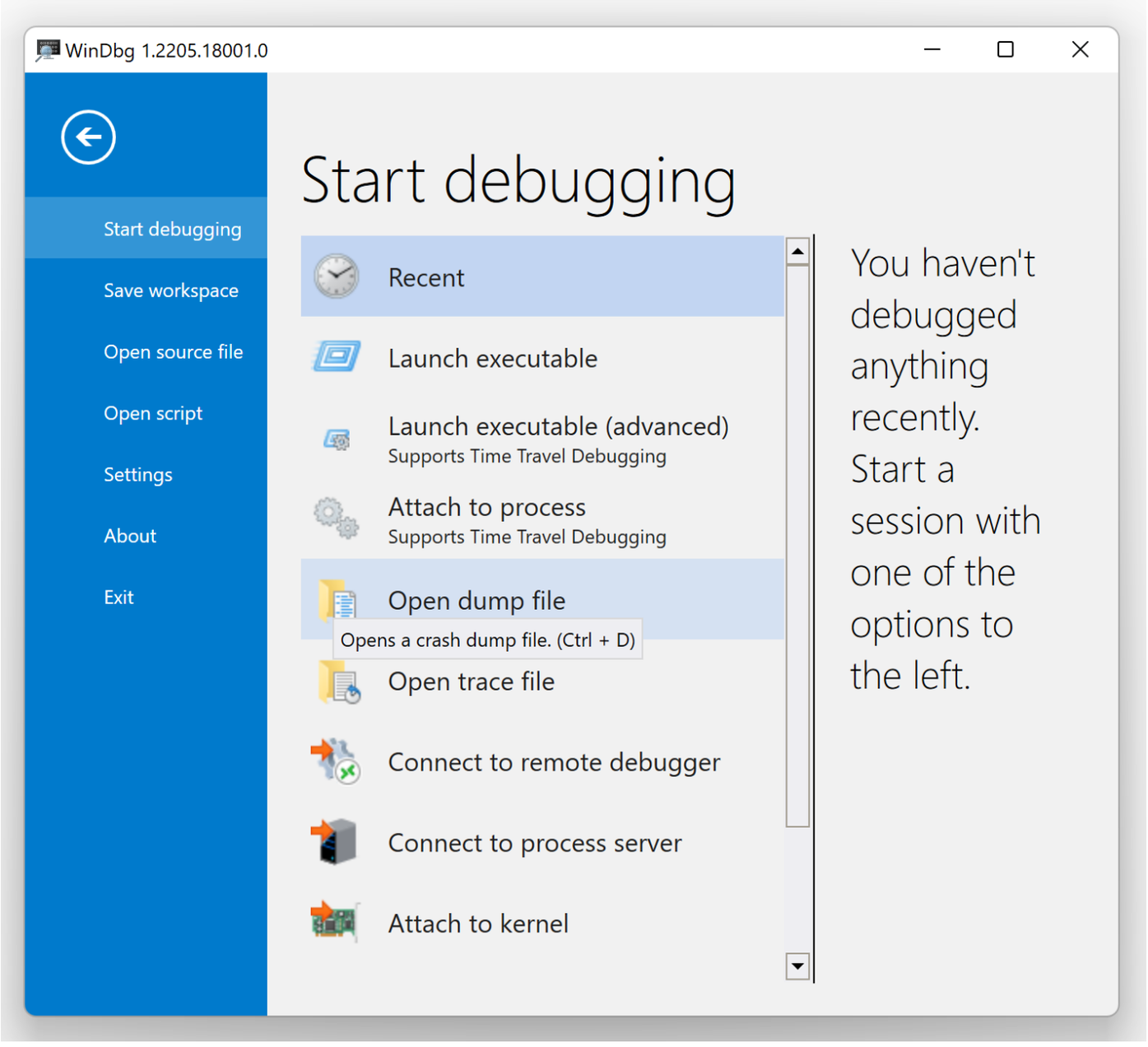
1. Download memory dump files if you haven't done that already and unpack the archives:

https://www.patterndiagnostics.com/Training/ALCDA/ALCDA-V2-Dumps.zip

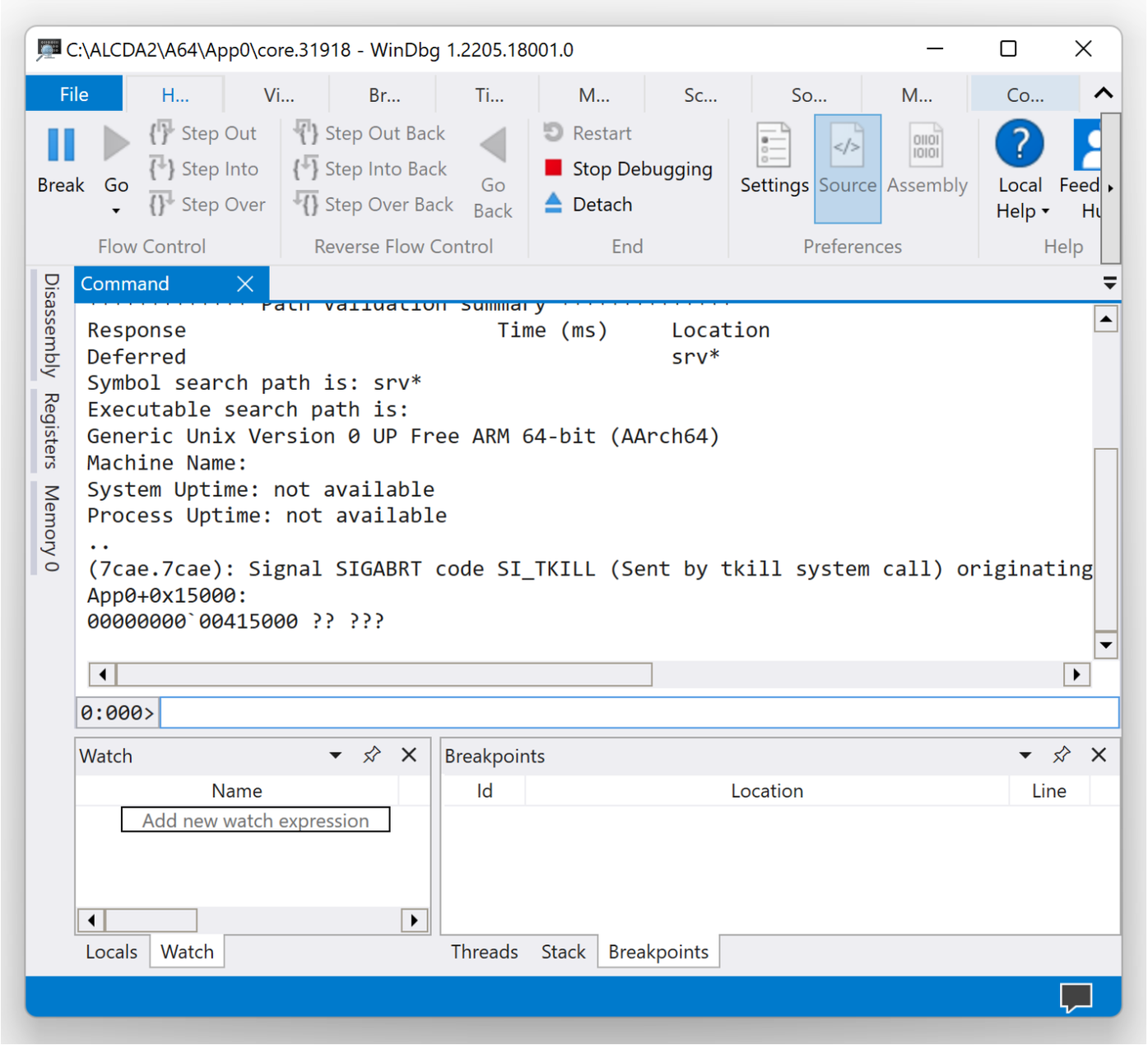
2. Install WinDbg Preview from Microsoft Store. Run WinDbg Preview app.



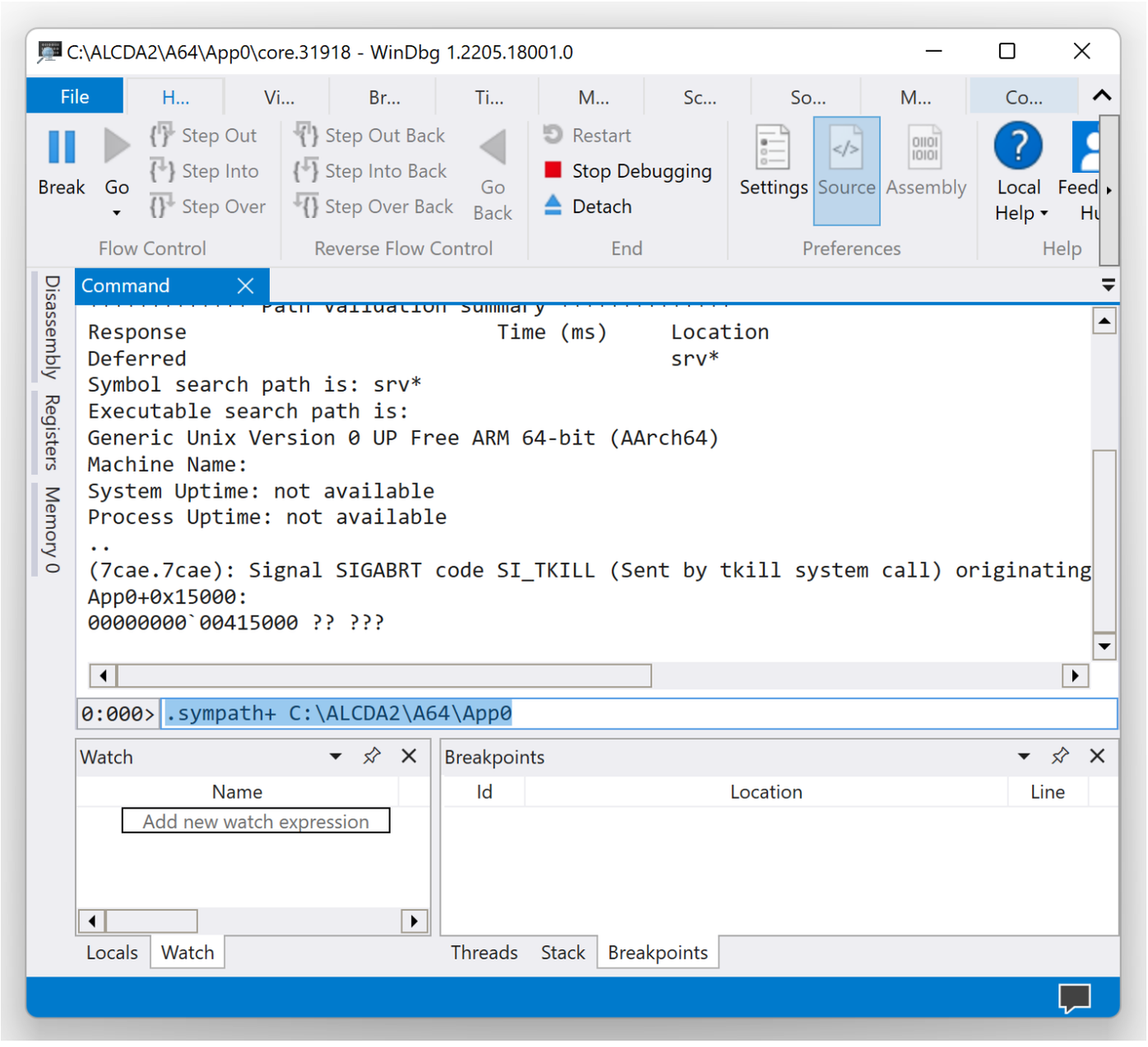
3. Open \ALCDA2\A64\App0\core.31918:



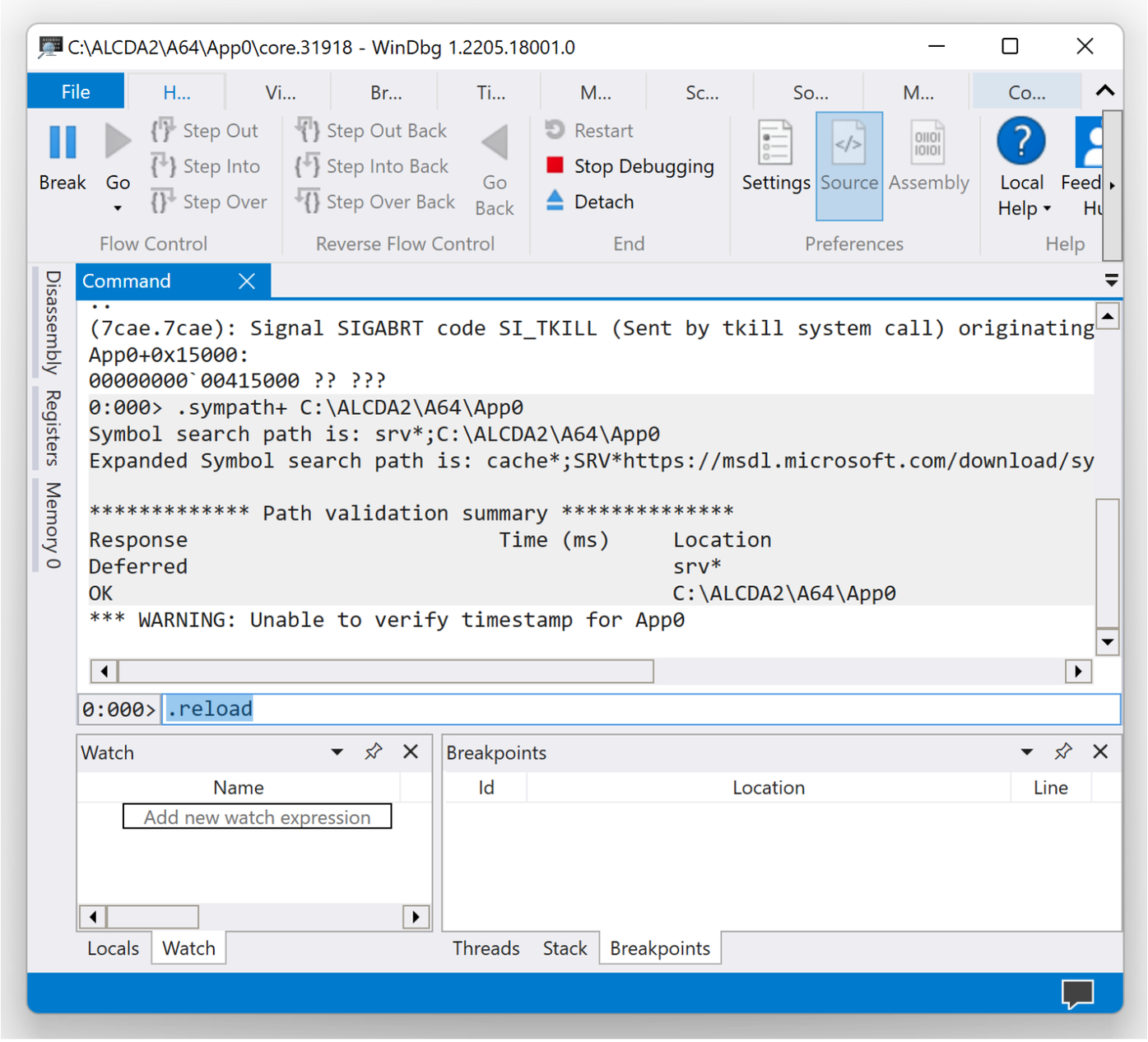
4. We get the dump file loaded:



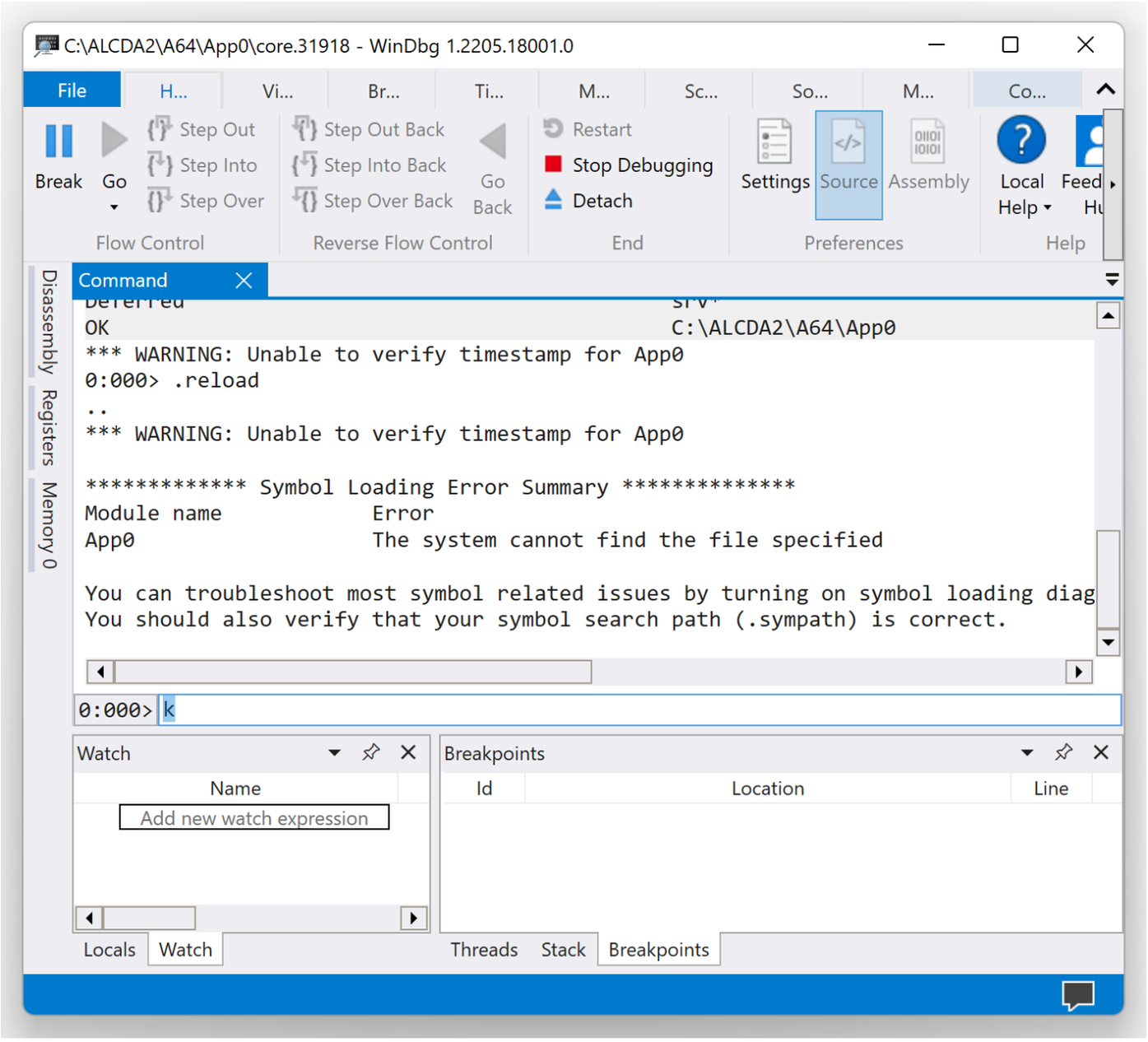
5. Type **.sympath+** <path> command to set symbol path:

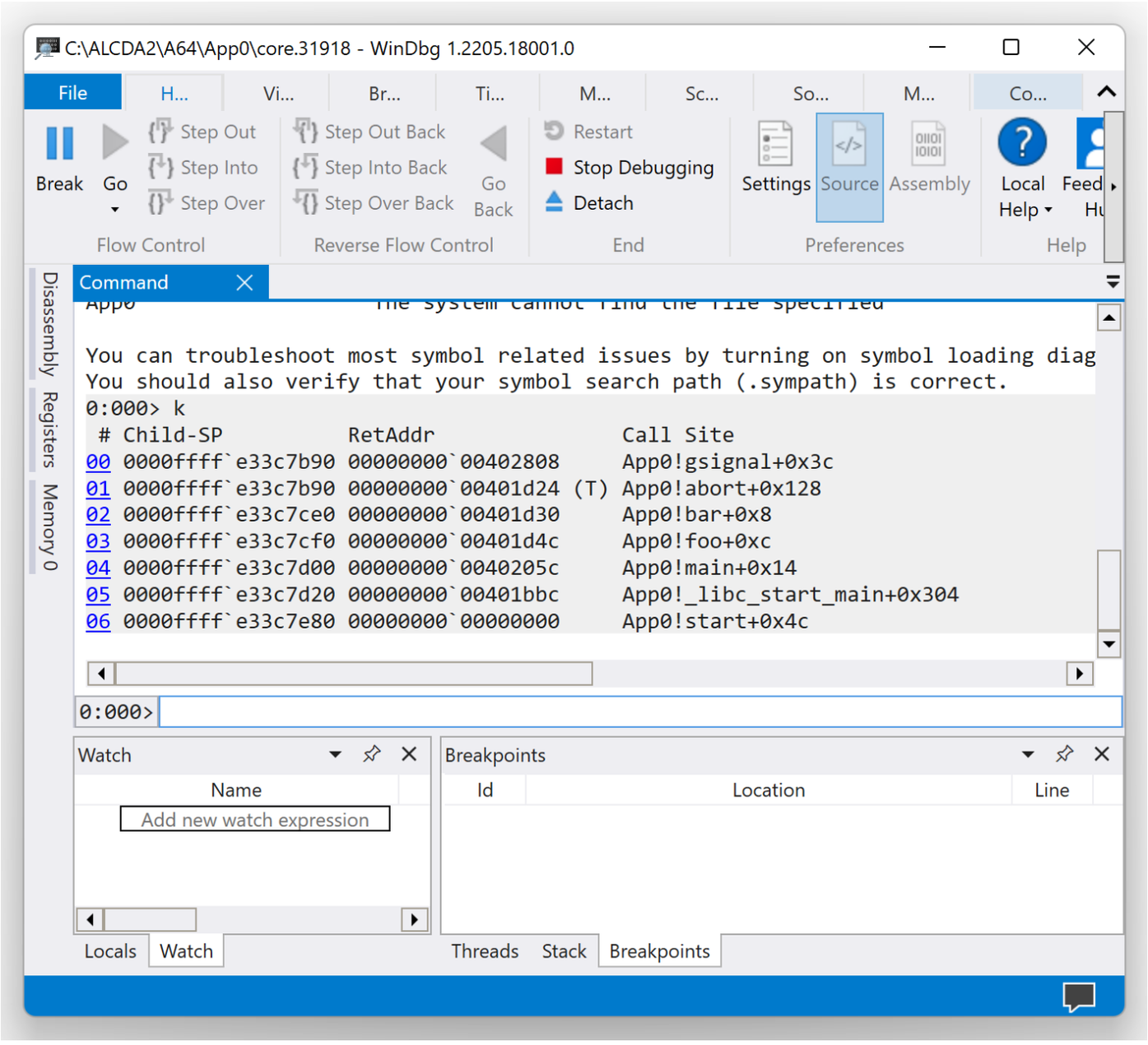


6. Type **.reload** command to reload symbols:



7. Type **k** command to verify the correctness of the stack trace:





8. The output of command should be this:

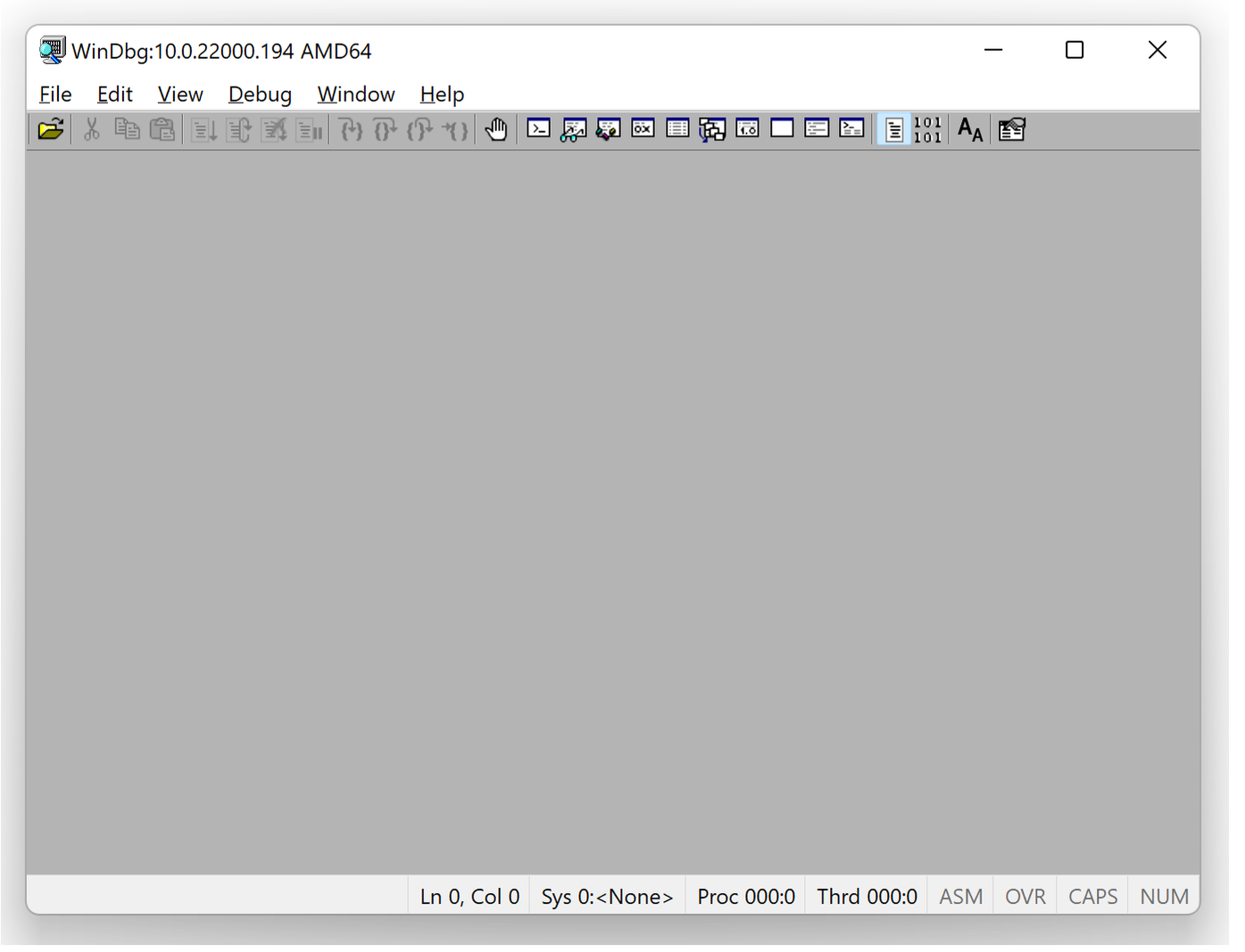
```
0:000> k
 # Child-SP          RetAddr               Call Site
00 0000ffff`e33c7b90 00000000`00402808     App0!gsignal+0x3c
01 0000ffff`e33c7b90 00000000`00401d24 (T) App0!abort+0x128
02 0000ffff`e33c7ce0 00000000`00401d30     App0!bar+0x8
03 0000ffff`e33c7cf0 00000000`00401d4c     App0!foo+0xc
04 0000ffff`e33c7d00 00000000`0040205c     App0!main+0x14
05 0000ffff`e33c7d20 00000000`00401bbc     App0!_libc_start_main+0x304
06 0000ffff`e33c7e80 00000000`00000000     App0!start+0x4c
```

If it has this form below with large offsets, then your symbol files were not set up correctly - **Incorrect Stack Trace** pattern:

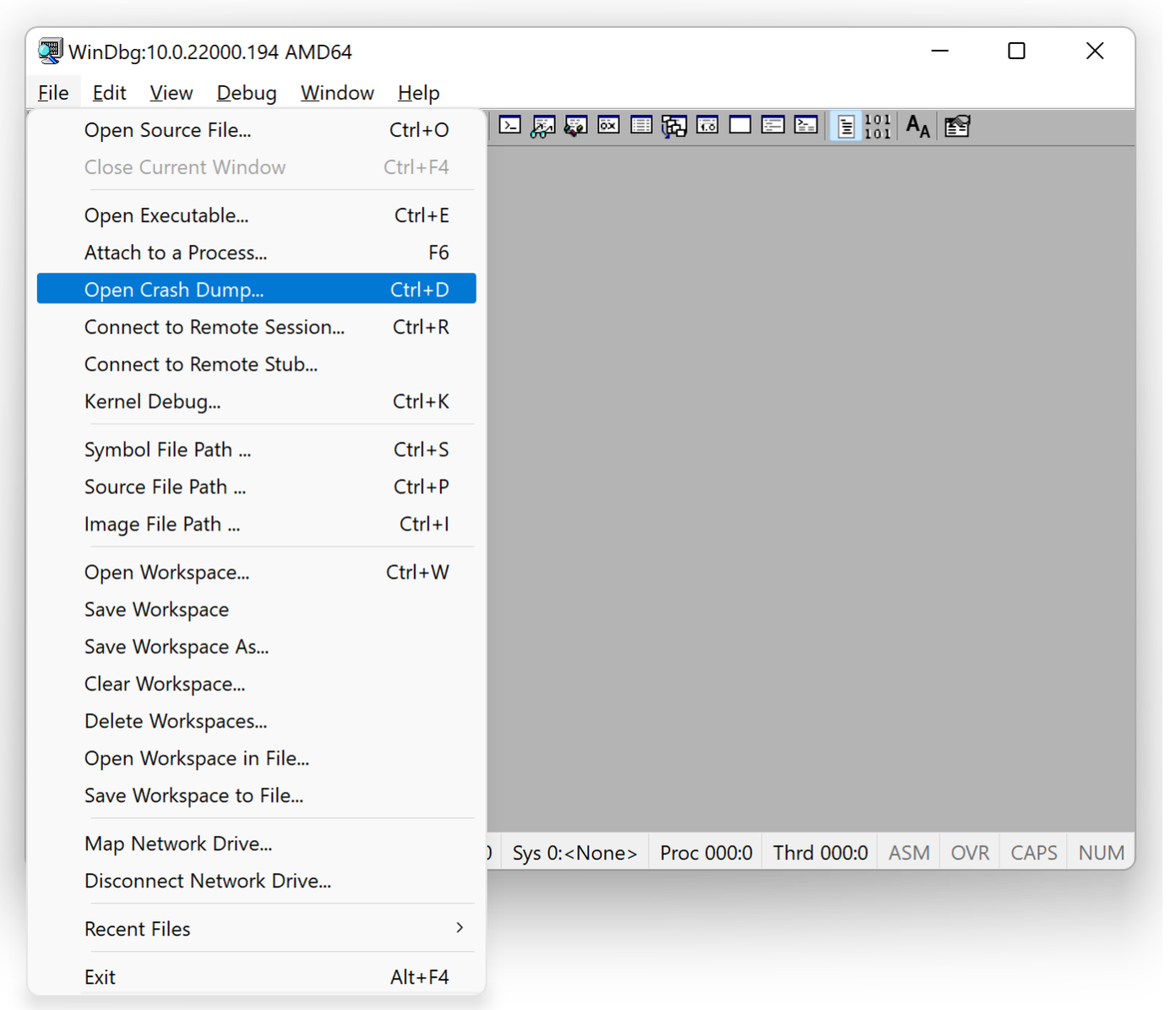
```
0:000> k
# Child-SP          RetAddr               Call Site
00 0000ffff`e33c7b90 00000000`00401d24     App0+0x15000
01 0000ffff`e33c7ba0 00000000`00401d30     App0+0x1d24
02 0000ffff`e33c7cf0 00000000`00401d4c     App0+0x1d30
03 0000ffff`e33c7d00 00000000`0040205c     App0+0x1d4c
04 0000ffff`e33c7d10 00000000`00000000     App0+0x205c
```

9. [Optional] Download and install the recommended version of Debugging Tools for Windows (See windbg.org for quick links, WinDbg Quick Links \ Download Debugging Tools for Windows). For this part, we use WinDbg 10.0.22000.194 from Windows 11 SDK version 10.0.22000. When installing it, choose Debugging Tools for Windows.

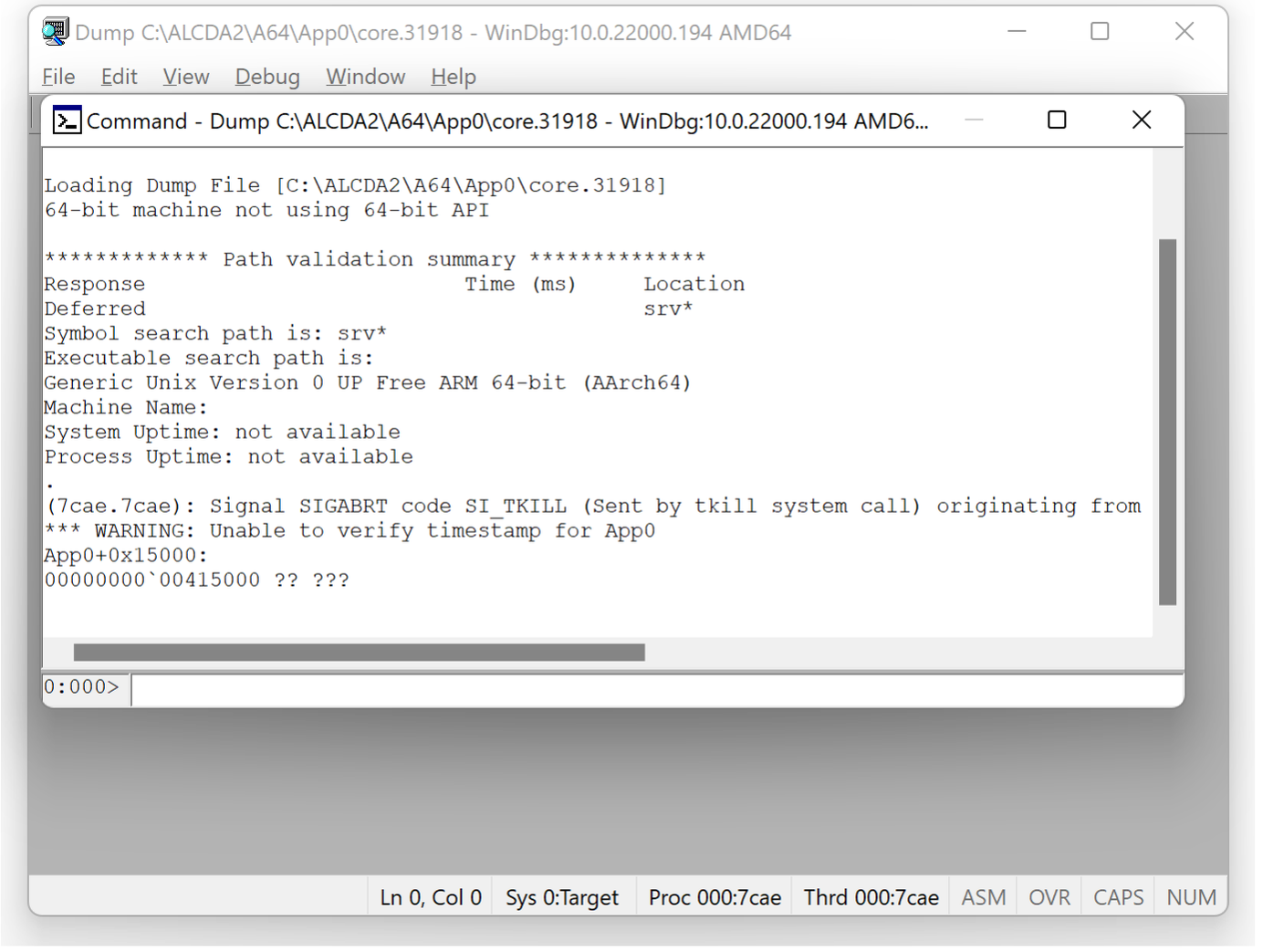
10. Launch WinDbg from Windows Kits \ WinDbg (X64) or Windows Kits \ WinDbg (X86). For uniformity, we use the X64 version of WinDbg throughout the exercises.



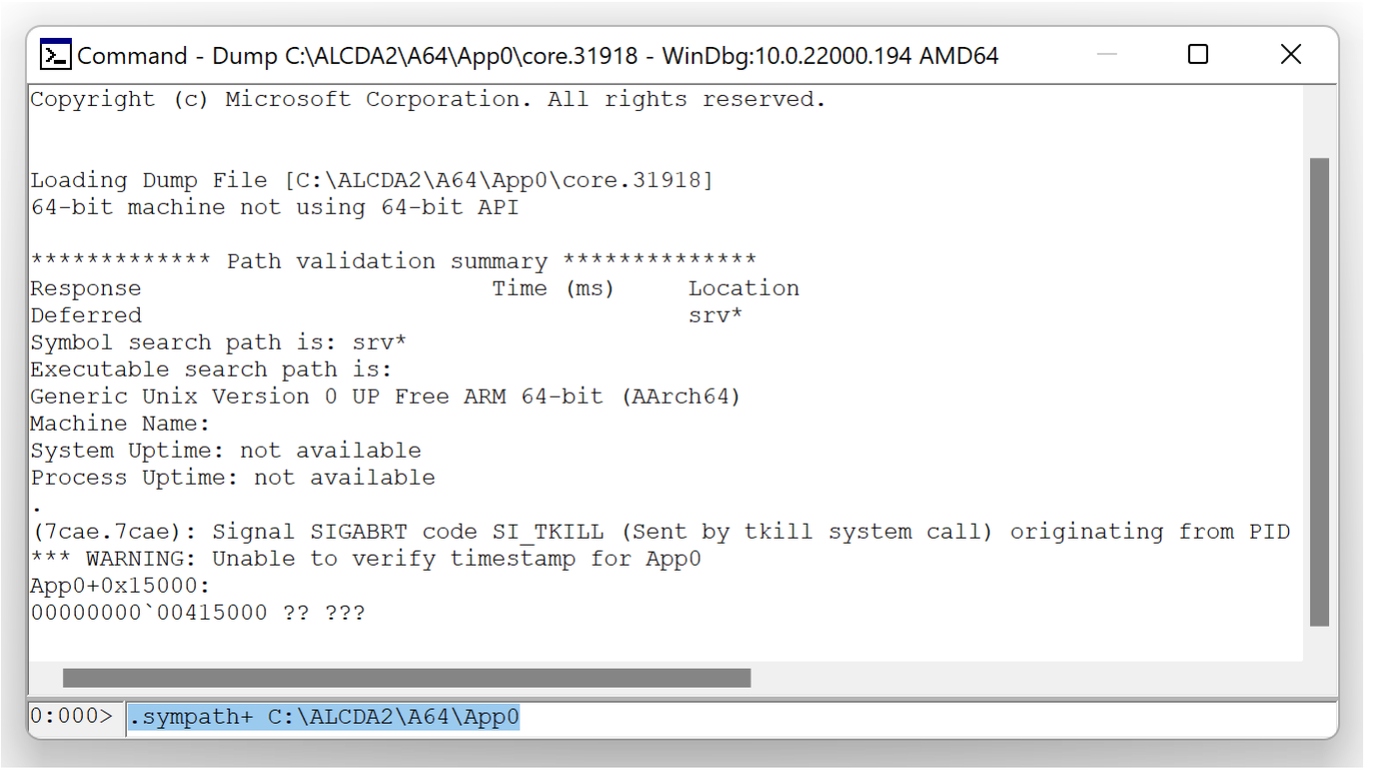
11. Open \ALCDA2\A64\App0\core.31918:



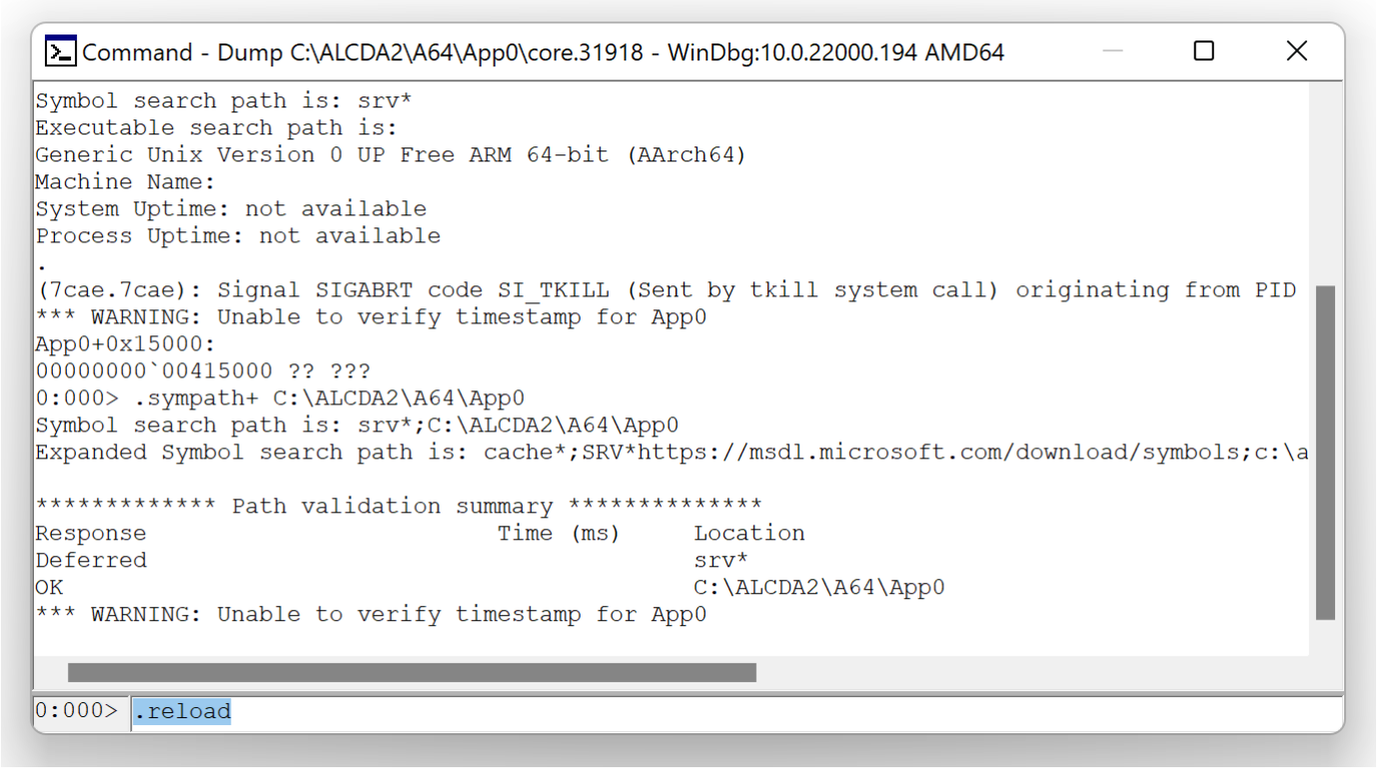
12. We get the dump file loaded:



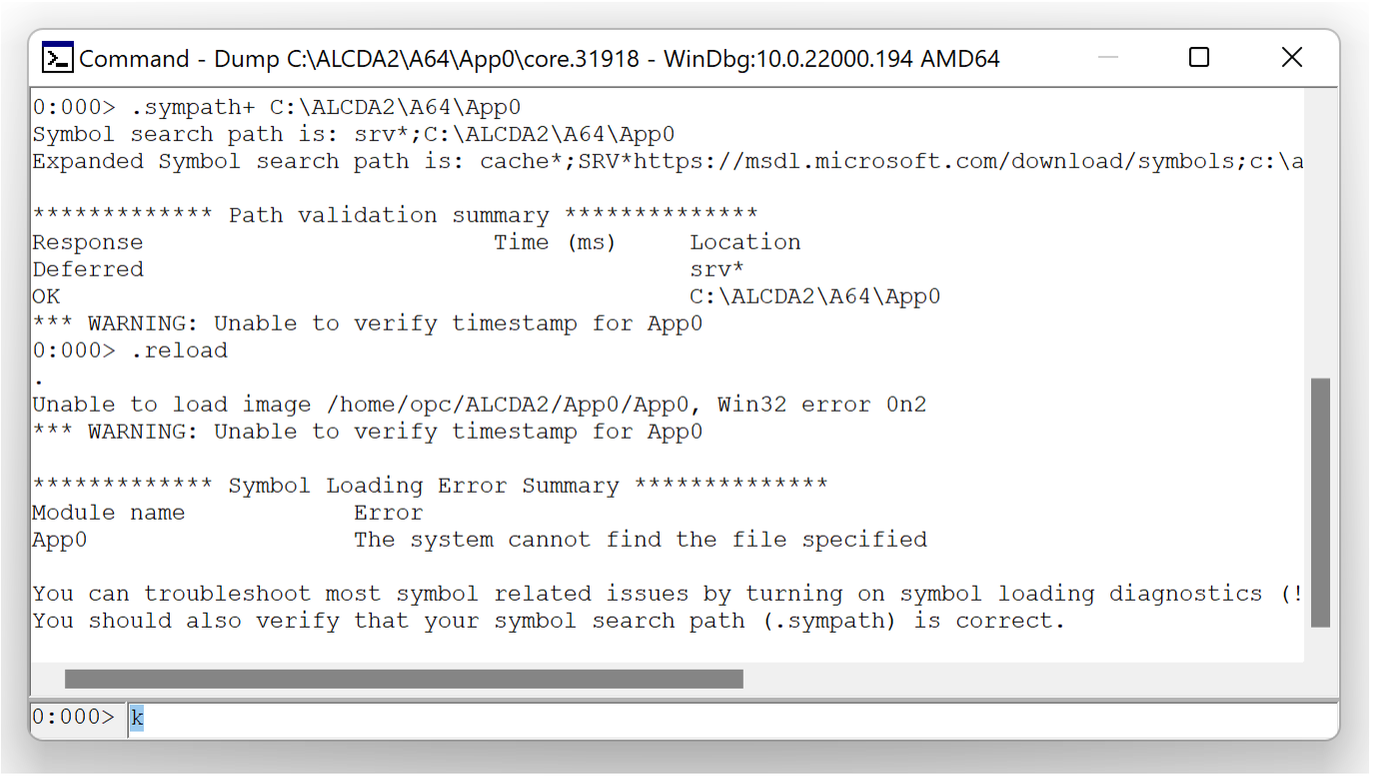
13. Type **.sympath+** <path> command to set symbol path:

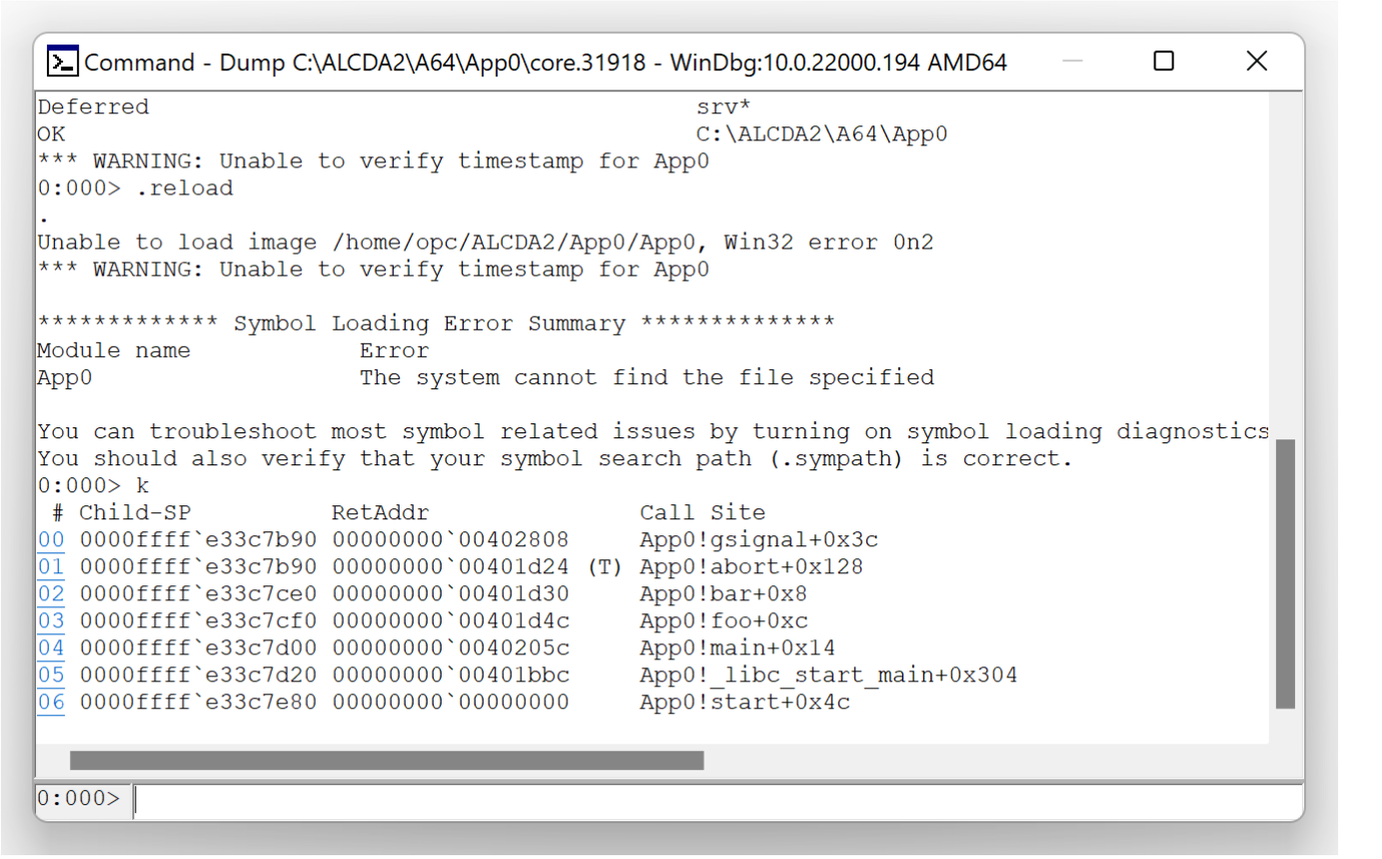


14. Type **.reload** command to reload symbols:



15. Type **k** command to verify the correctness of the stack trace:

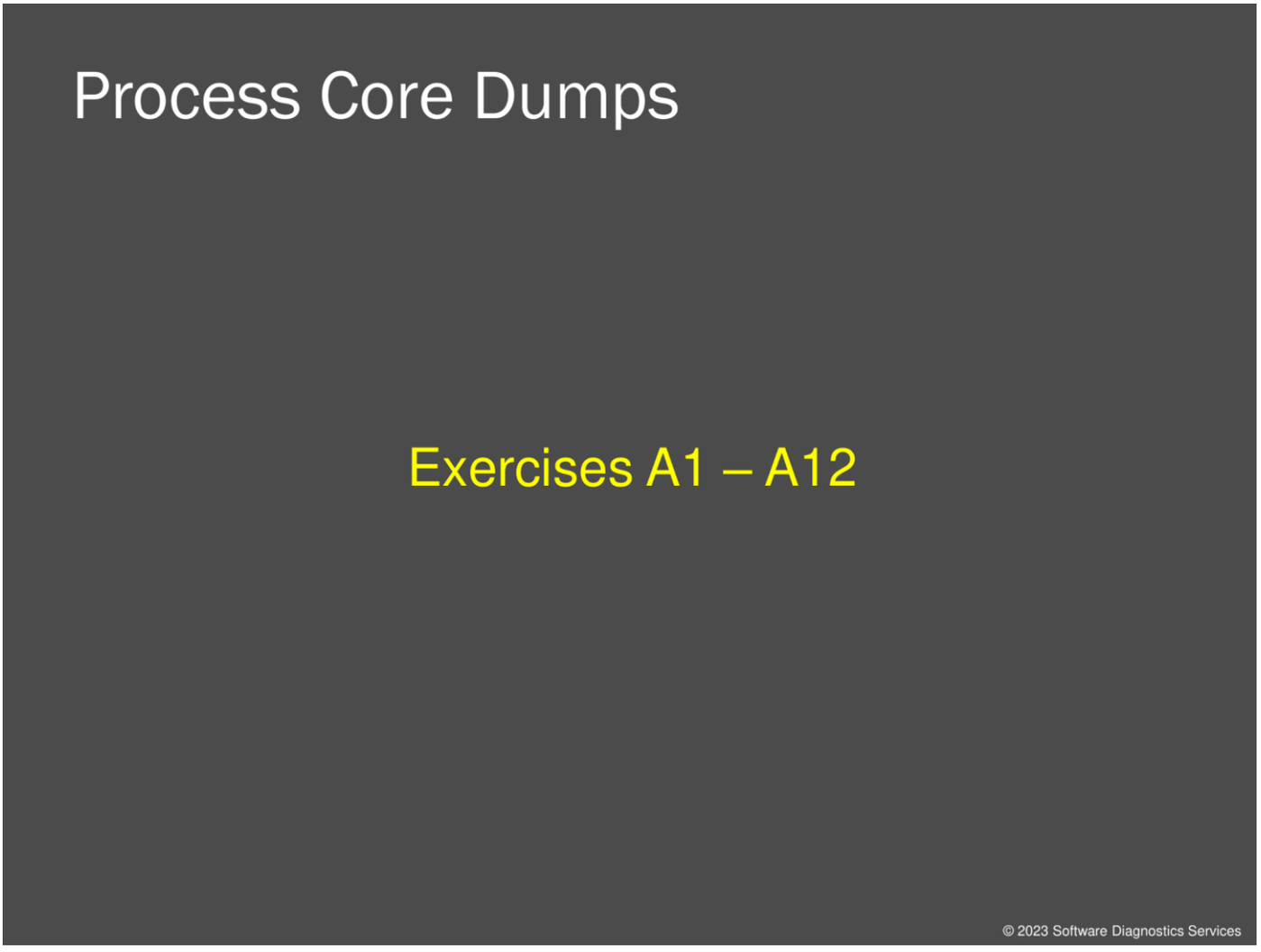




16. [Optional] If you prefer using Docker image with WinDbg and symbol files included, follow these steps below.

```
c:\ALCDA2>docker pull patterndiagnostics/windbg:10.0.22000.194-wsl
10.0.22000.194-wsl: Pulling from patterndiagnostics/windbg
8f616e6e9eec: Pull complete
b03bbc71f925: Pull complete
4c7d8699f10d: Pull complete
2c76fbacfcb8: Pull complete
0692b7e8acd8: Pull complete
2ce4617bc74f: Pull complete
Digest: sha256:ad644af7ff34dac06dd89f6063b047a82865d0027745bfc85210ea62e1d2e365
Status: Downloaded newer image for patterndiagnostics/windbg:10.0.22000.194-wsl
docker.io/patterndiagnostics/windbg:10.0.22000.194-wsl

c:\ALCDA2>docker run -it -v C:\ALCDA2:C:\ALCDA2 patterndiagnostics/windbg:10.0.22000.194-wsl
Microsoft Windows [Version 10.0.20348.288]
(c) Microsoft Corporation. All rights reserved.

C:\WinDbg>windbg.bat C:\ALCDA2\A64\App0\core.31918

Microsoft (R) Windows Debugger Version 10.0.22000.194 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.

Loading Dump File [C:\ALCDA2\A64\App0\core.31918]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
OK                                             .
Symbol search path is: .
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
.
(7cae.7cae): Signal SIGABRT code SI_TKILL (Sent by tkill system call) originating from PID 7caeUnable to
load image /home/opc/ALCDA2/App0/App0, Win32 error 0n2
*** WARNING: Unable to verify timestamp for App0
App0+0x15000:
00000000`00415000 ?? ???

0:000> .sympath+ C:\ALCDA2\A64\App0
Symbol search path is: .;C:\ALCDA2\A64\App0
Expanded Symbol search path is: .;c:\alcda2\a64\app0

************* Path validation summary **************
Response                         Time (ms)     Location
OK                                             .
OK                                             C:\ALCDA2\A64\App0

0:000> .reload
.
Unable to load image /home/opc/ALCDA2/App0/App0, Win32 error 0n2
*** WARNING: Unable to verify timestamp for App0

************* Symbol Loading Error Summary **************
Module name            Error
App0                   The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym noisy) and
repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

```
0:000> k
Child-SP          RetAddr               Call Site
0000ffff`e33c7b90 00000000`00402808     App0!gsignal+0x3c
0000ffff`e33c7b90 00000000`00401d24 (T) App0!abort+0x128
0000ffff`e33c7ce0 00000000`00401d30     App0!bar+0x8
0000ffff`e33c7cf0 00000000`00401d4c     App0!foo+0xc
0000ffff`e33c7d00 00000000`0040205c     App0!main+0x14
0000ffff`e33c7d20 00000000`00401bbc     App0!_libc_start_main+0x304
0000ffff`e33c7e80 00000000`00000000     App0!start+0x4c

0:000> q
quit:
NatVis script unloaded from 'C:\Program Files\Windows Kits\10\Debuggers\x64\Visualizers\gstl.natvis'

C:\WinDbg>exit

c:\ALCDA2>
```



All exercises were modeled on real-life examples using specially constructed applications. We learn how to recognize and use almost 40 analysis patterns.

Exercise A1

- **Goal:** Learn how to list stack traces, disassemble functions, check their correctness, dump data, get environment
- **Patterns:** Manual Dump (Process); Stack Trace; Stack Trace Collection; Annotated Disassembly; Paratext; Not My Version; Environment Hint
- \ALCDA-Dumps\Exercise-A1-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A1-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A1-A64-WinDbg.pdf

Exercise A1 (x64, GDB)

Goal: Learn how to list stack traces, disassemble functions, check their correctness, dump data, get environment.

Patterns: Manual Dump (Process); Stack Trace; Stack Trace Collection; Annotated Disassembly; Paratext; Not My Version; Environment Hint.

1. Load a core dump *App1.core.253* and *App1* executable from the x64/App1 directory:

```
~/ALCDA2/x64/App1$ gdb -c App1.core.253 -se App1
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App1...done.
[New LWP 253]
[New LWP 254]
[New LWP 255]
[New LWP 256]
[New LWP 257]
[New LWP 258]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App1'.
#0  0x0000000000441a10 in nanosleep ()
[Current thread is 1 (Thread 0x21b3880 (LWP 253))]
```

2. Set logging to a file in case of lengthy output from some commands:

```
(gdb) set logging on App1.log
Copying output to App1.log.
```

3. List all threads:

```
(gdb) info threads
  Id   Target Id                      Frame
* 1    Thread 0x21b3880 (LWP 253)     0x0000000000441a10 in nanosleep ()
  2    Thread 0x7f0fc16fb700 (LWP 254) 0x0000000000441a10 in nanosleep ()
  3    Thread 0x7f0fc0efa700 (LWP 255) 0x0000000000441a10 in nanosleep ()
  4    Thread 0x7f0fc06f9700 (LWP 256) 0x0000000000441a10 in nanosleep ()
  5    Thread 0x7f0fbfef8700 (LWP 257) 0x0000000000441a10 in nanosleep ()
  6    Thread 0x7f0fbf6f7700 (LWP 258) 0x0000000000441a10 in nanosleep ()
```

4. Get the current thread stack trace:

```
(gdb) bt
#0  0x0000000000441a10 in nanosleep ()
#1  0x000000000044199a in sleep ()
#2  0x0000000000401d92 in main () at pthread_create.c:688
```

5. Get all thread stack traces:

```
(gdb) thread apply all bt

Thread 6 (Thread 0x7f0fbf6f7700 (LWP 258)):
#0  0x0000000000441a10 in nanosleep ()
#1  0x000000000044199a in sleep ()
#2  0x0000000000401cb7 in bar_five ()
#3  0x0000000000401cc8 in foo_five ()
#4  0x0000000000401ce1 in thread_five ()
#5  0x00000000004030d3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000044426f in clone ()

Thread 5 (Thread 0x7f0fbfef8700 (LWP 257)):
#0  0x0000000000441a10 in nanosleep ()
#1  0x000000000044199a in sleep ()
#2  0x0000000000401c78 in bar_four () at pthread_create.c:688
#3  0x0000000000401c89 in foo_four () at pthread_create.c:688
#4  0x0000000000401ca2 in thread_four () at pthread_create.c:688
#5  0x00000000004030d3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000044426f in clone ()

Thread 4 (Thread 0x7f0fc06f9700 (LWP 256)):
#0  0x0000000000441a10 in nanosleep ()
#1  0x000000000044199a in sleep ()
#2  0x0000000000401c39 in bar_three () at pthread_create.c:688
#3  0x0000000000401c4a in foo_three () at pthread_create.c:688
#4  0x0000000000401c63 in thread_three () at pthread_create.c:688
#5  0x00000000004030d3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000044426f in clone ()

Thread 3 (Thread 0x7f0fc0efa700 (LWP 255)):
#0  0x0000000000441a10 in nanosleep ()
#1  0x000000000044199a in sleep ()
#2  0x0000000000401bfa in bar_two () at pthread_create.c:688
#3  0x0000000000401c0b in foo_two () at pthread_create.c:688
#4  0x0000000000401c24 in thread_two () at pthread_create.c:688
#5  0x00000000004030d3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000044426f in clone ()

Thread 2 (Thread 0x7f0fc16fb700 (LWP 254)):
#0  0x0000000000441a10 in nanosleep ()
#1  0x000000000044199a in sleep ()
#2  0x0000000000401bbb in bar_one () at pthread_create.c:688
#3  0x0000000000401bcc in foo_one () at pthread_create.c:688
#4  0x0000000000401be5 in thread_one () at pthread_create.c:688
#5  0x00000000004030d3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000044426f in clone ()

Thread 1 (Thread 0x21b3880 (LWP 253)):
#0  0x0000000000441a10 in nanosleep ()
#1  0x000000000044199a in sleep ()
#2  0x0000000000401d92 in main () at pthread_create.c:688
```

6. Switch to thread #2 and get its stack trace:

```
(gdb) thread 2
[Switching to thread 2 (Thread 0x7f0fc16fb700 (LWP 254))]
#0  0x0000000000441a10 in nanosleep ()

(gdb) bt
#0  0x0000000000441a10 in nanosleep ()
#1  0x000000000044199a in sleep ()
#2  0x0000000000401bbb in bar_one () at pthread_create.c:688
#3  0x0000000000401bcc in foo_one () at pthread_create.c:688
#4  0x0000000000401be5 in thread_one () at pthread_create.c:688
#5  0x00000000004030d3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000044426f in clone ()

(gdb) info threads
  Id   Target Id                      Frame
  1    Thread 0x21b3880 (LWP 253)     0x0000000000441a10 in nanosleep ()
* 2    Thread 0x7f0fc16fb700 (LWP 254) 0x0000000000441a10 in nanosleep ()
  3    Thread 0x7f0fc0efa700 (LWP 255) 0x0000000000441a10 in nanosleep ()
  4    Thread 0x7f0fc06f9700 (LWP 256) 0x0000000000441a10 in nanosleep ()
  5    Thread 0x7f0fbfef8700 (LWP 257) 0x0000000000441a10 in nanosleep ()
  6    Thread 0x7f0fbf6f7700 (LWP 258) 0x0000000000441a10 in nanosleep ()
```

7. Check that *bar_one* called the *sleep* function by comparing the return address on the call stack from the disassembly output:

```
(gdb) disassemble bar_one
Dump of assembler code for function bar_one:
   0x0000000000401bad <+0>:     push   %rbp
   0x0000000000401bae <+1>:     mov    %rsp,%rbp
   0x0000000000401bb1 <+4>:     mov    $0xffffffff,%edi
   0x0000000000401bb6 <+9>:     callq  0x441960 <sleep>
   0x0000000000401bbb <+14>:    nop
   0x0000000000401bbc <+15>:    pop    %rbp
   0x0000000000401bbd <+16>:    retq
End of assembler dump.
```

We see that the address in the stack trace for the *bar_one* function is the address to return to after calling the *sleep* function.

8. Compare with Intel disassembly flavor:

```
(gdb) set disassembly-flavor intel

(gdb) disassemble bar_two
Dump of assembler code for function bar_one:
   0x0000000000401bad <+0>:     push   rbp
   0x0000000000401bae <+1>:     mov    rbp,rsp
   0x0000000000401bb1 <+4>:     mov    edi,0xffffffff
   0x0000000000401bb6 <+9>:     call   0x441960 <sleep>
   0x0000000000401bbb <+14>:    nop
   0x0000000000401bbc <+15>:    pop    rbp
   0x0000000000401bbd <+16>:    ret
End of assembler dump.

(gdb) set disassembly-flavor att
```

9. Get *App1* data section from the output of pmap (*App1.pmap.253*):

```
(gdb) ^Z
[2]+  Stopped                 gdb -c App1.core.253 -se App1

~/ALCDA2/x64/App1$ cat App1.pmap.253
253:   ./App1
0000000000400000      4K r---- App1
0000000000401000    588K r-x-- App1
0000000000494000    156K r---- App1
00000000004bc000     24K rw--- App1
00000000004c2000     24K rw---   [ anon ]
00000000021b3000    140K rw---   [ anon ]
00007f0fbeef7000      4K -----   [ anon ]
00007f0fbeef8000   8192K rw---   [ anon ]
00007f0fbf6f8000      4K -----   [ anon ]
00007f0fbf6f9000   8192K rw---   [ anon ]
00007f0fbfef9000      4K -----   [ anon ]
00007f0fbfefa000   8192K rw---   [ anon ]
00007f0fc06fa000      4K -----   [ anon ]
00007f0fc06fb000   8192K rw---   [ anon ]
00007f0fc0efb000      4K -----   [ anon ]
00007f0fc0efc000   8192K rw---   [ anon ]
00007ffdf4545000    132K rw---   [ stack ]
00007ffdf45c6000     16K r----   [ anon ]
00007ffdf45ca000      4K r-x--   [ anon ]
 total            42068K

~/ALCDA2/x64/App1$ fg
gdb -c App1.core.253 -se App1


(gdb)
```

10. Compare with the section information in the core dump:

```
(gdb) maintenance info sections
Exec file:
    `/home/coredump/ALCDA2/x64/App1/App1', file type elf64-x86-64.
 [0]     0x00400200->0x00400220 at 0x00000200: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]     0x00400220->0x00400244 at 0x00000220: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]     0x00400248->0x004004d0 at 0x00000248: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]     0x00401000->0x00401017 at 0x00001000: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
 [4]     0x00401018->0x004010f0 at 0x00001018: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
 [5]     0x004010f0->0x004933d0 at 0x000010f0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]     0x004933d0->0x00493f77 at 0x000933d0: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]     0x00493f78->0x00493f81 at 0x00093f78: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [8]     0x00494000->0x004ae73c at 0x00094000: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
 [9]     0x004ae740->0x004bab50 at 0x000ae740: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]     0x004bab50->0x004babfc at 0x000bab50: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]     0x004bc0b0->0x004bc0d8 at 0x000bb0b0: .tdata ALLOC LOAD DATA HAS_CONTENTS
 [12]     0x004bc0d8->0x004bc120 at 0x000bb0d8: .tbss ALLOC
 [13]     0x004bc0d8->0x004bc0e0 at 0x000bb0d8: .preinit_array ALLOC LOAD DATA HAS_CONTENTS
 [14]     0x004bc0e0->0x004bc0f0 at 0x000bb0e0: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [15]     0x004bc0f0->0x004bc100 at 0x000bb0f0: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [16]     0x004bc100->0x004beef4 at 0x000bb100: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [17]     0x004beef8->0x004bf000 at 0x000bdef8: .got ALLOC LOAD DATA HAS_CONTENTS
 [18]     0x004bf000->0x004bf0f0 at 0x000be000: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [19]     0x004bf100->0x004c0c30 at 0x000be100: .data ALLOC LOAD DATA HAS_CONTENTS
 [20]     0x004c0c30->0x004c0c90 at 0x000bfc30: __libc_subfreeres ALLOC LOAD DATA HAS_CONTENTS
 [21]     0x004c0ca0->0x004c1408 at 0x000bfca0: __libc_IO_vtables ALLOC LOAD DATA HAS_CONTENTS
 [22]     0x004c1408->0x004c1410 at 0x000c0408: __libc_atexit ALLOC LOAD DATA HAS_CONTENTS
 [23]     0x004c1420->0x004c7528 at 0x000c0410: .bss ALLOC
```

```
 [24]     0x004c7528->0x004c7558 at 0x000c0410: __libc_freeres_ptrs ALLOC
 [25]     0x00000000->0x00000038 at 0x000c0410: .comment READONLY HAS_CONTENTS
 [26]     0x00000000->0x00000420 at 0x000c0450: .debug_aranges READONLY HAS_CONTENTS
 [27]     0x00000000->0x000372ad at 0x000c0870: .debug_info READONLY HAS_CONTENTS
 [28]     0x00000000->0x000057e8 at 0x000f7b1d: .debug_abbrev READONLY HAS_CONTENTS
 [29]     0x00000000->0x0000aa2b at 0x000fd305: .debug_line READONLY HAS_CONTENTS
 [30]     0x00000000->0x00004d08 at 0x00107d30: .debug_str READONLY HAS_CONTENTS
 [31]     0x00000000->0x0000d4b8 at 0x0010ca38: .debug_loc READONLY HAS_CONTENTS
 [32]     0x00000000->0x000024c0 at 0x00119ef0: .debug_ranges READONLY HAS_CONTENTS
Core file:
    `/home/coredump/ALCDA2/x64/App1/App1.core.253', file type elf64-x86-64.
 [0]     0x00000000->0x00002ec4 at 0x000003f8: note0 READONLY HAS_CONTENTS
 [1]     0x00000000->0x000000d8 at 0x00000518: .reg/253 HAS_CONTENTS
 [2]     0x00000000->0x000000d8 at 0x00000518: .reg HAS_CONTENTS
 [3]     0x00000000->0x00000200 at 0x0000060c: .reg2/253 HAS_CONTENTS
 [4]     0x00000000->0x00000200 at 0x0000060c: .reg2 HAS_CONTENTS
 [5]     0x00000000->0x00000340 at 0x00000820: .reg-xstate/253 HAS_CONTENTS
 [6]     0x00000000->0x00000340 at 0x00000820: .reg-xstate HAS_CONTENTS
 [7]     0x00000000->0x00000080 at 0x00000b74: .note.linuxcore.siginfo/253 HAS_CONTENTS
 [8]     0x00000000->0x00000080 at 0x00000b74: .note.linuxcore.siginfo HAS_CONTENTS
 [9]     0x00000000->0x000000d8 at 0x00000c78: .reg/254 HAS_CONTENTS
 [10]     0x00000000->0x00000200 at 0x00000d6c: .reg2/254 HAS_CONTENTS
 [11]     0x00000000->0x00000340 at 0x00000f80: .reg-xstate/254 HAS_CONTENTS
 [12]     0x00000000->0x00000080 at 0x000012d4: .note.linuxcore.siginfo/254 HAS_CONTENTS
 [13]     0x00000000->0x000000d8 at 0x000013d8: .reg/255 HAS_CONTENTS
 [14]     0x00000000->0x00000200 at 0x000014cc: .reg2/255 HAS_CONTENTS
 [15]     0x00000000->0x00000340 at 0x000016e0: .reg-xstate/255 HAS_CONTENTS
 [16]     0x00000000->0x00000080 at 0x00001a34: .note.linuxcore.siginfo/255 HAS_CONTENTS
 [17]     0x00000000->0x000000d8 at 0x00001b38: .reg/256 HAS_CONTENTS
 [18]     0x00000000->0x00000200 at 0x00001c2c: .reg2/256 HAS_CONTENTS
 [19]     0x00000000->0x00000340 at 0x00001e40: .reg-xstate/256 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
 [20]     0x00000000->0x00000080 at 0x00002194: .note.linuxcore.siginfo/256 HAS_CONTENTS
 [21]     0x00000000->0x000000d8 at 0x00002298: .reg/257 HAS_CONTENTS
 [22]     0x00000000->0x00000200 at 0x0000238c: .reg2/257 HAS_CONTENTS
 [23]     0x00000000->0x00000340 at 0x000025a0: .reg-xstate/257 HAS_CONTENTS
 [24]     0x00000000->0x00000080 at 0x000028f4: .note.linuxcore.siginfo/257 HAS_CONTENTS
 [25]     0x00000000->0x000000d8 at 0x000029f8: .reg/258 HAS_CONTENTS
 [26]     0x00000000->0x00000200 at 0x00002aec: .reg2/258 HAS_CONTENTS
 [27]     0x00000000->0x00000340 at 0x00002d00: .reg-xstate/258 HAS_CONTENTS
 [28]     0x00000000->0x00000080 at 0x00003054: .note.linuxcore.siginfo/258 HAS_CONTENTS
 [29]     0x00000000->0x00000140 at 0x000030e8: .auxv HAS_CONTENTS
 [30]     0x00000000->0x0000007e at 0x0000323c: .note.linuxcore.file/258 HAS_CONTENTS
 [31]     0x00000000->0x0000007e at 0x0000323c: .note.linuxcore.file HAS_CONTENTS
 [32]     0x00401000->0x00494000 at 0x000032bc: load1 ALLOC LOAD READONLY CODE HAS_CONTENTS
 [33]     0x004bc000->0x004c2000 at 0x000962bc: load2 ALLOC LOAD HAS_CONTENTS
 [34]     0x004c2000->0x004c8000 at 0x0009c2bc: load3 ALLOC LOAD HAS_CONTENTS
 [35]     0x021b3000->0x021d6000 at 0x000a22bc: load4 ALLOC LOAD HAS_CONTENTS
 [36]     0x7f0fbeef7000->0x7f0fbeef8000 at 0x000c52bc: load5 ALLOC LOAD READONLY HAS_CONTENTS
 [37]     0x7f0fbeef8000->0x7f0fbf6f8000 at 0x000c62bc: load6 ALLOC LOAD HAS_CONTENTS
 [38]     0x7f0fbf6f8000->0x7f0fbf6f9000 at 0x008c62bc: load7 ALLOC LOAD READONLY HAS_CONTENTS
 [39]     0x7f0fbf6f9000->0x7f0fbfef9000 at 0x008c72bc: load8 ALLOC LOAD HAS_CONTENTS
 [40]     0x7f0fbfef9000->0x7f0fbfefa000 at 0x010c72bc: load9 ALLOC LOAD READONLY HAS_CONTENTS
 [41]     0x7f0fbfefa000->0x7f0fc06fa000 at 0x010c82bc: load10 ALLOC LOAD HAS_CONTENTS
 [42]     0x7f0fc06fa000->0x7f0fc06fb000 at 0x018c82bc: load11 ALLOC LOAD READONLY HAS_CONTENTS
 [43]     0x7f0fc06fb000->0x7f0fc0efb000 at 0x018c92bc: load12 ALLOC LOAD HAS_CONTENTS
 [44]     0x7f0fc0efb000->0x7f0fc0efc000 at 0x020c92bc: load13 ALLOC LOAD READONLY HAS_CONTENTS
 [45]     0x7f0fc0efc000->0x7f0fc16fc000 at 0x020ca2bc: load14 ALLOC LOAD HAS_CONTENTS
 [46]     0x7ffdf4545000->0x7ffdf4566000 at 0x028ca2bc: load15 ALLOC LOAD HAS_CONTENTS
 [47]     0x7ffdf45ca000->0x7ffdf45cb000 at 0x028eb2bc: load16 ALLOC LOAD READONLY CODE HAS_CONTENTS
```

11. Dump **.data** section with possible symbolic information:

```
(gdb) x/256a 0x004bf100
0x4bf100:       0x0     0x0
0x4bf110 <__nptl_nthreads>:     0x6     0x0
0x4bf120 <stack_used>:  0x7f0fbf6f79c0  0x7f0fc16fb9c0
0x4bf130 <stack_cache>: 0x4bf130 <stack_cache>  0x4bf130 <stack_cache>
0x4bf140 <__sched_fifo_max_prio>:       0xffffffffffffffff      0x0
0x4bf150 <__elision_aconf>:     0x300000003     0x300000000
0x4bf160 <_dl_tls_static_size>: 0x1180  0x494a88 <_nl_default_default_domain>
0x4bf170 <__exit_funcs>:        0x4c5a80 <initial>      0x493040 <__gcc_personality_v0>
0x4bf180 <_IO_list_all>:        0x4bf1a0 <_IO_2_1_stderr_>      0x0
0x4bf190:       0x0     0x0
0x4bf1a0 <_IO_2_1_stderr_>:     0xfbad2086      0x0
0x4bf1b0 <_IO_2_1_stderr_+16>:  0x0     0x0
0x4bf1c0 <_IO_2_1_stderr_+32>:  0x0     0x0
0x4bf1d0 <_IO_2_1_stderr_+48>:  0x0     0x0
0x4bf1e0 <_IO_2_1_stderr_+64>:  0x0     0x0
0x4bf1f0 <_IO_2_1_stderr_+80>:  0x0     0x0
0x4bf200 <_IO_2_1_stderr_+96>:  0x0     0x4bf3c0 <_IO_2_1_stdout_>
0x4bf210 <_IO_2_1_stderr_+112>: 0x8000000002    0xffffffffffffffff
0x4bf220 <_IO_2_1_stderr_+128>: 0x0     0x4c5ec0 <_IO_stdfile_2_lock>
0x4bf230 <_IO_2_1_stderr_+144>: 0xffffffffffffffff      0x0
0x4bf240 <_IO_2_1_stderr_+160>: 0x4bf280 <_IO_wide_data_2>      0x0
0x4bf250 <_IO_2_1_stderr_+176>: 0x0     0x0
0x4bf260 <_IO_2_1_stderr_+192>: 0x0     0x0
0x4bf270 <_IO_2_1_stderr_+208>: 0x0     0x4c1060 <_IO_file_jumps>
0x4bf280 <_IO_wide_data_2>:     0x0     0x0
0x4bf290 <_IO_wide_data_2+16>:  0x0     0x0
0x4bf2a0 <_IO_wide_data_2+32>:  0x0     0x0
0x4bf2b0 <_IO_wide_data_2+48>:  0x0     0x0
0x4bf2c0 <_IO_wide_data_2+64>:  0x0     0x0
0x4bf2d0 <_IO_wide_data_2+80>:  0x0     0x0
0x4bf2e0 <_IO_wide_data_2+96>:  0x0     0x0
0x4bf2f0 <_IO_wide_data_2+112>: 0x0     0x0
0x4bf300 <_IO_wide_data_2+128>: 0x0     0x0
0x4bf310 <_IO_wide_data_2+144>: 0x0     0x0
0x4bf320 <_IO_wide_data_2+160>: 0x0     0x0
0x4bf330 <_IO_wide_data_2+176>: 0x0     0x0
0x4bf340 <_IO_wide_data_2+192>: 0x0     0x0
0x4bf350 <_IO_wide_data_2+208>: 0x0     0x0
0x4bf360 <_IO_wide_data_2+224>: 0x0     0x0
0x4bf370 <_IO_wide_data_2+240>: 0x0     0x0
0x4bf380 <_IO_wide_data_2+256>: 0x0     0x0
0x4bf390 <_IO_wide_data_2+272>: 0x0     0x0
0x4bf3a0 <_IO_wide_data_2+288>: 0x0     0x0
0x4bf3b0 <_IO_wide_data_2+304>: 0x4c0e20 <_IO_wfile_jumps>      0x0
0x4bf3c0 <_IO_2_1_stdout_>:     0xfbad2084      0x0
0x4bf3d0 <_IO_2_1_stdout_+16>:  0x0     0x0
0x4bf3e0 <_IO_2_1_stdout_+32>:  0x0     0x0
0x4bf3f0 <_IO_2_1_stdout_+48>:  0x0     0x0
0x4bf400 <_IO_2_1_stdout_+64>:  0x0     0x0
0x4bf410 <_IO_2_1_stdout_+80>:  0x0     0x0
0x4bf420 <_IO_2_1_stdout_+96>:  0x0     0x4bf5e0 <_IO_2_1_stdin_>
0x4bf430 <_IO_2_1_stdout_+112>: 0x8000000001    0xffffffffffffffff
0x4bf440 <_IO_2_1_stdout_+128>: 0x0     0x4c5ed0 <_IO_stdfile_1_lock>
0x4bf450 <_IO_2_1_stdout_+144>: 0xffffffffffffffff      0x0
0x4bf460 <_IO_2_1_stdout_+160>: 0x4bf4a0 <_IO_wide_data_1>      0x0
0x4bf470 <_IO_2_1_stdout_+176>: 0x0     0x0
0x4bf480 <_IO_2_1_stdout_+192>: 0x0     0x0
```

```
--Type <RET> for more, q to quit, c to continue without paging--
0x4bf490 <_IO_2_1_stdout_+208>: 0x0     0x4c1060 <_IO_file_jumps>
0x4bf4a0 <_IO_wide_data_1>:     0x0     0x0
0x4bf4b0 <_IO_wide_data_1+16>:  0x0     0x0
0x4bf4c0 <_IO_wide_data_1+32>:  0x0     0x0
0x4bf4d0 <_IO_wide_data_1+48>:  0x0     0x0
0x4bf4e0 <_IO_wide_data_1+64>:  0x0     0x0
0x4bf4f0 <_IO_wide_data_1+80>:  0x0     0x0
0x4bf500 <_IO_wide_data_1+96>:  0x0     0x0
0x4bf510 <_IO_wide_data_1+112>: 0x0     0x0
0x4bf520 <_IO_wide_data_1+128>: 0x0     0x0
0x4bf530 <_IO_wide_data_1+144>: 0x0     0x0
0x4bf540 <_IO_wide_data_1+160>: 0x0     0x0
0x4bf550 <_IO_wide_data_1+176>: 0x0     0x0
0x4bf560 <_IO_wide_data_1+192>: 0x0     0x0
0x4bf570 <_IO_wide_data_1+208>: 0x0     0x0
0x4bf580 <_IO_wide_data_1+224>: 0x0     0x0
0x4bf590 <_IO_wide_data_1+240>: 0x0     0x0
0x4bf5a0 <_IO_wide_data_1+256>: 0x0     0x0
0x4bf5b0 <_IO_wide_data_1+272>: 0x0     0x0
0x4bf5c0 <_IO_wide_data_1+288>: 0x0     0x0
0x4bf5d0 <_IO_wide_data_1+304>: 0x4c0e20 <_IO_wfile_jumps>      0x0
0x4bf5e0 <_IO_2_1_stdin_>:      0xfbad2088      0x0
0x4bf5f0 <_IO_2_1_stdin_+16>:   0x0     0x0
0x4bf600 <_IO_2_1_stdin_+32>:   0x0     0x0
0x4bf610 <_IO_2_1_stdin_+48>:   0x0     0x0
0x4bf620 <_IO_2_1_stdin_+64>:   0x0     0x0
0x4bf630 <_IO_2_1_stdin_+80>:   0x0     0x0
0x4bf640 <_IO_2_1_stdin_+96>:   0x0     0x0
0x4bf650 <_IO_2_1_stdin_+112>:  0x8000000000    0xffffffffffffffff
0x4bf660 <_IO_2_1_stdin_+128>:  0x0     0x4c5ee0 <_IO_stdfile_0_lock>
0x4bf670 <_IO_2_1_stdin_+144>:  0xffffffffffffffff      0x0
0x4bf680 <_IO_2_1_stdin_+160>:  0x4bf6c0 <_IO_wide_data_0>      0x0
0x4bf690 <_IO_2_1_stdin_+176>:  0x0     0x0
0x4bf6a0 <_IO_2_1_stdin_+192>:  0x0     0x0
0x4bf6b0 <_IO_2_1_stdin_+208>:  0x0     0x4c1060 <_IO_file_jumps>
0x4bf6c0 <_IO_wide_data_0>:     0x0     0x0
0x4bf6d0 <_IO_wide_data_0+16>:  0x0     0x0
0x4bf6e0 <_IO_wide_data_0+32>:  0x0     0x0
0x4bf6f0 <_IO_wide_data_0+48>:  0x0     0x0
0x4bf700 <_IO_wide_data_0+64>:  0x0     0x0
0x4bf710 <_IO_wide_data_0+80>:  0x0     0x0
0x4bf720 <_IO_wide_data_0+96>:  0x0     0x0
0x4bf730 <_IO_wide_data_0+112>: 0x0     0x0
0x4bf740 <_IO_wide_data_0+128>: 0x0     0x0
0x4bf750 <_IO_wide_data_0+144>: 0x0     0x0
0x4bf760 <_IO_wide_data_0+160>: 0x0     0x0
0x4bf770 <_IO_wide_data_0+176>: 0x0     0x0
0x4bf780 <_IO_wide_data_0+192>: 0x0     0x0
0x4bf790 <_IO_wide_data_0+208>: 0x0     0x0
0x4bf7a0 <_IO_wide_data_0+224>: 0x0     0x0
0x4bf7b0 <_IO_wide_data_0+240>: 0x0     0x0
0x4bf7c0 <_IO_wide_data_0+256>: 0x0     0x0
0x4bf7d0 <_IO_wide_data_0+272>: 0x0     0x0
0x4bf7e0 <_IO_wide_data_0+288>: 0x0     0x0
0x4bf7f0 <_IO_wide_data_0+304>: 0x4c0e20 <_IO_wfile_jumps>      0x4bf1a0 <_IO_2_1_stderr_>
0x4bf800 <stdout>:      0x4bf3c0 <_IO_2_1_stdout_>      0x4bf5e0 <_IO_2_1_stdin_>
0x4bf810:       0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x4bf820 <may_shrink_heap.11591>:       0x1fffffff      0x1
```

```
0x4bf830:       0x0     0x0
0x4bf840 <mp_>: 0x20000 0x20000
0x4bf850 <mp_+16>:      0x20000 0x8
0x4bf860 <mp_+32>:      0x0     0x1000000000000
0x4bf870 <mp_+48>:      0x0     0x0
0x4bf880 <mp_+64>:      0x0     0x21b41c0
0x4bf890 <mp_+80>:      0x40    0x408
0x4bf8a0 <mp_+96>:      0x7     0x0
0x4bf8b0:       0x0     0x0
0x4bf8c0 <__memalign_hook>:     0x41aad0 <memalign_hook_ini>    0x41b0e0 <realloc_hook_ini>
0x4bf8d0 <__malloc_hook>:       0x0     0x0
0x4bf8e0 <main_arena>:  0x0     0x0
0x4bf8f0 <main_arena+16>:       0x0     0x0
```

The output is in the following format:

```
address:    value1    value2
```

Because the size of each value is 8 bytes, the next address is +16 bytes or $+10_{hex}$. The addresses can have associated symbolic names:

```
address <name>:  value1  value2
```

For example, from the output above:

```
0x4bf110 <__nptl_nthreads>:     0x6     0x0
```

Each value may also have an associated symbolic value:

```
address <name>:  value1 <name1>    value2
```

For example, from the output above:

```
0x4bf8c0 <__memalign_hook>:     0x41aad0 <memalign_hook_ini>    0x41b0e0 <realloc_hook_ini>
```

12. Explore the contents of memory pointed to by *__nptl_nthreads*, *_nl_default_default_domain*, and *__memalign_hook* addresses (**/x** is for hex, **/d** is for decimals, **/u** is for unsigned decimals, **/g** is for 64-bit values, **/w** is for 32-bit values, **/h** is for 16-bit values, **/b** is for byte values, **/a** is for addresses, **/c** and **/s** are for chars and strings):

```
(gdb) x/d 0x4bf110
0x4bf110 <__nptl_nthreads>:     6

(gdb) x/u &__nptl_nthreads
0x4bf110 <__nptl_nthreads>:     6

(gdb) x/wx 0x4bf110
0x4bf110 <__nptl_nthreads>:     0x00000006

(gdb) x/gx 0x4bf110
0x4bf110 <__nptl_nthreads>:     0x0000000000000006

(gdb) x/hx 0x4bf110
0x4bf110 <__nptl_nthreads>:     0x0006

(gdb) x/bx 0x4bf110
0x4bf110 <__nptl_nthreads>:     0x06
```

```
(gdb) x/2a 0x4bf160
0x4bf160 <_dl_tls_static_size>: 0x1180  0x494a88 <_nl_default_default_domain>
```

Note: Some symbols and addresses (for example, **0x494a88**) belong to read-only sections of executable image. If GDB refuses to read them you may need to run this command:

set trust-readonly-sections on

```
(gdb) x/a &_nl_default_default_domain
0x494a88 <_nl_default_default_domain>:  0x736567617373656d

(gdb) set trust-readonly-sections on

(gdb) x/a 0x494a88
0x494a88 <_nl_default_default_domain>:  0x736567617373656d

(gdb) x/s 0x494a88
0x494a88 <_nl_default_default_domain>:  "messages"

(gdb) x/10a 0x494a88
0x494a88 <_nl_default_default_domain>:  0x736567617373656d      0x6c006f6c00756c00
0x494a98:       0x786c00586c0069        0x7273752f00656372
0x494aa8:       0x6c2f65726168732f      0x656c61636f
0x494ab8 <aliasfile.10131>:     0x2e656c61636f6c2f      0x7361696c61
0x494ac8:       0x0     0x0

(gdb) x/8c 0x494a88
0x494a88 <_nl_default_default_domain>:  109 'm' 101 'e' 115 's' 115 's' 97 'a'  103 'g' 101 'e'
115 's'

(gdb) x/10s 0x494a88
0x494a88 <_nl_default_default_domain>:  "messages"
0x494a91:       "lu"
0x494a94:       "lo"
0x494a97:       "li"
0x494a9a:       "lX"
0x494a9d:       "lx"
0x494aa0:       "rce"
0x494aa4:       "/usr/share/locale"
0x494ab6:       ""
0x494ab7:       ""
```

Note: We see that a hook function is installed for *memalign* but not *malloc*. Please find the following documentation for hook functions here:

https://www.gnu.org/software/libc/manual/html_node/Hooks-for-Malloc.html

13. Explore the contents of memory pointed to by *environ* variable address:

```
(gdb) x/a &environ
0x4c5f48 <environ>:     0x7ffdf45637f8

(gdb) x/10a 0x7ffdf45637f8
0x7ffdf45637f8: 0x7ffdf4565756  0x7ffdf4565766
0x7ffdf4563808: 0x7ffdf456577d  0x7ffdf4565794
0x7ffdf4563818: 0x7ffdf45657a9  0x7ffdf45657c7
0x7ffdf4563828: 0x7ffdf45657d8  0x7ffdf45657f3
0x7ffdf4563838: 0x7ffdf45657fe  0x7ffdf4565812
```

```
(gdb) x/10s 0x7ffdf4565756
0x7ffdf4565756: "SHELL=/bin/bash"
0x7ffdf4565766: "HISTCONTROL=ignoreboth"
0x7ffdf456577d: "WSL_DISTRO_NAME=Debian"
0x7ffdf4565794: "NAME=DESKTOP-IS6V2L0"
0x7ffdf45657a9: "PWD=/home/coredump/ALCDA/App1"
0x7ffdf45657c7: "LOGNAME=coredump"
0x7ffdf45657d8: "MC_TMPDIR=/tmp/mc-coredump"
0x7ffdf45657f3: "MC_SID=192"
0x7ffdf45657fe: "HOME=/home/coredump"
0x7ffdf4565812: "LANG=en_US.UTF-8"
```

14. Now we look at how to perform a memory search. It is not possible to search in the entire virtual memory, only in the valid regions.

```
(gdb) find /g 0x004bc000, 0x004d2000, 6
0x4bd5f8 <_nl_C_LC_NUMERIC+56>
0x4be880 <tunable_list+928>
0x4bea40 <dyn_temp.10655+32>
0x4bf110 <__nptl_nthreads>
warning: Unable to access 16000 bytes of target memory at 0x4c6e18, halting search.
4 patterns found.

(gdb) x/gd 0x4bf110
0x4bf110 <__nptl_nthreads>:     6

(gdb) x/s 0x7ffdf4565756
0x7ffdf4565756: "SHELL=/bin/bash"

(gdb) find 0x7ffdf4565756, +100, "bash"
0x7ffdf4565761
1 pattern found.
```

Note: "bash" is considered a null-terminated array of characters for the search. To search for a string sequence without a null terminator, use a sequence of characters:

```
(gdb) find 0x7ffdf4565756, +100, "bin"
Pattern not found.

(gdb) find 0x7ffdf4565756, +100, 'b', 'i', 'n'
0x7ffdf456575d
1 pattern found.
```

15. Get the list of loaded modules:

```
(gdb) info sharedlibrary
No shared libraries loaded at this time.
```

Note: We don't see any shared libraries because they were statically linked. We also created the version of a dynamically linked *App1.shared* executable. If we load its core dump *App1.shared.core.275*, we see the list of shared libraries:

```
~/ALCDA2/x64/App1$ gdb -c App1.shared.core.275 -se App1.shared
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
```

```
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App1.shared...(no debugging symbols found)...done.
[New LWP 275]
[New LWP 276]
[New LWP 277]
[New LWP 278]
[New LWP 279]
[New LWP 280]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App1.shared'.
#0  0x00007f1ae471e720 in __GI___nanosleep (requested_time=requested_time@entry=0x7ffc74957a90,
remaining=remaining@entry=0x7ffc74957a90) at ../sysdeps/unix/sysv/linux/nanosleep.c:28
28      ../sysdeps/unix/sysv/linux/nanosleep.c: No such file or directory.
[Current thread is 1 (Thread 0x7f1ae4655740 (LWP 275))]
```

```
(gdb) info sharedlibrary
From                To                  Syms Read   Shared Object Library
0x00007f1ae481f5b0  0x00007f1ae482d641  Yes         /lib/x86_64-linux-gnu/libpthread.so.0
0x00007f1ae467a320  0x00007f1ae47c039b  Yes         /lib/x86_64-linux-gnu/libc.so.6
0x00007f1ae4848090  0x00007f1ae4865b20  Yes         /lib64/ld-linux-x86-64.so.2
```

16. Disassemble *bar_one* function and follow the indirect *sleep* function call:

```
(gdb) disassemble bar_one
Dump of assembler code for function bar_one:
   0x0000557e17348145 <+0>:     push   %rbp
   0x0000557e17348146 <+1>:     mov    %rsp,%rbp
   0x0000557e17348149 <+4>:     mov    $0xffffffff,%edi
   0x0000557e1734814e <+9>:     callq  0x557e17348040 <sleep@plt>
   0x0000557e17348153 <+14>:    nop
   0x0000557e17348154 <+15>:    pop    %rbp
   0x0000557e17348155 <+16>:    retq
End of assembler dump.
```

```
(gdb) disassemble 0x557e17348040
Dump of assembler code for function sleep@plt:
   0x0000557e17348040 <+0>:     jmpq   *0x2fda(%rip)        # 0x557e1734b020 <sleep@got.plt>
   0x0000557e17348046 <+6>:     pushq  $0x1
   0x0000557e1734804b <+11>:    jmpq   0x557e17348020
End of assembler dump.
```

17. Dump the annotated value as a memory address interpreting its contents as a symbol:

```
(gdb) p/x 0x0000557e17348046+0x2fda
$1 = 0x557e1734b020

(gdb) x/a 0x557e1734b020
0x557e1734b020 <sleep@got.plt>: 0x7f1ae471e5f0 <__sleep>
```

Note: Since GDB gets shared library images from your analysis system which do not correspond to shared libraries from the crash system, most likely you get some random symbolic information (and also an invalid backtrace from the **bt** command):

```
(gdb) x/a 0x557e1734b020
0x557e1734b020 <sleep@got.plt>:  0x7f1ae471e5f0 <__getpwnam_r+288>
```

Note: You need the original shared library images and debug symbol files from the problem system. To get the right results for this exercise, you can recreate the *App1.shared* core dump (see *main.c* for build instructions if necessary).

18. *App1.shared.pmap.275* also shows library memory regions:

```
(gdb) q

~/ALCDA2/x64/App1$ cat App1.shared.pmap.275
275:   ./App1.shared
0000557e17347000      4K r---- App1.shared
0000557e17348000      4K r-x-- App1.shared
0000557e17349000      4K r---- App1.shared
0000557e1734a000      4K r---- App1.shared
0000557e1734b000      4K rw--- App1.shared
0000557e179ca000    132K rw---   [ anon ]
00007f1ae1e50000      4K -----   [ anon ]
00007f1ae1e51000   8192K rw---   [ anon ]
00007f1ae2651000      4K -----   [ anon ]
00007f1ae2652000   8192K rw---   [ anon ]
00007f1ae2e52000      4K -----   [ anon ]
00007f1ae2e53000   8192K rw---   [ anon ]
00007f1ae3653000      4K -----   [ anon ]
00007f1ae3654000   8192K rw---   [ anon ]
00007f1ae3e54000      4K -----   [ anon ]
00007f1ae3e55000   8204K rw---   [ anon ]
00007f1ae4658000    136K r---- libc-2.28.so
00007f1ae467a000   1312K r-x-- libc-2.28.so
00007f1ae47c2000    304K r---- libc-2.28.so
00007f1ae480e000      4K ----- libc-2.28.so
00007f1ae480f000     16K r---- libc-2.28.so
00007f1ae4813000      8K rw--- libc-2.28.so
00007f1ae4815000     16K rw---   [ anon ]
00007f1ae4819000     24K r---- libpthread-2.28.so
00007f1ae481f000     60K r-x-- libpthread-2.28.so
00007f1ae482e000     24K r---- libpthread-2.28.so
00007f1ae4834000      4K r---- libpthread-2.28.so
00007f1ae4835000      4K rw--- libpthread-2.28.so
00007f1ae4836000     24K rw---   [ anon ]
00007f1ae4847000      4K r---- ld-2.28.so
00007f1ae4848000    120K r-x-- ld-2.28.so
00007f1ae4866000     32K r---- ld-2.28.so
00007f1ae486e000      4K r---- ld-2.28.so
00007f1ae486f000      4K rw--- ld-2.28.so
00007f1ae4870000      4K rw---   [ anon ]
00007ffc74939000    132K rw---   [ stack ]
00007ffc749ac000     16K r----   [ anon ]
00007ffc749b0000      4K r-x--   [ anon ]
 total            43400K
```

Exercise A1 (A64, GDB)

Goal: Learn how to list stack traces, disassemble functions, check their correctness, dump data, get environment.

Patterns: Manual Dump (Process); Stack Trace; Stack Trace Collection; Annotated Disassembly; Paratext; Not My Version; Environment Hint.

1. Load a core dump *App1.core.21174* and *App1* executable from the A64/App1 directory:

```
~/ALCDA2/A64/App1$ gdb -c App1.core.21174 -se App1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App1...
(No debugging symbols found in App1)

warning: Can't open file /home/opc/ALCDA2/App1/App1 during file-backed mapping note processing
[New LWP 21175]
[New LWP 21176]
[New LWP 21177]
[New LWP 21178]
[New LWP 21179]
[New LWP 21174]
Core was generated by `./App1'.
#0  0x000000000040c9b4 in nanosleep ()
[Current thread is 1 (LWP 21175)]
```

2. Set logging to a file in case of lengthy output from some commands and set color highlighting off:

```
(gdb) set logging file App1.log

(gdb) set logging enabled on
Copying output to App1.log.
Copying debug output to App1.log.

(gdb) set style enabled off
```

3. List all threads:

```
(gdb) info threads
  Id   Target Id         Frame
* 1    LWP 21175         0x000000000040c9b4 in nanosleep ()
  2    LWP 21176         0x000000000040c9b4 in nanosleep ()
  3    LWP 21177         0x000000000040c9b4 in nanosleep ()
  4    LWP 21178         0x000000000040c9b4 in nanosleep ()
  5    LWP 21179         0x000000000040c9b4 in nanosleep ()
```

```
  6    LWP 21174         0x000000000040c9b4 in nanosleep ()
```

4. Get the current thread stack trace:

```
(gdb) bt
#0  0x000000000040c9b4 in nanosleep ()
#1  0x0000000000424cb4 in sleep ()
#2  0x00000000004031f8 in bar_one ()
#3  0x000000000040320c in foo_one ()
#4  0x0000000000403224 in thread_one ()
#5  0x0000000000404c34 in start_thread ()
#6  0x0000000000429b60 in thread_start ()
```

5. Get all thread stack traces:

```
(gdb) thread apply all bt

Thread 6 (LWP 21174):
#0  0x000000000040c9b4 in nanosleep ()
#1  0x0000000000424cb4 in sleep ()
#2  0x00000000004033e0 in main ()

Thread 5 (LWP 21179):
#0  0x000000000040c9b4 in nanosleep ()
#1  0x0000000000424cb4 in sleep ()
#2  0x0000000000403318 in bar_five ()
#3  0x000000000040332c in foo_five ()
#4  0x0000000000403344 in thread_five ()
#5  0x0000000000404c34 in start_thread ()
#6  0x0000000000429b60 in thread_start ()

Thread 4 (LWP 21178):
#0  0x000000000040c9b4 in nanosleep ()
#1  0x0000000000424cb4 in sleep ()
#2  0x00000000004032d0 in bar_four ()
#3  0x00000000004032e4 in foo_four ()
#4  0x00000000004032fc in thread_four ()
#5  0x0000000000404c34 in start_thread ()
#6  0x0000000000429b60 in thread_start ()

Thread 3 (LWP 21177):
#0  0x000000000040c9b4 in nanosleep ()
#1  0x0000000000424cb4 in sleep ()
#2  0x0000000000403288 in bar_three ()
#3  0x000000000040329c in foo_three ()
#4  0x00000000004032b4 in thread_three ()
#5  0x0000000000404c34 in start_thread ()
#6  0x0000000000429b60 in thread_start ()

Thread 2 (LWP 21176):
#0  0x000000000040c9b4 in nanosleep ()
#1  0x0000000000424cb4 in sleep ()
#2  0x0000000000403240 in bar_two ()
#3  0x0000000000403254 in foo_two ()
#4  0x000000000040326c in thread_two ()
#5  0x0000000000404c34 in start_thread ()
#6  0x0000000000429b60 in thread_start ()
```

```
Thread 1 (LWP 21175):
#0  0x000000000040c9b4 in nanosleep ()
#1  0x0000000000424cb4 in sleep ()
#2  0x00000000004031f8 in bar_one ()
#3  0x000000000040320c in foo_one ()
#4  0x0000000000403224 in thread_one ()
#5  0x0000000000404c34 in start_thread ()
#6  0x0000000000429b60 in thread_start ()
```

6. Switch to thread #2 and get its stack trace:

```
(gdb) thread 2
[Switching to thread 2 (LWP 21176)]
#0  0x000000000040c9b4 in nanosleep ()

(gdb) bt
#0  0x000000000040c9b4 in nanosleep ()
#1  0x0000000000424cb4 in sleep ()
#2  0x0000000000403240 in bar_two ()
#3  0x0000000000403254 in foo_two ()
#4  0x000000000040326c in thread_two ()
#5  0x0000000000404c34 in start_thread ()
#6  0x0000000000429b60 in thread_start ()

(gdb) info threads
  Id   Target Id         Frame
  1    LWP 21175         0x000000000040c9b4 in nanosleep ()
* 2    LWP 21176         0x000000000040c9b4 in nanosleep ()
  3    LWP 21177         0x000000000040c9b4 in nanosleep ()
  4    LWP 21178         0x000000000040c9b4 in nanosleep ()
  5    LWP 21179         0x000000000040c9b4 in nanosleep ()
  6    LWP 21174         0x000000000040c9b4 in nanosleep ()
```

7. Check that *bar_two* called the *sleep* function by comparing the return address on the call stack from the disassembly output:

```
(gdb) disassemble bar_two
Dump of assembler code for function bar_two:
   0x0000000000403230 <+0>:     stp     x29, x30, [sp, #-16]!
   0x0000000000403234 <+4>:     mov     x29, sp
   0x0000000000403238 <+8>:     mov     w0, #0xffffffff                 // #-1
   0x000000000040323c <+12>:    bl      0x424ba4 <sleep>
   0x0000000000403240 <+16>:    ldp     x29, x30, [sp], #16
   0x0000000000403244 <+20>:    ret
End of assembler dump.
```

We see that the address in the stack trace for the *bar_two* function is the address to return to after calling the *sleep* function.

8. Get *App1* data section from the output of pmap (*App1.pmap.21174*):

```
(gdb) ^Z
[1]+  Stopped                 gdb -c App1.core.21174 -se App1

~/ALCDA2/A64/App1$ cat App1.pmap.21174
21174:   ./App1
0000000000400000    768K r-x-- App1
00000000004c0000    128K rw--- App1
0000000001fa0000    256K rw---   [ anon ]
```

```
0000fffccab40000     64K -----   [ anon ]
0000fffccab50000   8192K rw---   [ anon ]
0000fffccb350000     64K -----   [ anon ]
0000fffccb360000   8192K rw---   [ anon ]
0000fffccbb60000     64K -----   [ anon ]
0000fffccbb70000   8192K rw---   [ anon ]
0000fffccc370000     64K -----   [ anon ]
0000fffccc380000   8192K rw---   [ anon ]
0000fffcccb80000     64K -----   [ anon ]
0000fffcccb90000   8192K rw---   [ anon ]
0000fffccd390000     64K r----   [ anon ]
0000fffccd3a0000     64K r-x--   [ anon ]
0000ffffd3090000    192K rw---   [ stack ]
 total            42752K

~/ALCDA2/A64/App1$ fg
gdb -c App1.core.21174 -se App1


(gdb)
```

9. Compare with the section information in the core dump:

```
(gdb) p/x 0x00000000004c0000+128*1024
$1 = 0x4e0000

(gdb) maintenance info sections
Exec file: `/home/ubuntu/ALCDA2/A64/App1/App1', file type elf64-littleaarch64.
 [0]      0x00400190->0x004001b0 at 0x00000190: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]      0x004001b0->0x004001d4 at 0x000001b0: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]      0x004001d8->0x00400250 at 0x000001d8: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]      0x00400250->0x00400264 at 0x00000250: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
 [4]      0x00400270->0x004002c0 at 0x00000270: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
 [5]      0x004002c0->0x00487098 at 0x000002c0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]      0x00487098->0x00488d68 at 0x00087098: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]      0x00488d68->0x004891b8 at 0x00088d68: __libc_thread_freeres_fn ALLOC LOAD READONLY CODE
HAS_CONTENTS
 [8]      0x004891b8->0x004891c8 at 0x000891b8: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [9]      0x004891d0->0x004a16ad at 0x000891d0: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]     0x004a16ad->0x004a16ae at 0x000a16ad: .stapsdt.base ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]     0x004a16b0->0x004a1de8 at 0x000a16b0: __libc_IO_vtables ALLOC LOAD READONLY DATA HAS_CONTENTS
 [12]     0x004a1de8->0x004a1e50 at 0x000a1de8: __libc_subfreeres ALLOC LOAD READONLY DATA HAS_CONTENTS
 [13]     0x004a1e50->0x004a1e58 at 0x000a1e50: __libc_atexit ALLOC LOAD READONLY DATA HAS_CONTENTS
 [14]     0x004a1e58->0x004a1e68 at 0x000a1e58: __libc_thread_subfreeres ALLOC LOAD READONLY DATA
HAS_CONTENTS
 [15]     0x004a1e68->0x004b047c at 0x000a1e68: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [16]     0x004b047c->0x004b0639 at 0x000b047c: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [17]     0x004cfb20->0x004cfb48 at 0x000bfb20: .tdata ALLOC LOAD DATA HAS_CONTENTS
 [18]     0x004cfb48->0x004cfb98 at 0x000bfb48: .tbss ALLOC
 [19]     0x004cfb48->0x004cfb50 at 0x000bfb48: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [20]     0x004cfb50->0x004cfb60 at 0x000bfb50: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [21]     0x004cfb60->0x004cfb68 at 0x000bfb60: .jcr ALLOC LOAD DATA HAS_CONTENTS
 [22]     0x004cfb68->0x004cff24 at 0x000bfb68: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [23]     0x004cff28->0x004cffe8 at 0x000bff28: .got ALLOC LOAD DATA HAS_CONTENTS
 [24]     0x004cffe8->0x004d0028 at 0x000bffe8: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [25]     0x004d0030->0x004d1580 at 0x000c0030: .data ALLOC LOAD DATA HAS_CONTENTS
 [26]     0x004d1580->0x004d8050 at 0x000c1580: .bss ALLOC
 [27]     0x004d8050->0x004d8088 at 0x000c1580: __libc_freeres_ptrs ALLOC
 [28]     0x00000000->0x00000031 at 0x000c1580: .comment READONLY HAS_CONTENTS
 [29]     0x00000000->0x00001cb0 at 0x000c15b4: .note.stapsdt READONLY HAS_CONTENTS
Core file: `/home/ubuntu/ALCDA2/A64/App1/App1.core.21174', file type elf64-littleaarch64.
 [0]      0x00000000->0x00001c94 at 0x000003c0: note0 READONLY HAS_CONTENTS
 [1]      0x00000000->0x00000110 at 0x000004e0: .reg/21175 HAS_CONTENTS
 [2]      0x00000000->0x00000110 at 0x000004e0: .reg HAS_CONTENTS
```

```
 [3]     0x00000000->0x00000210 at 0x0000060c: .reg2/21175 HAS_CONTENTS
 [4]     0x00000000->0x00000210 at 0x0000060c: .reg2 HAS_CONTENTS
 [5]     0x00000000->0x00000080 at 0x00000830: .note.linuxcore.siginfo/21175 HAS_CONTENTS
 [6]     0x00000000->0x00000080 at 0x00000830: .note.linuxcore.siginfo HAS_CONTENTS
 [7]     0x00000000->0x00000110 at 0x00000934: .reg/21176 HAS_CONTENTS
 [8]     0x00000000->0x00000210 at 0x00000a60: .reg2/21176 HAS_CONTENTS
 [9]     0x00000000->0x00000080 at 0x00000c84: .note.linuxcore.siginfo/21176 HAS_CONTENTS
 [10]    0x00000000->0x00000110 at 0x00000d88: .reg/21177 HAS_CONTENTS
 [11]    0x00000000->0x00000210 at 0x00000eb4: .reg2/21177 HAS_CONTENTS
 [12]    0x00000000->0x00000080 at 0x000010d8: .note.linuxcore.siginfo/21177 HAS_CONTENTS
 [13]    0x00000000->0x00000110 at 0x000011dc: .reg/21178 HAS_CONTENTS
 [14]    0x00000000->0x00000210 at 0x00001308: .reg2/21178 HAS_CONTENTS
 [15]    0x00000000->0x00000080 at 0x0000152c: .note.linuxcore.siginfo/21178 HAS_CONTENTS
 [16]    0x00000000->0x00000110 at 0x00001630: .reg/21179 HAS_CONTENTS
 [17]    0x00000000->0x00000210 at 0x0000175c: .reg2/21179 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
 [18]    0x00000000->0x00000080 at 0x00001980: .note.linuxcore.siginfo/21179 HAS_CONTENTS
 [19]    0x00000000->0x00000110 at 0x00001a84: .reg/21174 HAS_CONTENTS
 [20]    0x00000000->0x00000210 at 0x00001bb0: .reg2/21174 HAS_CONTENTS
 [21]    0x00000000->0x00000080 at 0x00001dd4: .note.linuxcore.siginfo/21174 HAS_CONTENTS
 [22]    0x00000000->0x00000160 at 0x00001e68: .auxv HAS_CONTENTS
 [23]    0x00000000->0x00000076 at 0x00001fdc: .note.linuxcore.file/21174 HAS_CONTENTS
 [24]    0x00000000->0x00000076 at 0x00001fdc: .note.linuxcore.file HAS_CONTENTS
 [25]    0x00400000->0x004c0000 at 0x00002054: load1 ALLOC LOAD READONLY CODE HAS_CONTENTS
 [26]    0x004c0000->0x004e0000 at 0x000c2054: load2 ALLOC LOAD HAS_CONTENTS
 [27]    0x01fa0000->0x01fe0000 at 0x000e2054: load3 ALLOC LOAD HAS_CONTENTS
 [28]    0xfffccab40000->0xfffccab50000 at 0x00122054: load4 ALLOC LOAD READONLY HAS_CONTENTS
 [29]    0xfffccab50000->0xfffccb350000 at 0x00132054: load5 ALLOC LOAD HAS_CONTENTS
 [30]    0xfffccb350000->0xfffccb360000 at 0x00932054: load6 ALLOC LOAD READONLY HAS_CONTENTS
 [31]    0xfffccb360000->0xfffccbb60000 at 0x00942054: load7 ALLOC LOAD HAS_CONTENTS
 [32]    0xfffccbb60000->0xfffccbb70000 at 0x01142054: load8 ALLOC LOAD READONLY HAS_CONTENTS
 [33]    0xfffccbb70000->0xfffccc370000 at 0x01152054: load9 ALLOC LOAD HAS_CONTENTS
 [34]    0xfffccc370000->0xfffccc380000 at 0x01952054: load10 ALLOC LOAD READONLY HAS_CONTENTS
 [35]    0xfffccc380000->0xfffcccb80000 at 0x01962054: load11 ALLOC LOAD HAS_CONTENTS
 [36]    0xfffcccb80000->0xfffcccb90000 at 0x02162054: load12 ALLOC LOAD READONLY HAS_CONTENTS
 [37]    0xfffcccb90000->0xfffccd390000 at 0x02172054: load13 ALLOC LOAD HAS_CONTENTS
 [38]    0xfffccd3a0000->0xfffccd3b0000 at 0x02972054: load14 ALLOC LOAD READONLY CODE HAS_CONTENTS
 [39]    0xffffd3090000->0xffffd30c0000 at 0x02982054: load15 ALLOC LOAD HAS_CONTENTS
```

10. Dump the first 600 addresses from the **.data** section with possible symbolic information:

```
(gdb) x/600a 0x004d0030
0x4d0030:       0x0     0x4d0038 <stack_cache>
0x4d0040 <stack_cache+8>:       0x4d0038 <stack_cache>  0x6
0x4d0050 <stack_used>:  0xfffccb34f140  0xfffccd38f140
0x4d0060 <__sched_fifo_max_prio>:       0xffffffffffffffff      0x890
0x4d0070 <__exit_funcs>:        0x4d5eb0 <initial>      0x486b88 <__gcc_personality_v0>
0x4d0080 <_IO_list_all>:        0x4d0088 <_IO_2_1_stderr_>      0xfbad2086
0x4d0090 <_IO_2_1_stderr_+8>:   0x0     0x0
0x4d00a0 <_IO_2_1_stderr_+24>:  0x0     0x0
0x4d00b0 <_IO_2_1_stderr_+40>:  0x0     0x0
0x4d00c0 <_IO_2_1_stderr_+56>:  0x0     0x0
0x4d00d0 <_IO_2_1_stderr_+72>:  0x0     0x0
0x4d00e0 <_IO_2_1_stderr_+88>:  0x0     0x0
0x4d00f0 <_IO_2_1_stderr_+104>: 0x4d02b0 <_IO_2_1_stdout_>      0x2
0x4d0100 <_IO_2_1_stderr_+120>: 0xffffffffffffffff      0x0
0x4d0110 <_IO_2_1_stderr_+136>: 0x4d6428 <_IO_stdfile_2_lock>   0xffffffffffffffff
0x4d0120 <_IO_2_1_stderr_+152>: 0x0     0x4d0168 <_IO_wide_data_2>
0x4d0130 <_IO_2_1_stderr_+168>: 0x0     0x0
0x4d0140 <_IO_2_1_stderr_+184>: 0x0     0x0
0x4d0150 <_IO_2_1_stderr_+200>: 0x0     0x0
0x4d0160 <_IO_2_1_stderr_+216>: 0x4a1950 <_IO_file_jumps>       0x0
0x4d0170 <_IO_wide_data_2+8>:   0x0     0x0
0x4d0180 <_IO_wide_data_2+24>:  0x0     0x0
```

```
0x4d0190 <_IO_wide_data_2+40>:  0x0     0x0
0x4d01a0 <_IO_wide_data_2+56>:  0x0     0x0
0x4d01b0 <_IO_wide_data_2+72>:  0x0     0x0
0x4d01c0 <_IO_wide_data_2+88>:  0x0     0x0
0x4d01d0 <_IO_wide_data_2+104>: 0x0     0x0
0x4d01e0 <_IO_wide_data_2+120>: 0x0     0x0
0x4d01f0 <_IO_wide_data_2+136>: 0x0     0x0
0x4d0200 <_IO_wide_data_2+152>: 0x0     0x0
0x4d0210 <_IO_wide_data_2+168>: 0x0     0x0
0x4d0220 <_IO_wide_data_2+184>: 0x0     0x0
0x4d0230 <_IO_wide_data_2+200>: 0x0     0x0
0x4d0240 <_IO_wide_data_2+216>: 0x0     0x0
0x4d0250 <_IO_wide_data_2+232>: 0x0     0x0
0x4d0260 <_IO_wide_data_2+248>: 0x0     0x0
0x4d0270 <_IO_wide_data_2+264>: 0x0     0x0
0x4d0280 <_IO_wide_data_2+280>: 0x0     0x0
0x4d0290 <_IO_wide_data_2+296>: 0x0     0x0
0x4d02a0 <_IO_wide_data_2+312>: 0x0     0x4a1800 <_IO_wfile_jumps>
0x4d02b0 <_IO_2_1_stdout_>:     0xfbad2084      0x0
0x4d02c0 <_IO_2_1_stdout_+16>:  0x0     0x0
0x4d02d0 <_IO_2_1_stdout_+32>:  0x0     0x0
0x4d02e0 <_IO_2_1_stdout_+48>:  0x0     0x0
0x4d02f0 <_IO_2_1_stdout_+64>:  0x0     0x0
0x4d0300 <_IO_2_1_stdout_+80>:  0x0     0x0
0x4d0310 <_IO_2_1_stdout_+96>:  0x0     0x4d04d8 <_IO_2_1_stdin_>
0x4d0320 <_IO_2_1_stdout_+112>: 0x1     0xffffffffffffffff
0x4d0330 <_IO_2_1_stdout_+128>: 0x0     0x4d6438 <_IO_stdfile_1_lock>
0x4d0340 <_IO_2_1_stdout_+144>: 0xffffffffffffffff      0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x4d0350 <_IO_2_1_stdout_+160>: 0x4d0390 <_IO_wide_data_1>      0x0
0x4d0360 <_IO_2_1_stdout_+176>: 0x0     0x0
0x4d0370 <_IO_2_1_stdout_+192>: 0x0     0x0
0x4d0380 <_IO_2_1_stdout_+208>: 0x0     0x4a1950 <_IO_file_jumps>
0x4d0390 <_IO_wide_data_1>:     0x0     0x0
0x4d03a0 <_IO_wide_data_1+16>:  0x0     0x0
0x4d03b0 <_IO_wide_data_1+32>:  0x0     0x0
0x4d03c0 <_IO_wide_data_1+48>:  0x0     0x0
0x4d03d0 <_IO_wide_data_1+64>:  0x0     0x0
0x4d03e0 <_IO_wide_data_1+80>:  0x0     0x0
0x4d03f0 <_IO_wide_data_1+96>:  0x0     0x0
0x4d0400 <_IO_wide_data_1+112>: 0x0     0x0
0x4d0410 <_IO_wide_data_1+128>: 0x0     0x0
0x4d0420 <_IO_wide_data_1+144>: 0x0     0x0
0x4d0430 <_IO_wide_data_1+160>: 0x0     0x0
0x4d0440 <_IO_wide_data_1+176>: 0x0     0x0
0x4d0450 <_IO_wide_data_1+192>: 0x0     0x0
0x4d0460 <_IO_wide_data_1+208>: 0x0     0x0
0x4d0470 <_IO_wide_data_1+224>: 0x0     0x0
0x4d0480 <_IO_wide_data_1+240>: 0x0     0x0
0x4d0490 <_IO_wide_data_1+256>: 0x0     0x0
0x4d04a0 <_IO_wide_data_1+272>: 0x0     0x0
0x4d04b0 <_IO_wide_data_1+288>: 0x0     0x0
0x4d04c0 <_IO_wide_data_1+304>: 0x0     0x0
0x4d04d0 <_IO_wide_data_1+320>: 0x4a1800 <_IO_wfile_jumps>      0xfbad2088
0x4d04e0 <_IO_2_1_stdin_+8>:    0x0     0x0
0x4d04f0 <_IO_2_1_stdin_+24>:   0x0     0x0
0x4d0500 <_IO_2_1_stdin_+40>:   0x0     0x0
0x4d0510 <_IO_2_1_stdin_+56>:   0x0     0x0
0x4d0520 <_IO_2_1_stdin_+72>:   0x0     0x0
0x4d0530 <_IO_2_1_stdin_+88>:   0x0     0x0
```

```
0x4d0540 <_IO_2_1_stdin_+104>:  0x0     0x0
0x4d0550 <_IO_2_1_stdin_+120>:  0xffffffffffffffff      0x0
0x4d0560 <_IO_2_1_stdin_+136>:  0x4d6448 <_IO_stdfile_0_lock>   0xffffffffffffffff
0x4d0570 <_IO_2_1_stdin_+152>:  0x0     0x4d05b8 <_IO_wide_data_0>
0x4d0580 <_IO_2_1_stdin_+168>:  0x0     0x0
0x4d0590 <_IO_2_1_stdin_+184>:  0x0     0x0
0x4d05a0 <_IO_2_1_stdin_+200>:  0x0     0x0
0x4d05b0 <_IO_2_1_stdin_+216>:  0x4a1950 <_IO_file_jumps>       0x0
0x4d05c0 <_IO_wide_data_0+8>:   0x0     0x0
0x4d05d0 <_IO_wide_data_0+24>:  0x0     0x0
0x4d05e0 <_IO_wide_data_0+40>:  0x0     0x0
0x4d05f0 <_IO_wide_data_0+56>:  0x0     0x0
0x4d0600 <_IO_wide_data_0+72>:  0x0     0x0
0x4d0610 <_IO_wide_data_0+88>:  0x0     0x0
0x4d0620 <_IO_wide_data_0+104>: 0x0     0x0
0x4d0630 <_IO_wide_data_0+120>: 0x0     0x0
0x4d0640 <_IO_wide_data_0+136>: 0x0     0x0
0x4d0650 <_IO_wide_data_0+152>: 0x0     0x0
0x4d0660 <_IO_wide_data_0+168>: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x4d0670 <_IO_wide_data_0+184>: 0x0     0x0
0x4d0680 <_IO_wide_data_0+200>: 0x0     0x0
0x4d0690 <_IO_wide_data_0+216>: 0x0     0x0
0x4d06a0 <_IO_wide_data_0+232>: 0x0     0x0
0x4d06b0 <_IO_wide_data_0+248>: 0x0     0x0
0x4d06c0 <_IO_wide_data_0+264>: 0x0     0x0
0x4d06d0 <_IO_wide_data_0+280>: 0x0     0x0
0x4d06e0 <_IO_wide_data_0+296>: 0x0     0x0
0x4d06f0 <_IO_wide_data_0+312>: 0x0     0x4a1800 <_IO_wfile_jumps>
0x4d0700 <stderr>:      0x4d0088 <_IO_2_1_stderr_>      0x4d02b0 <_IO_2_1_stdout_>
0x4d0710 <stdin>:       0x4d04d8 <_IO_2_1_stdin_>       0x20000
0x4d0720 <mp_+8>:       0x20000 0x20000
0x4d0730 <mp_+24>:      0x8     0x0
0x4d0740 <mp_+40>:      0x1000000000000 0x0
0x4d0750 <mp_+56>:      0x0     0x0
0x4d0760 <mp_+72>:      0x1fa0f88       0x40
0x4d0770 <mp_+88>:      0x408   0x7
0x4d0780 <mp_+104>:     0x0     0x0
0x4d0790 <main_arena+8>:        0x0     0x0
0x4d07a0 <main_arena+24>:       0x0     0x0
0x4d07b0 <main_arena+40>:       0x0     0x0
0x4d07c0 <main_arena+56>:       0x0     0x0
0x4d07d0 <main_arena+72>:       0x0     0x0
0x4d07e0 <main_arena+88>:       0x0     0x1fa28a0
0x4d07f0 <main_arena+104>:      0x0     0x4d07e8 <main_arena+96>
0x4d0800 <main_arena+120>:      0x4d07e8 <main_arena+96>        0x4d07f8 <main_arena+112>
0x4d0810 <main_arena+136>:      0x4d07f8 <main_arena+112>       0x4d0808 <main_arena+128>
0x4d0820 <main_arena+152>:      0x4d0808 <main_arena+128>       0x4d0818 <main_arena+144>
0x4d0830 <main_arena+168>:      0x4d0818 <main_arena+144>       0x4d0828 <main_arena+160>
0x4d0840 <main_arena+184>:      0x4d0828 <main_arena+160>       0x4d0838 <main_arena+176>
0x4d0850 <main_arena+200>:      0x4d0838 <main_arena+176>       0x4d0848 <main_arena+192>
0x4d0860 <main_arena+216>:      0x4d0848 <main_arena+192>       0x4d0858 <main_arena+208>
0x4d0870 <main_arena+232>:      0x4d0858 <main_arena+208>       0x4d0868 <main_arena+224>
0x4d0880 <main_arena+248>:      0x4d0868 <main_arena+224>       0x4d0878 <main_arena+240>
0x4d0890 <main_arena+264>:      0x4d0878 <main_arena+240>       0x4d0888 <main_arena+256>
0x4d08a0 <main_arena+280>:      0x4d0888 <main_arena+256>       0x4d0898 <main_arena+272>
0x4d08b0 <main_arena+296>:      0x4d0898 <main_arena+272>       0x4d08a8 <main_arena+288>
0x4d08c0 <main_arena+312>:      0x4d08a8 <main_arena+288>       0x4d08b8 <main_arena+304>
0x4d08d0 <main_arena+328>:      0x4d08b8 <main_arena+304>       0x4d08c8 <main_arena+320>
0x4d08e0 <main_arena+344>:      0x4d08c8 <main_arena+320>       0x4d08d8 <main_arena+336>
```

```
0x4d08f0 <main_arena+360>:      0x4d08d8 <main_arena+336>       0x4d08e8 <main_arena+352>
0x4d0900 <main_arena+376>:      0x4d08e8 <main_arena+352>       0x4d08f8 <main_arena+368>
0x4d0910 <main_arena+392>:      0x4d08f8 <main_arena+368>       0x4d0908 <main_arena+384>
0x4d0920 <main_arena+408>:      0x4d0908 <main_arena+384>       0x4d0918 <main_arena+400>
0x4d0930 <main_arena+424>:      0x4d0918 <main_arena+400>       0x4d0928 <main_arena+416>
0x4d0940 <main_arena+440>:      0x4d0928 <main_arena+416>       0x4d0938 <main_arena+432>
0x4d0950 <main_arena+456>:      0x4d0938 <main_arena+432>       0x4d0948 <main_arena+448>
0x4d0960 <main_arena+472>:      0x4d0948 <main_arena+448>       0x4d0958 <main_arena+464>
0x4d0970 <main_arena+488>:      0x4d0958 <main_arena+464>       0x4d0968 <main_arena+480>
0x4d0980 <main_arena+504>:      0x4d0968 <main_arena+480>       0x4d0978 <main_arena+496>
--Type <RET> for more, q to quit, c to continue without paging--
0x4d0990 <main_arena+520>:      0x4d0978 <main_arena+496>       0x4d0988 <main_arena+512>
0x4d09a0 <main_arena+536>:      0x4d0988 <main_arena+512>       0x4d0998 <main_arena+528>
0x4d09b0 <main_arena+552>:      0x4d0998 <main_arena+528>       0x4d09a8 <main_arena+544>
0x4d09c0 <main_arena+568>:      0x4d09a8 <main_arena+544>       0x4d09b8 <main_arena+560>
0x4d09d0 <main_arena+584>:      0x4d09b8 <main_arena+560>       0x4d09c8 <main_arena+576>
0x4d09e0 <main_arena+600>:      0x4d09c8 <main_arena+576>       0x4d09d8 <main_arena+592>
0x4d09f0 <main_arena+616>:      0x4d09d8 <main_arena+592>       0x4d09e8 <main_arena+608>
0x4d0a00 <main_arena+632>:      0x4d09e8 <main_arena+608>       0x4d09f8 <main_arena+624>
0x4d0a10 <main_arena+648>:      0x4d09f8 <main_arena+624>       0x4d0a08 <main_arena+640>
0x4d0a20 <main_arena+664>:      0x4d0a08 <main_arena+640>       0x4d0a18 <main_arena+656>
0x4d0a30 <main_arena+680>:      0x4d0a18 <main_arena+656>       0x4d0a28 <main_arena+672>
0x4d0a40 <main_arena+696>:      0x4d0a28 <main_arena+672>       0x4d0a38 <main_arena+688>
0x4d0a50 <main_arena+712>:      0x4d0a38 <main_arena+688>       0x4d0a48 <main_arena+704>
0x4d0a60 <main_arena+728>:      0x4d0a48 <main_arena+704>       0x4d0a58 <main_arena+720>
0x4d0a70 <main_arena+744>:      0x4d0a58 <main_arena+720>       0x4d0a68 <main_arena+736>
0x4d0a80 <main_arena+760>:      0x4d0a68 <main_arena+736>       0x4d0a78 <main_arena+752>
0x4d0a90 <main_arena+776>:      0x4d0a78 <main_arena+752>       0x4d0a88 <main_arena+768>
0x4d0aa0 <main_arena+792>:      0x4d0a88 <main_arena+768>       0x4d0a98 <main_arena+784>
0x4d0ab0 <main_arena+808>:      0x4d0a98 <main_arena+784>       0x4d0aa8 <main_arena+800>
0x4d0ac0 <main_arena+824>:      0x4d0aa8 <main_arena+800>       0x4d0ab8 <main_arena+816>
0x4d0ad0 <main_arena+840>:      0x4d0ab8 <main_arena+816>       0x4d0ac8 <main_arena+832>
0x4d0ae0 <main_arena+856>:      0x4d0ac8 <main_arena+832>       0x4d0ad8 <main_arena+848>
0x4d0af0 <main_arena+872>:      0x4d0ad8 <main_arena+848>       0x4d0ae8 <main_arena+864>
0x4d0b00 <main_arena+888>:      0x4d0ae8 <main_arena+864>       0x4d0af8 <main_arena+880>
0x4d0b10 <main_arena+904>:      0x4d0af8 <main_arena+880>       0x4d0b08 <main_arena+896>
0x4d0b20 <main_arena+920>:      0x4d0b08 <main_arena+896>       0x4d0b18 <main_arena+912>
0x4d0b30 <main_arena+936>:      0x4d0b18 <main_arena+912>       0x4d0b28 <main_arena+928>
0x4d0b40 <main_arena+952>:      0x4d0b28 <main_arena+928>       0x4d0b38 <main_arena+944>
0x4d0b50 <main_arena+968>:      0x4d0b38 <main_arena+944>       0x4d0b48 <main_arena+960>
0x4d0b60 <main_arena+984>:      0x4d0b48 <main_arena+960>       0x4d0b58 <main_arena+976>
0x4d0b70 <main_arena+1000>:     0x4d0b58 <main_arena+976>       0x4d0b68 <main_arena+992>
0x4d0b80 <main_arena+1016>:     0x4d0b68 <main_arena+992>       0x4d0b78 <main_arena+1008>
0x4d0b90 <main_arena+1032>:     0x4d0b78 <main_arena+1008>      0x4d0b88 <main_arena+1024>
0x4d0ba0 <main_arena+1048>:     0x4d0b88 <main_arena+1024>      0x4d0b98 <main_arena+1040>
0x4d0bb0 <main_arena+1064>:     0x4d0b98 <main_arena+1040>      0x4d0ba8 <main_arena+1056>
0x4d0bc0 <main_arena+1080>:     0x4d0ba8 <main_arena+1056>      0x4d0bb8 <main_arena+1072>
0x4d0bd0 <main_arena+1096>:     0x4d0bb8 <main_arena+1072>      0x4d0bc8 <main_arena+1088>
0x4d0be0 <main_arena+1112>:     0x4d0bc8 <main_arena+1088>      0x4d0bd8 <main_arena+1104>
0x4d0bf0 <main_arena+1128>:     0x4d0bd8 <main_arena+1104>      0x4d0be8 <main_arena+1120>
0x4d0c00 <main_arena+1144>:     0x4d0be8 <main_arena+1120>      0x4d0bf8 <main_arena+1136>
0x4d0c10 <main_arena+1160>:     0x4d0bf8 <main_arena+1136>      0x4d0c08 <main_arena+1152>
0x4d0c20 <main_arena+1176>:     0x4d0c08 <main_arena+1152>      0x4d0c18 <main_arena+1168>
0x4d0c30 <main_arena+1192>:     0x4d0c18 <main_arena+1168>      0x4d0c28 <main_arena+1184>
0x4d0c40 <main_arena+1208>:     0x4d0c28 <main_arena+1184>      0x4d0c38 <main_arena+1200>
0x4d0c50 <main_arena+1224>:     0x4d0c38 <main_arena+1200>      0x4d0c48 <main_arena+1216>
0x4d0c60 <main_arena+1240>:     0x4d0c48 <main_arena+1216>      0x4d0c58 <main_arena+1232>
0x4d0c70 <main_arena+1256>:     0x4d0c58 <main_arena+1232>      0x4d0c68 <main_arena+1248>
0x4d0c80 <main_arena+1272>:     0x4d0c68 <main_arena+1248>      0x4d0c78 <main_arena+1264>
0x4d0c90 <main_arena+1288>:     0x4d0c78 <main_arena+1264>      0x4d0c88 <main_arena+1280>
```

```
0x4d0ca0 <main_arena+1304>:     0x4d0c88 <main_arena+1280>      0x4d0c98 <main_arena+1296>
--Type <RET> for more, q to quit, c to continue without paging--
0x4d0cb0 <main_arena+1320>:     0x4d0c98 <main_arena+1296>      0x4d0ca8 <main_arena+1312>
0x4d0cc0 <main_arena+1336>:     0x4d0ca8 <main_arena+1312>      0x4d0cb8 <main_arena+1328>
0x4d0cd0 <main_arena+1352>:     0x4d0cb8 <main_arena+1328>      0x4d0cc8 <main_arena+1344>
0x4d0ce0 <main_arena+1368>:     0x4d0cc8 <main_arena+1344>      0x4d0cd8 <main_arena+1360>
0x4d0cf0 <main_arena+1384>:     0x4d0cd8 <main_arena+1360>      0x4d0ce8 <main_arena+1376>
0x4d0d00 <main_arena+1400>:     0x4d0ce8 <main_arena+1376>      0x4d0cf8 <main_arena+1392>
0x4d0d10 <main_arena+1416>:     0x4d0cf8 <main_arena+1392>      0x4d0d08 <main_arena+1408>
0x4d0d20 <main_arena+1432>:     0x4d0d08 <main_arena+1408>      0x4d0d18 <main_arena+1424>
0x4d0d30 <main_arena+1448>:     0x4d0d18 <main_arena+1424>      0x4d0d28 <main_arena+1440>
0x4d0d40 <main_arena+1464>:     0x4d0d28 <main_arena+1440>      0x4d0d38 <main_arena+1456>
0x4d0d50 <main_arena+1480>:     0x4d0d38 <main_arena+1456>      0x4d0d48 <main_arena+1472>
0x4d0d60 <main_arena+1496>:     0x4d0d48 <main_arena+1472>      0x4d0d58 <main_arena+1488>
0x4d0d70 <main_arena+1512>:     0x4d0d58 <main_arena+1488>      0x4d0d68 <main_arena+1504>
0x4d0d80 <main_arena+1528>:     0x4d0d68 <main_arena+1504>      0x4d0d78 <main_arena+1520>
0x4d0d90 <main_arena+1544>:     0x4d0d78 <main_arena+1520>      0x4d0d88 <main_arena+1536>
0x4d0da0 <main_arena+1560>:     0x4d0d88 <main_arena+1536>      0x4d0d98 <main_arena+1552>
0x4d0db0 <main_arena+1576>:     0x4d0d98 <main_arena+1552>      0x4d0da8 <main_arena+1568>
0x4d0dc0 <main_arena+1592>:     0x4d0da8 <main_arena+1568>      0x4d0db8 <main_arena+1584>
0x4d0dd0 <main_arena+1608>:     0x4d0db8 <main_arena+1584>      0x4d0dc8 <main_arena+1600>
0x4d0de0 <main_arena+1624>:     0x4d0dc8 <main_arena+1600>      0x4d0dd8 <main_arena+1616>
0x4d0df0 <main_arena+1640>:     0x4d0dd8 <main_arena+1616>      0x4d0de8 <main_arena+1632>
0x4d0e00 <main_arena+1656>:     0x4d0de8 <main_arena+1632>      0x4d0df8 <main_arena+1648>
0x4d0e10 <main_arena+1672>:     0x4d0df8 <main_arena+1648>      0x4d0e08 <main_arena+1664>
0x4d0e20 <main_arena+1688>:     0x4d0e08 <main_arena+1664>      0x4d0e18 <main_arena+1680>
0x4d0e30 <main_arena+1704>:     0x4d0e18 <main_arena+1680>      0x4d0e28 <main_arena+1696>
0x4d0e40 <main_arena+1720>:     0x4d0e28 <main_arena+1696>      0x4d0e38 <main_arena+1712>
0x4d0e50 <main_arena+1736>:     0x4d0e38 <main_arena+1712>      0x4d0e48 <main_arena+1728>
0x4d0e60 <main_arena+1752>:     0x4d0e48 <main_arena+1728>      0x4d0e58 <main_arena+1744>
0x4d0e70 <main_arena+1768>:     0x4d0e58 <main_arena+1744>      0x4d0e68 <main_arena+1760>
0x4d0e80 <main_arena+1784>:     0x4d0e68 <main_arena+1760>      0x4d0e78 <main_arena+1776>
0x4d0e90 <main_arena+1800>:     0x4d0e78 <main_arena+1776>      0x4d0e88 <main_arena+1792>
0x4d0ea0 <main_arena+1816>:     0x4d0e88 <main_arena+1792>      0x4d0e98 <main_arena+1808>
0x4d0eb0 <main_arena+1832>:     0x4d0e98 <main_arena+1808>      0x4d0ea8 <main_arena+1824>
0x4d0ec0 <main_arena+1848>:     0x4d0ea8 <main_arena+1824>      0x4d0eb8 <main_arena+1840>
0x4d0ed0 <main_arena+1864>:     0x4d0eb8 <main_arena+1840>      0x4d0ec8 <main_arena+1856>
0x4d0ee0 <main_arena+1880>:     0x4d0ec8 <main_arena+1856>      0x4d0ed8 <main_arena+1872>
0x4d0ef0 <main_arena+1896>:     0x4d0ed8 <main_arena+1872>      0x4d0ee8 <main_arena+1888>
0x4d0f00 <main_arena+1912>:     0x4d0ee8 <main_arena+1888>      0x4d0ef8 <main_arena+1904>
0x4d0f10 <main_arena+1928>:     0x4d0ef8 <main_arena+1904>      0x4d0f08 <main_arena+1920>
0x4d0f20 <main_arena+1944>:     0x4d0f08 <main_arena+1920>      0x4d0f18 <main_arena+1936>
0x4d0f30 <main_arena+1960>:     0x4d0f18 <main_arena+1936>      0x4d0f28 <main_arena+1952>
0x4d0f40 <main_arena+1976>:     0x4d0f28 <main_arena+1952>      0x4d0f38 <main_arena+1968>
0x4d0f50 <main_arena+1992>:     0x4d0f38 <main_arena+1968>      0x4d0f48 <main_arena+1984>
0x4d0f60 <main_arena+2008>:     0x4d0f48 <main_arena+1984>      0x4d0f58 <main_arena+2000>
0x4d0f70 <main_arena+2024>:     0x4d0f58 <main_arena+2000>      0x4d0f68 <main_arena+2016>
0x4d0f80 <main_arena+2040>:     0x4d0f68 <main_arena+2016>      0x4d0f78 <main_arena+2032>
0x4d0f90 <main_arena+2056>:     0x4d0f78 <main_arena+2032>      0x4d0f88 <main_arena+2048>
0x4d0fa0 <main_arena+2072>:     0x4d0f88 <main_arena+2048>      0x4d0f98 <main_arena+2064>
0x4d0fb0 <main_arena+2088>:     0x4d0f98 <main_arena+2064>      0x4d0fa8 <main_arena+2080>
0x4d0fc0 <main_arena+2104>:     0x4d0fa8 <main_arena+2080>      0x4d0fb8 <main_arena+2096>
--Type <RET> for more, q to quit, c to continue without paging--
0x4d0fd0 <main_arena+2120>:     0x4d0fb8 <main_arena+2096>      0x4d0fc8 <main_arena+2112>
0x4d0fe0 <main_arena+2136>:     0x4d0fc8 <main_arena+2112>      0x0
0x4d0ff0 <main_arena+2152>:     0x0     0x4d0788 <main_arena>
0x4d1000 <main_arena+2168>:     0x0     0x1
0x4d1010 <main_arena+2184>:     0x3f078 0x3f078
0x4d1020 <__morecore>:  0x421c08 <__default_morecore>   0x1
0x4d1030 <__libc_malloc_initialized>:   0xffffffff00000001      0x41cc00 <memalign_hook_ini>
```

```
0x4d1040 <__realloc_hook>:      0x41d688 <realloc_hook_ini>     0x0
0x4d1050 <LogFacility>: 0xffffffff00000008      0xff00000002
0x4d1060 <cached_result.10628>: 0xffffffff      0xffffd30bf6dd
0x4d1070 <program_invocation_name>:     0xffffd30bf6db  0x10000
0x4d1080 <_dl_stack_flags>:     0x6     0x0
0x4d1090 <_dl_load_write_lock+8>:       0x0     0x1
0x4d10a0 <_dl_load_write_lock+24>:      0x0     0x0
0x4d10b0 <_dl_load_write_lock+40>:      0x0     0x0
0x4d10c0 <_dl_load_lock+8>:     0x0     0x1
0x4d10d0 <_dl_load_lock+24>:    0x0     0x0
0x4d10e0 <_dl_load_lock+40>:    0x0     0x42c6a0 <_dl_make_stack_executable>
0x4d10f0 <_dl_correct_cache_id>:        0x200000a03     0x4045a8 <__pthread_init_static_tls>
0x4d1100 <_dl_starting_up>:     0x1     0xfffffffffffffffe
0x4d1110 <_dl_argv>:    0x4d1068 <program_invocation_short_name>        0x0
0x4d1120 <builtin_modules>:     0x48ad20        0x48ac30
0x4d1130 <builtin_modules+16>:  0x7fffffff00000001      0x48ac40
0x4d1140 <builtin_modules+32>:  0x0     0x0
0x4d1150 <builtin_modules+48>:  0x0     0x48ac30
0x4d1160 <builtin_modules+64>:  0x48ad20        0x7fffffff00000001
0x4d1170 <builtin_modules+80>:  0x48ac50        0x0
0x4d1180 <builtin_modules+96>:  0x0     0x0
0x4d1190 <builtin_modules+112>: 0x48ad20        0x48ac60
0x4d11a0 <builtin_modules+128>: 0x7fffffff00000001      0x48ac70
0x4d11b0 <builtin_modules+144>: 0x0     0x0
0x4d11c0 <builtin_modules+160>: 0x0     0x48ac60
0x4d11d0 <builtin_modules+176>: 0x48ad20        0x7fffffff00000001
0x4d11e0 <builtin_modules+192>: 0x48ac88        0x0
0x4d11f0 <builtin_modules+208>: 0x0     0x0
0x4d1200 <builtin_modules+224>: 0x48ad20        0x48aca0
0x4d1210 <builtin_modules+240>: 0x7fffffff00000001      0x48acb0
0x4d1220 <builtin_modules+256>: 0x0     0x0
0x4d1230 <builtin_modules+272>: 0x0     0x48aca0
0x4d1240 <builtin_modules+288>: 0x48ad20        0x7fffffff00000001
0x4d1250 <builtin_modules+304>: 0x48acc0        0x0
0x4d1260 <builtin_modules+320>: 0x0     0x0
0x4d1270 <builtin_modules+336>: 0x48acd0        0x48ad20
0x4d1280 <builtin_modules+352>: 0x7fffffff00000001      0x48ace0
0x4d1290 <builtin_modules+368>: 0x0     0x0
0x4d12a0 <builtin_modules+384>: 0x0     0x48ad20
0x4d12b0 <builtin_modules+400>: 0x48acd0        0x7fffffff00000001
0x4d12c0 <builtin_modules+416>: 0x48acf0        0x0
0x4d12d0 <builtin_modules+432>: 0x0     0x0
0x4d12e0 <builtin_modules+448>: 0x48ad00        0x48ad20
```

The output is in the following format:

```
address:    value1    value2
```

Because the size of each value is 8 bytes, the next address is +16 bytes or $+10_{hex}$. The addresses can have associated symbolic names:

```
address <name>:  value1  value2
```

Each value may also have an associated symbolic value:

```
address <name>:  value1 <name1>    value2
```

For example, from the output above:

```
0x4d1110 <_dl_argv>:    0x4d1068 <program_invocation_short_name>        0x0
```

11. Explore the contents of memory pointed to by *__nptl_nthreads*, *program_invocation_short_name*, and *__realloc_hook* addresses (**/x** is for hex, **/d** is for decimals, **/u** is for unsigned decimals, **/g** is for 64-bit values, **/w** is for 32-bit values, **/h** is for 16-bit values, **/b** is for byte values, **/a** is for addresses, **/c** and **/s** are for chars and strings):

```
(gdb) x/d &__nptl_nthreads
0x4d0048 <__nptl_nthreads>:     6

(gdb) x/u 0x4d0048
0x4d0048 <__nptl_nthreads>:     6

(gdb) x/wx 0x4d0048
0x4d0048 <__nptl_nthreads>:     0x00000006

(gdb) x/gx 0x4d0048
0x4d0048 <__nptl_nthreads>:     0x0000000000000006

(gdb) x/hx 0x4d0048
0x4d0048 <__nptl_nthreads>:     0x0006

(gdb) x/bx 0x4d0048
0x4d0048 <__nptl_nthreads>:     0x06

(gdb) x/a 0x4d1068
0x4d1068 <program_invocation_short_name>:       0xffffd30bf6dd

(gdb) x/a 0xffffd30bf6dd
0xffffd30bf6dd: 0x4744580031707041

(gdb) x/s 0xffffd30bf6dd
0xffffd30bf6dd: "App1"

(gdb) x/8c 0xffffd30bf6dd
0xffffd30bf6dd: 65 'A'  112 'p' 112 'p' 49 '1'  0 '\000'        88 'X'  68 'D'  71 'G'

(gdb) x/10s 0xffffd30bf6dd
0xffffd30bf6dd: "App1"
0xffffd30bf6e2: "XDG_SESSION_ID=6850"
0xffffd30bf6f6: "HOSTNAME=instance-20211109-2004"
0xffffd30bf716: "SELINUX_ROLE_REQUESTED="
0xffffd30bf72e: "TERM=xterm-256color"
0xffffd30bf742: "SHELL=/bin/bash"
0xffffd30bf752: "HISTSIZE=1000"
0xffffd30bf760: "SSH_CLIENT=37.228.238.120 61099 22"
0xffffd30bf783: "SELINUX_USE_CURRENT_RANGE="
0xffffd30bf79e: "SSH_TTY=/dev/pts/1"

(gdb) x/a &__realloc_hook
0x4d1040 <__realloc_hook>:      0x41d688 <realloc_hook_ini>

(gdb) x/10i 0x41d688
   0x41d688 <realloc_hook_ini>: stp     x29, x30, [sp, #-112]!
   0x41d68c <realloc_hook_ini+4>:       mov     x29, sp
   0x41d690 <realloc_hook_ini+8>:       stp     x25, x26, [sp, #64]
   0x41d694 <realloc_hook_ini+12>:      adrp    x25, 0x4d0000
   0x41d698 <realloc_hook_ini+16>:      add     x2, x25, #0x718
```

```
   0x41d69c <realloc_hook_ini+20>:      stp     x21, x22, [sp, #32]
   0x41d6a0 <realloc_hook_ini+24>:      ldr     w3, [x2, #2328]
   0x41d6a4 <realloc_hook_ini+28>:      ldr     x21, 0x41da48
   0x41d6a8 <realloc_hook_ini+32>:      ldr     x2, 0x41da40
   0x41d6ac <realloc_hook_ini+36>:      stp     x19, x20, [sp, #16]
```

Note: We see that a hook function is installed for *realloc*. Please find the following documentation for hook functions here:

https://www.gnu.org/software/libc/manual/html_node/Hooks-for-Malloc.html

12. Explore the contents of memory pointed to by *environ* variable address:

```
(gdb) x/a &environ
0x4d64c8 <environ>:     0xffffd30b8888

(gdb) x/10a 0xffffd30b8888
0xffffd30b8888: 0xffffd30bf6e2  0xffffd30bf6f6
0xffffd30b8898: 0xffffd30bf716  0xffffd30bf72e
0xffffd30b88a8: 0xffffd30bf742  0xffffd30bf752
0xffffd30b88b8: 0xffffd30bf760  0xffffd30bf783
0xffffd30b88c8: 0xffffd30bf79e  0xffffd30bf7b1

(gdb) x/10s 0xffffd30bf6e2
0xffffd30bf6e2: "XDG_SESSION_ID=6850"
0xffffd30bf6f6: "HOSTNAME=instance-20211109-2004"
0xffffd30bf716: "SELINUX_ROLE_REQUESTED="
0xffffd30bf72e: "TERM=xterm-256color"
0xffffd30bf742: "SHELL=/bin/bash"
0xffffd30bf752: "HISTSIZE=1000"
0xffffd30bf760: "SSH_CLIENT=37.228.238.120 61099 22"
0xffffd30bf783: "SELINUX_USE_CURRENT_RANGE="
0xffffd30bf79e: "SSH_TTY=/dev/pts/1"
0xffffd30bf7b1: "USER=opc"
```

13. Now we look at how to perform a memory search. It is not possible to search in the entire virtual memory, only in the valid regions.

```
(gdb) find /g 0x004d0030, 0x005d0030, 6
0x4d0048 <__nptl_nthreads>
0x4d1080 <_dl_stack_flags>
0x4d7e00 <_dl_phnum>
warning: Unable to access 16000 bytes of target memory at 0x4dfb08, halting search.
3 patterns found.

(gdb) x/gd 0x4d0048
0x4d0048 <__nptl_nthreads>:     6

(gdb) find 0xffffd30bf6e2, +1000, "bash"
0xffffd30bf74d
1 pattern found.

(gdb) x/s 0xffffd30bf74d-11
0xffffd30bf742: "SHELL=/bin/bash"
```

Note: "bash" is considered a null-terminated array of characters for the search. To search for a string sequence without a null terminator, use a sequence of characters:

```
(gdb) find 0xffffd30bf6e2, +1000, "bin"
Pattern not found.

(gdb) find 0xffffd30bf6e2, +1000, 'b', 'i', 'n'
0xffffd30bf749
1 pattern found.
```

14. Get the list of loaded modules:

```
(gdb) info sharedlibrary
No shared libraries loaded at this time.
```

Note: We don't see any shared libraries because they were statically linked. We also created the version of a dynamically linked *App1.shared* executable. If we load its core dump *App1.shared.core.184724* from the App1S directory, we see the list of shared libraries:

```
~/ALCDA2/A64/App1S$ gdb -c App1.shared.core.184724 -se App1.shared
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App1.shared...
(No debugging symbols found in App1.shared)
[New LWP 184724]
[New LWP 184725]
[New LWP 184726]
[New LWP 184727]
[New LWP 184728]
[New LWP 184729]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/aarch64-linux-gnu/libthread_db.so.1".
Core was generated by `./App1.shared'.
#0  0x0000ffff81451924 in __GI___clock_nanosleep (clock_id=<optimized out>, clock_id@entry=0,
    flags=flags@entry=0, req=req@entry=0xffffc23e78a8, rem=rem@entry=0xffffc23e78a8)
    at ../sysdeps/unix/sysv/linux/clock_nanosleep.c:78
78      ../sysdeps/unix/sysv/linux/clock_nanosleep.c: No such file or directory.
[Current thread is 1 (Thread 0xffff81585e80 (LWP 184724))]

(gdb) set style enabled off

(gdb) info sharedlibrary
From                To                  Syms Read   Shared Object Library
0x0000ffff813c7040  0x0000ffff814d3f20  Yes         /lib/aarch64-linux-gnu/libc.so.6
0x0000ffff81551c40  0x0000ffff81570064  Yes         /lib/ld-linux-aarch64.so.1
```

15. Disassemble the *bar_one* function and follow the indirect *sleep* function call:

```
(gdb) disassemble bar_one
Dump of assembler code for function bar_one:
   0x0000aaaad0be0894 <+0>:     stp     x29, x30, [sp, #-16]!
   0x0000aaaad0be0898 <+4>:     mov     x29, sp
   0x0000aaaad0be089c <+8>:     mov     w0, #0xffffffff                 // #-1
   0x0000aaaad0be08a0 <+12>:    bl      0xaaaad0be0710 <sleep@plt>
   0x0000aaaad0be08a4 <+16>:    ldp     x29, x30, [sp], #16
   0x0000aaaad0be08a8 <+20>:    ret
End of assembler dump.

(gdb) disassemble 0xaaaad0be0710
Dump of assembler code for function sleep@plt:
   0x0000aaaad0be0710 <+0>:     adrp    x16, 0xaaaad0bf1000
   0x0000aaaad0be0714 <+4>:     ldr     x17, [x16, #4000]
   0x0000aaaad0be0718 <+8>:     add     x16, x16, #0xfa0
   0x0000aaaad0be071c <+12>:    br      x17
End of assembler dump.

(gdb) x/a 0xaaaad0bf1000+4000
0xaaaad0bf1fa0 <sleep@got.plt>: 0xffff81456970 <__sleep>
```

Note: Since GDB gets shared library images from your analysis system which do not correspond to shared libraries from the crash system, most likely you get some random symbolic information (and, also, an invalid backtrace from the **bt** command). This is an example using *App1.shared.core.22442* from the App1 directory:

```
(gdb) bt
#0  0x0000ffff0496dd64 in ?? ()
Backtrace stopped: previous frame identical to this frame (corrupt stack?)

(gdb) info sharedlibrary
From                To                  Syms Read   Shared Object Library
                                        No          /lib64/libpthread.so.0
                                        No          /lib64/libc.so.6
0x0000ffff04ab5bf8  0x0000ffff04ad401c  Yes         /lib/ld-linux-aarch64.so.1

(gdb) disassemble bar_one
Dump of assembler code for function bar_one:
   0x0000000000400728 <+0>:     stp     x29, x30, [sp, #-16]!
   0x000000000040072c <+4>:     mov     x29, sp
   0x0000000000400730 <+8>:     mov     w0, #0xffffffff                 // #-1
   0x0000000000400734 <+12>:    bl      0x400580 <sleep@plt>
   0x0000000000400738 <+16>:    ldp     x29, x30, [sp], #16
   0x000000000040073c <+20>:    ret
End of assembler dump.

(gdb) disassemble 0x400580
Dump of assembler code for function sleep@plt:
   0x0000000000400580 <+0>:     adrp    x16, 0x420000 <__libc_start_main@got.plt>
   0x0000000000400584 <+4>:     ldr     x17, [x16, #8]
   0x0000000000400588 <+8>:     add     x16, x16, #0x8
   0x000000000040058c <+12>:    br      x17
End of assembler dump.

(gdb) x/a 0x420000+8
0x420008 <sleep@got.plt>:       0xffff0496d904
```

Note: You need the original shared library images and debug symbol files from the problem system. To get the right results for this exercise, you can recreate the *App1.shared* core dump (see *main.c* for build instructions if necessary).

16. *App1.shared.pmap.184724* also shows library memory regions:

```
(gdb) q

~/ALCDA2/A64/App1S$ cat App1.shared.pmap.184724
184724:   ./App1.shared
0000aaaad0be0000      4K r-x-- App1.shared
0000aaaad0bf1000      4K r---- App1.shared
0000aaaad0bf2000      4K rw--- App1.shared
0000aaaafe503000    132K rw---   [ anon ]
0000ffff7eb50000     64K -----   [ anon ]
0000ffff7eb60000   8192K rw---   [ anon ]
0000ffff7f360000     64K -----   [ anon ]
0000ffff7f370000   8192K rw---   [ anon ]
0000ffff7fb70000     64K -----   [ anon ]
0000ffff7fb80000   8192K rw---   [ anon ]
0000ffff80380000     64K -----   [ anon ]
0000ffff80390000   8192K rw---   [ anon ]
0000ffff80b90000     64K -----   [ anon ]
0000ffff80ba0000   8192K rw---   [ anon ]
0000ffff813a0000   1572K r-x-- libc.so.6
0000ffff81529000     60K ----- libc.so.6
0000ffff81538000     16K r---- libc.so.6
0000ffff8153c000      8K rw--- libc.so.6
0000ffff8153e000     48K rw---   [ anon ]
0000ffff81550000    172K r-x-- ld-linux-aarch64.so.1
0000ffff81585000      8K rw---   [ anon ]
0000ffff81587000      8K r----   [ anon ]
0000ffff81589000      4K r-x--   [ anon ]
0000ffff8158a000      8K r---- ld-linux-aarch64.so.1
0000ffff8158c000      8K rw--- ld-linux-aarch64.so.1
0000ffffc23c8000    132K rw---   [ stack ]
 total            43468K
```

Exercise A1 (A64, WinDbg Preview)

Goal: Learn how to list stack traces, disassemble functions, check their correctness, dump data, get environment.

Patterns: Manual Dump; Stack Trace; Stack Trace Collection; Annotated Disassembly; Paratext; Not My Version; Environment Hint.

1. Launch WinDbg Preview.

2. Load a core dump *App1.core.21174* from the A64\App1 folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App1\App1.core.21174]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
*** WARNING: Unable to verify timestamp for App1
App1+0xc9b4:
00000000`0040c9b4 d4000001 svc         #0
```

3. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App1\App1.log
Opened log file 'C:\ALCDA2\A64\App1\App1.log'
```

4. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App1\
Symbol search path is: srv*;C:\ALCDA2\A64\App1\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app1\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App1\
*** WARNING: Unable to verify timestamp for App1
```

```
0:000> .reload
..
*** WARNING: Unable to verify timestamp for App1

************* Symbol Loading Error Summary **************
Module name            Error
App1                   The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

5. List all threads:

```
0:000> ~
Unable to get thread data for thread 0
.  0  Id: 52b6.52b7 Suspend: 0 Teb: 00000000`00000000 Unfrozen
Unable to get thread data for thread 1
   1  Id: 52b6.52b8 Suspend: 0 Teb: 00000000`00000000 Unfrozen
Unable to get thread data for thread 2
   2  Id: 52b6.52b9 Suspend: 0 Teb: 00000000`00000000 Unfrozen
Unable to get thread data for thread 3
   3  Id: 52b6.52ba Suspend: 0 Teb: 00000000`00000000 Unfrozen
Unable to get thread data for thread 4
   4  Id: 52b6.52bb Suspend: 0 Teb: 00000000`00000000 Unfrozen
Unable to get thread data for thread 5
   5  Id: 52b6.52b6 Suspend: 0 Teb: 00000000`00000000 Unfrozen
```

Note: WinDbg uses the same output format as for Windows memory dumps. Therefore, some data is either reported as errors or shows 0 or NULL pointer values. However, we see process and threads IDs in the format PID.TID:

```
0:000> .formats 52b6
Evaluate expression:
  Hex:     00000000`000052b6
  Decimal: 21174
  Octal:   0000000000000051266
  Binary:  00000000 00000000 00000000 00000000 00000000 00000000 01010010 10110110
  Chars:   ......R.
  Time:    Thu Jan  1 05:52:54 1970
  Float:   low 2.96711e-041 high 0
  Double:  1.04613e-319

0:000> ? 52b6
Evaluate expression: 21174 = 00000000`000052b6
```

6. Get the current thread stack trace:

```
0:000> k
 # Child-SP          RetAddr               Call Site
00 0000fffc`cd38e5f0 00000000`00424cb4     App1!_libc_nanosleep+0x24
01 0000fffc`cd38e630 00000000`004031f8     App1!sleep+0x110
02 0000fffc`cd38e820 00000000`0040320c     App1!bar_one+0x10
03 0000fffc`cd38e830 00000000`00403224     App1!foo_one+0xc
04 0000fffc`cd38e840 00000000`00404c34     App1!thread_one+0x10
05 0000fffc`cd38e860 00000000`00429b60     App1!start_thread+0xb4
```

```
06 0000fffc`cd38e990 ffffffff`ffffffff     App1!thread_start+0x30
07 0000fffc`cd38e990 00000000`00000000     0xffffffff`ffffffff
```

7. Get all thread stack traces:

```
0:000> ~*k

Unable to get thread data for thread 0
.  0  Id: 52b6.52b7 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`cd38e5f0 00000000`00424cb4     App1!_libc_nanosleep+0x24
01 0000fffc`cd38e630 00000000`004031f8     App1!sleep+0x110
02 0000fffc`cd38e820 00000000`0040320c     App1!bar_one+0x10
03 0000fffc`cd38e830 00000000`00403224     App1!foo_one+0xc
04 0000fffc`cd38e840 00000000`00404c34     App1!thread_one+0x10
05 0000fffc`cd38e860 00000000`00429b60     App1!start_thread+0xb4
06 0000fffc`cd38e990 ffffffff`ffffffff     App1!thread_start+0x30
07 0000fffc`cd38e990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 1
   1  Id: 52b6.52b8 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`ccb7e5f0 00000000`00424cb4     App1!_libc_nanosleep+0x24
01 0000fffc`ccb7e630 00000000`00403240     App1!sleep+0x110
02 0000fffc`ccb7e820 00000000`00403254     App1!bar_two+0x10
03 0000fffc`ccb7e830 00000000`0040326c     App1!foo_two+0xc
04 0000fffc`ccb7e840 00000000`00404c34     App1!thread_two+0x10
05 0000fffc`ccb7e860 00000000`00429b60     App1!start_thread+0xb4
06 0000fffc`ccb7e990 ffffffff`ffffffff     App1!thread_start+0x30
07 0000fffc`ccb7e990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 2
   2  Id: 52b6.52b9 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`cc36e5f0 00000000`00424cb4     App1!_libc_nanosleep+0x24
01 0000fffc`cc36e630 00000000`00403288     App1!sleep+0x110
02 0000fffc`cc36e820 00000000`0040329c     App1!bar_three+0x10
03 0000fffc`cc36e830 00000000`004032b4     App1!foo_three+0xc
04 0000fffc`cc36e840 00000000`00404c34     App1!thread_three+0x10
05 0000fffc`cc36e860 00000000`00429b60     App1!start_thread+0xb4
06 0000fffc`cc36e990 ffffffff`ffffffff     App1!thread_start+0x30
07 0000fffc`cc36e990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 3
   3  Id: 52b6.52ba Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`cbb5e5f0 00000000`00424cb4     App1!_libc_nanosleep+0x24
01 0000fffc`cbb5e630 00000000`004032d0     App1!sleep+0x110
02 0000fffc`cbb5e820 00000000`004032e4     App1!bar_four+0x10
03 0000fffc`cbb5e830 00000000`004032fc     App1!foo_four+0xc
04 0000fffc`cbb5e840 00000000`00404c34     App1!thread_four+0x10
05 0000fffc`cbb5e860 00000000`00429b60     App1!start_thread+0xb4
06 0000fffc`cbb5e990 ffffffff`ffffffff     App1!thread_start+0x30
07 0000fffc`cbb5e990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 4
   4  Id: 52b6.52bb Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`cb34e5f0 00000000`00424cb4     App1!_libc_nanosleep+0x24
01 0000fffc`cb34e630 00000000`00403318     App1!sleep+0x110
```

```
02 0000fffc`cb34e820 00000000`0040332c     App1!bar_five+0x10
03 0000fffc`cb34e830 00000000`00403344     App1!foo_five+0xc
04 0000fffc`cb34e840 00000000`00404c34     App1!thread_five+0x10
05 0000fffc`cb34e860 00000000`00429b60     App1!start_thread+0xb4
06 0000fffc`cb34e990 ffffffff`ffffffff     App1!thread_start+0x30
07 0000fffc`cb34e990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 5
   5  Id: 52b6.52b6 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`d30b8490 00000000`00424cb4     App1!_libc_nanosleep+0x24
01 0000ffff`d30b84d0 00000000`004033e0     App1!sleep+0x110
02 0000ffff`d30b86c0 00000000`0040ec4c     App1!main+0x90
03 0000ffff`d30b8710 00000000`00403090     App1!_libc_start_main+0x304
04 0000ffff`d30b8870 00000000`00000000     App1!start+0x4c
```

8. Switch to thread #1 (threads are numbered from 0) and get its stack trace:

```
0:000> ~1s
App1!_libc_nanosleep+0x24:
00000000`0040c9b4 d4000001 svc         #0

0:001> k
 # Child-SP          RetAddr               Call Site
00 0000fffc`ccb7e5f0 00000000`00424cb4     App1!_libc_nanosleep+0x24
01 0000fffc`ccb7e630 00000000`00403240     App1!sleep+0x110
02 0000fffc`ccb7e820 00000000`00403254     App1!bar_two+0x10
03 0000fffc`ccb7e830 00000000`0040326c     App1!foo_two+0xc
04 0000fffc`ccb7e840 00000000`00404c34     App1!thread_two+0x10
05 0000fffc`ccb7e860 00000000`00429b60     App1!start_thread+0xb4
06 0000fffc`ccb7e990 ffffffff`ffffffff     App1!thread_start+0x30
07 0000fffc`ccb7e990 00000000`00000000     0xffffffff`ffffffff
```

9. Check that *bar_two* called *sleep* function by comparing the return address on the call stack from the disassembly output:

```
0:001> uf bar_two
App1!bar_two:
00000000`00403230 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`00403234 910003fd mov         fp,sp
00000000`00403238 12800000 mov         w0,#-1
00000000`0040323c 9400865a bl          App1!sleep (00000000`00424ba4)
00000000`00403240 a8c17bfd ldp         fp,lr,[sp],#0x10
00000000`00403244 d65f03c0 ret
```

Another way to do that is to disassemble backward the return address and check if the last instruction is BL:

```
0:001> ub 00000000`00403240
App1!thread_one+0xc:
00000000`00403220 97fffff8 bl          App1!foo_one (00000000`00403200)
00000000`00403224 d2800000 mov         x0,#0
00000000`00403228 a8c27bfd ldp         fp,lr,[sp],#0x20
00000000`0040322c d65f03c0 ret
App1!bar_two:
00000000`00403230 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`00403234 910003fd mov         fp,sp
00000000`00403238 12800000 mov         w0,#-1
00000000`0040323c 9400865a bl          App1!sleep (00000000`00424ba4)
```

10. Get *App1* data section from the contents of pmap (*App1.pmap.21174*):

```
21174:   ./App1
0000000000400000    768K r-x-- App1
00000000004c0000    128K rw--- App1
0000000001fa0000    256K rw---   [ anon ]
0000fffccab40000     64K -----   [ anon ]
0000fffccab50000   8192K rw---   [ anon ]
0000fffccb350000     64K -----   [ anon ]
0000fffccb360000   8192K rw---   [ anon ]
0000fffccbb60000     64K -----   [ anon ]
0000fffccbb70000   8192K rw---   [ anon ]
0000fffccc370000     64K -----   [ anon ]
0000fffccc380000   8192K rw---   [ anon ]
0000fffcccb80000     64K -----   [ anon ]
0000fffcccb90000   8192K rw---   [ anon ]
0000fffccd390000     64K r----   [ anon ]
0000fffccd3a0000     64K r-x--   [ anon ]
0000ffffd3090000    192K rw---   [ stack ]
 total            42752K
```

11. Compare with the region information in the core dump:

```
0:001> !address

Mapping file section regions...
Mapping module regions...

        BaseAddress      EndAddress+1        RegionSize     Type       State                 Protect             Usage
----------------------------------------------------------------------------------------------------------------------
+       0`00000000       0`00400000        0`00400000                                                        <unknown>
+       0`00400000       0`004c0000        0`000c0000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ              Image      [App1; "/home/opc/ALCDA2/App1/App1"]
+       0`004c0000       0`004e0000        0`00020000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                 Image      [App1; "/home/opc/ALCDA2/App1/App1"]
+       0`004e0000       0`01fa0000        0`01ac0000                                                        <unknown>
+       0`01fa0000       0`01fe0000        0`00040000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                 <unknown>  [................]
+       0`01fe0000     fffc`cab40000     fffc`c8b60000                                                       <unknown>
+     fffc`cab40000    fffc`cab50000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                <unknown>  [................]
+     fffc`cab50000    fffc`cb350000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE               <unknown>  [................]
+     fffc`cb350000    fffc`cb360000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                <unknown>  [................]
+     fffc`cb360000    fffc`cbb60000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE               <unknown>  [................]
+     fffc`cbb60000    fffc`cbb70000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                <unknown>  [................]
+     fffc`cbb70000    fffc`cc370000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE               <unknown>  [................]
+     fffc`cc370000    fffc`cc380000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                <unknown>  [................]
+     fffc`cc380000    fffc`ccb80000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE               <unknown>  [................]
+     fffc`ccb80000    fffc`ccb90000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                <unknown>  [................]
+     fffc`ccb90000    fffc`cd390000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE               <unknown>  [................]
+     fffc`cd390000    fffc`cd3a0000        0`00010000                                                        <unknown>
+     fffc`cd3a0000    fffc`cd3b0000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ              Image      [linux_vdso_so; "linux-vdso.so.1"]
+     fffc`cd3b0000    ffff`d3090000        3`05ce0000                                                        <unknown>
+     ffff`d3090000    ffff`d30c0000        0`00030000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE               <unknown>  [................]
```

12. Dump the data region with possible symbolic information (we truncated the output):

```
0:001> dps 0`004c0000 0`004e0000
[...]
00000000`004d0fe8  00000000`00000000
00000000`004d0ff0  00000000`00000000
00000000`004d0ff8  00000000`004d0788 App1!main_arena
00000000`004d1000  00000000`00000000
00000000`004d1008  00000000`00000001
00000000`004d1010  00000000`0003f078
00000000`004d1018  00000000`0003f078
00000000`004d1020  00000000`00421c08 App1!_default_morecore
00000000`004d1028  00000000`00000001
00000000`004d1030  ffffffff`00000001
00000000`004d1038  00000000`0041cc00 App1!memalign_hook_ini
00000000`004d1040  00000000`0041d688 App1!realloc_hook_ini
00000000`004d1048  00000000`00000000
00000000`004d1050  ffffffff`00000008
00000000`004d1058  000000ff`00000002
00000000`004d1060  00000000`ffffffff
```

```
00000000`004d1068  0000ffff`d30bf6dd
00000000`004d1070  0000ffff`d30bf6db
00000000`004d1078  00000000`00010000
00000000`004d1080  00000000`00000006
00000000`004d1088  00000000`00000000
00000000`004d1090  00000000`00000000
00000000`004d1098  00000000`00000001
00000000`004d10a0  00000000`00000000
00000000`004d10a8  00000000`00000000
00000000`004d10b0  00000000`00000000
00000000`004d10b8  00000000`00000000
00000000`004d10c0  00000000`00000000
00000000`004d10c8  00000000`00000001
00000000`004d10d0  00000000`00000000
00000000`004d10d8  00000000`00000000
00000000`004d10e0  00000000`00000000
00000000`004d10e8  00000000`0042c6a0 App1!dl_make_stack_executable
00000000`004d10f0  00000002`00000a03
00000000`004d10f8  00000000`004045a8 App1!_pthread_init_static_tls
00000000`004d1100  00000000`00000001
00000000`004d1108  ffffffff`fffffffe
00000000`004d1110  00000000`004d1068 App1!_progname
00000000`004d1118  00000000`00000000
00000000`004d1120  00000000`0048ad20 App1!$d+0xe0
00000000`004d1128  00000000`0048ac30 App1!$d+0x38
00000000`004d1130  7fffffff`00000001
00000000`004d1138  00000000`0048ac40 App1!$d
00000000`004d1140  00000000`00000000
00000000`004d1148  00000000`00000000
00000000`004d1150  00000000`00000000
[...]
```

The output is in the following format:

```
address  value
```

Some values may have associated symbols in the format module!name+offset:

```
address  value  symbol
```

For example, from the output above:

```
00000000`004d1110  00000000`004d1068 App1!_progname
```

To list all values with symbols, we can use the **dpS** command (it doesn't show the value addresses):

```
0:001> dpS 0`004c0000 0`004e0000
 00000000`004d6e70 App1!res
 00000000`004d13c0 App1!nl_global_locale
 00000000`004d13c0 App1!nl_global_locale
 00000000`004d13e0 App1!nl_global_locale+0x20
 00000000`004d13c8 App1!nl_global_locale+0x8
 00000000`00403190 App1!frame_dummy
 00000000`00403140 App1!__do_global_dtors_aux
 00000000`00402ffc App1!fini
 00000000`0048a2d0 App1!$d+0x20
 00000000`0048a2f0 App1!$d+0x40
 00000000`0048a308 App1!$d+0x58
 00000000`0048a320 App1!$d+0x70
 00000000`0048a330 App1!$d+0x80
```

```
00000000`0048a348 App1!$d+0x98
00000000`0048a358 App1!$d+0xa8
00000000`0048a368 App1!$d+0xb8
00000000`0048a380 App1!$d+0xd0
00000000`0048a398 App1!$d+0xe8
00000000`0048a3c0 App1!$d+0x110
00000000`0048a3d8 App1!$d+0x128
00000000`0048a3e8 App1!$d+0x138
00000000`0048a400 App1!$d+0x150
00000000`0048a418 App1!$d+0x168
00000000`0048a438 App1!$d+0x188
00000000`0048a450 App1!$d+0x1a0
00000000`0048a470 App1!$d+0x1c0
00000000`0048a488 App1!$d+0x1d8
00000000`0048a4a0 App1!$d+0x1f0
00000000`0048a4b8 App1!$d+0x208
00000000`0048a4d0 App1!$d+0x220
00000000`00409a50 App1!_pthread_key_create
00000000`004231c0 App1!_memmove_generic
00000000`004231d0 App1!_memcpy_generic
00000000`00423fc0 App1!_memset_generic
00000000`00424480 App1!_strlen_generic
00000000`00424480 App1!_strlen_generic
00000000`004d0038 App1!stack_cache
00000000`004d0038 App1!stack_cache
00000000`004d5eb0 App1!initial
00000000`00486b88 App1!_gcc_personality_v0
00000000`004d0088 App1!IO_2_1_stderr_
00000000`004d02b0 App1!IO_2_1_stdout_
00000000`004d6428 App1!IO_stdfile_2_lock
00000000`004d0168 App1!IO_wide_data_2
00000000`004a1950 App1!IO_file_jumps
00000000`004a1800 App1!IO_wfile_jumps
00000000`004d04d8 App1!IO_2_1_stdin_
00000000`004d6438 App1!IO_stdfile_1_lock
00000000`004d0390 App1!IO_wide_data_1
00000000`004a1950 App1!IO_file_jumps
00000000`004a1800 App1!IO_wfile_jumps
00000000`004d6448 App1!IO_stdfile_0_lock
00000000`004d05b8 App1!IO_wide_data_0
00000000`004a1950 App1!IO_file_jumps
00000000`004a1800 App1!IO_wfile_jumps
00000000`004d0088 App1!IO_2_1_stderr_
00000000`004d02b0 App1!IO_2_1_stdout_
00000000`004d04d8 App1!IO_2_1_stdin_
00000000`004d07e8 App1!main_arena+0x60
00000000`004d07e8 App1!main_arena+0x60
00000000`004d07f8 App1!main_arena+0x70
00000000`004d07f8 App1!main_arena+0x70
00000000`004d0808 App1!main_arena+0x80
00000000`004d0808 App1!main_arena+0x80
00000000`004d0818 App1!main_arena+0x90
00000000`004d0818 App1!main_arena+0x90
00000000`004d0828 App1!main_arena+0xa0
00000000`004d0828 App1!main_arena+0xa0
00000000`004d0838 App1!main_arena+0xb0
00000000`004d0838 App1!main_arena+0xb0
00000000`004d0848 App1!main_arena+0xc0
00000000`004d0848 App1!main_arena+0xc0
00000000`004d0858 App1!main_arena+0xd0
```

```
00000000`004d0858 App1!main_arena+0xd0
00000000`004d0868 App1!main_arena+0xe0
00000000`004d0868 App1!main_arena+0xe0
00000000`004d0878 App1!main_arena+0xf0
00000000`004d0878 App1!main_arena+0xf0
00000000`004d0888 App1!main_arena+0x100
00000000`004d0888 App1!main_arena+0x100
00000000`004d0898 App1!main_arena+0x110
[...]
00000000`004d0fc8 App1!main_arena+0x840
00000000`004d0788 App1!main_arena
00000000`00421c08 App1!_default_morecore
00000000`0041cc00 App1!memalign_hook_ini
00000000`0041d688 App1!realloc_hook_ini
00000000`0042c6a0 App1!dl_make_stack_executable
00000000`004045a8 App1!_pthread_init_static_tls
00000000`004d1068 App1!_progname
00000000`0048ad20 App1!$d+0xe0
00000000`0048ac30 App1!$d+0x38
00000000`0048ac40 App1!$d
00000000`0048ac30 App1!$d+0x38
00000000`0048ad20 App1!$d+0xe0
00000000`0048ac50 App1!$d+0x10
00000000`0048ad20 App1!$d+0xe0
00000000`0048ac60 App1!$d+0x20
00000000`0048ac70 App1!$d+0x30
00000000`0048ac60 App1!$d+0x20
00000000`0048ad20 App1!$d+0xe0
00000000`0048ac88 App1!$d+0x48
00000000`0048ad20 App1!$d+0xe0
00000000`0048aca0 App1!$d+0x60
00000000`0048acb0 App1!$d+0x70
00000000`0048aca0 App1!$d+0x60
00000000`0048ad20 App1!$d+0xe0
00000000`0048acc0 App1!$d+0x80
00000000`0048acd0 App1!$d+0x90
00000000`0048ad20 App1!$d+0xe0
00000000`0048ace0 App1!$d+0xa0
00000000`0048ad20 App1!$d+0xe0
00000000`0048acd0 App1!$d+0x90
00000000`0048acf0 App1!$d+0xb0
00000000`0048ad00 App1!$d+0xc0
00000000`0048ad20 App1!$d+0xe0
00000000`0048ad18 App1!$d+0xd8
00000000`0048ad20 App1!$d+0xe0
00000000`0048ad00 App1!$d+0xc0
00000000`0048ad30 App1!$d+0xf0
00000000`0048ad48 App1!$d+0x108
00000000`0048ad20 App1!$d+0xe0
00000000`0048ad58 App1!$d+0x118
00000000`0048ad20 App1!$d+0xe0
00000000`0048ad48 App1!$d+0x108
00000000`0048ad70 App1!$d+0x130
00000000`0048b888 App1!nl_C_LC_CTYPE
00000000`00499f18 App1!nl_C_LC_NUMERIC
00000000`00499f88 App1!nl_C_LC_TIME
00000000`0049aec0 App1!nl_C_LC_COLLATE
00000000`00499d58 App1!nl_C_LC_MONETARY
00000000`00499ce0 App1!nl_C_LC_MESSAGES
00000000`0049a9e0 App1!nl_C_LC_PAPER
```

```
00000000`0049aa38 App1!nl_C_LC_NAME
00000000`0049aac0 App1!nl_C_LC_ADDRESS
00000000`0049ab98 App1!nl_C_LC_TELEPHONE
00000000`0049ac10 App1!nl_C_LC_MEASUREMENT
00000000`0049ad08 App1!nl_C_LC_IDENTIFICATION
00000000`0048d1c0 App1!nl_C_LC_CTYPE_class+0x100
00000000`0048c2c0 App1!nl_C_LC_CTYPE_tolower+0x200
00000000`0048c8c0 App1!nl_C_LC_CTYPE_toupper+0x200
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`00497450 App1!nl_C_name
00000000`004975d0 App1!nl_C_locobj+0x158
00000000`00498850 App1!$d+0x30
00000000`00498850 App1!$d+0x30
00000000`00465268 App1!_libc_dlopen_mode
00000000`004651ec App1!_libc_dlsym
00000000`0046517c App1!_libc_dlclose
00000000`00465460 App1!dl_initial_error_catch_tsd
00000000`0049b540 App1!nl_default_default_domain
00000000`0047e9e8 App1!_dlopen
00000000`0047ea3c App1!_dlclose
00000000`0047ea98 App1!_dlsym
00000000`0047eb4c App1!_dlvsym
00000000`00470f60 App1!_dlerror
00000000`00471324 App1!_dladdr
00000000`00471330 App1!_dladdr1
00000000`00471470 App1!_dlinfo
00000000`00471528 App1!_dlmopen
00000000`004d1078 App1!dl_pagesize
00000000`004a1e68 App1!_EH_FRAME_BEGIN__
00000000`004cfb20 App1!
00000000`004d5618 App1!static_map
00000000`00403f44 App1!_reclaim_stacks
00000000`004d1588 App1!object.6205
00000000`004d7d40 App1!_libc_multiple_threads
00000000`004d5a78 App1!static_slotinfo
00000000`004d78e0 App1!_fork_generation
00000000`004d6570 App1!fork_handler_pool+0x8
00000000`0048a618 App1!unsecure_envvars.10865+0x118
00000000`004046f0 App1!_wait_lookup_done
00000000`00400040 App1+0x40
????????`????????
```

13. Explore the contents of memory pointed to by *App1!memalign_hook_ini* and *App1!_progname* addresses:

```
0:001> u 00000000`0041cc00
App1!memalign_hook_ini:
00000000`0041cc00 a9b97bfd stp         fp,lr,[sp,#-0x70]!
00000000`0041cc04 910003fd mov         fp,sp
00000000`0041cc08 a9025bf5 stp         x21,x22,[sp,#0x20]
00000000`0041cc0c 900005b6 adrp        x22,App1!+0x18 (00000000`004d0000)
00000000`0041cc10 58004815 ldr         x21,App1!$d (00000000`0041d510)
00000000`0041cc14 911c62c2 add         x2,x22,#0x718
00000000`0041cc18 a90153f3 stp         x19,x20,[sp,#0x10]
00000000`0041cc1c a90363f7 stp         x23,x24,[sp,#0x30]

0:001> dp App1!_progname
00000000`004d1068  0000ffff`d30bf6dd 0000ffff`d30bf6db
00000000`004d1078  00000000`00010000 00000000`00000006
00000000`004d1088  00000000`00000000 00000000`00000000
00000000`004d1098  00000000`00000001 00000000`00000000
00000000`004d10a8  00000000`00000000 00000000`00000000
00000000`004d10b8  00000000`00000000 00000000`00000000
00000000`004d10c8  00000000`00000001 00000000`00000000
00000000`004d10d8  00000000`00000000 00000000`00000000

0:001> dc 0000ffff`d30bf6dd
0000ffff`d30bf6dd  31707041 47445800 5345535f 4e4f4953  App1.XDG_SESSION
0000ffff`d30bf6ed  3d44495f 30353836 534f4800 4d414e54  _ID=6850.HOSTNAM
0000ffff`d30bf6fd  6e693d45 6e617473 322d6563 31313230  E=instance-20211
0000ffff`d30bf70d  2d393031 34303032 4c455300 58554e49  109-2004.SELINUX
0000ffff`d30bf71d  4c4f525f 45525f45 53455551 3d444554  _ROLE_REQUESTED=
0000ffff`d30bf72d  52455400 74783d4d 2d6d7265 63363532  .TERM=xterm-256c
0000ffff`d30bf73d  726f6c6f 45485300 2f3d4c4c 2f6e6962  olor.SHELL=/bin/
0000ffff`d30bf74d  68736162 53494800 5a495354 30313d45  bash.HISTSIZE=10

0:001> da 0000ffff`d30bf6dd
0000ffff`d30bf6dd  "App1"

0:001> db 0000ffff`d30bf6dd
0000ffff`d30bf6dd  41 70 70 31 00 58 44 47-5f 53 45 53 53 49 4f 4e  App1.XDG_SESSION
0000ffff`d30bf6ed  5f 49 44 3d 36 38 35 30-00 48 4f 53 54 4e 41 4d  _ID=6850.HOSTNAM
0000ffff`d30bf6fd  45 3d 69 6e 73 74 61 6e-63 65 2d 32 30 32 31 31  E=instance-20211
0000ffff`d30bf70d  31 30 39 2d 32 30 30 34-00 53 45 4c 49 4e 55 58  109-2004.SELINUX
0000ffff`d30bf71d  5f 52 4f 4c 45 5f 52 45-51 55 45 53 54 45 44 3d  _ROLE_REQUESTED=
0000ffff`d30bf72d  00 54 45 52 4d 3d 78 74-65 72 6d 2d 32 35 36 63  .TERM=xterm-256c
0000ffff`d30bf73d  6f 6c 6f 72 00 53 48 45-4c 4c 3d 2f 62 69 6e 2f  olor.SHELL=/bin/
0000ffff`d30bf74d  62 61 73 68 00 48 49 53-54 53 49 5a 45 3d 31 30  bash.HISTSIZE=10
```

Note: We see that a hook function is installed for *memalign* and *realloc*. Please find the following documentation for hook functions here:

https://www.gnu.org/software/libc/manual/html_node/Hooks-for-Malloc.html

14. Explore the contents of memory pointed to by *environ* variable:

```
0:001> dp environ
00000000`004d64c8  0000ffff`d30b8888 00000000`00000000
00000000`004d64d8  00000000`00000000 00000000`00000000
00000000`004d64e8  00000000`00000000 00000000`00000000
00000000`004d64f8  00000000`00000000 00000000`00000000
00000000`004d6508  00000000`00000000 00000000`00000000
```

```
00000000`004d6518  00000000`00000000 00000000`00000000
00000000`004d6528  00000000`00000000 00000000`00000000
00000000`004d6538  00000000`00000000 00000000`00000000

0:001> dp 0000ffff`d30b8888
0000ffff`d30b8888  0000ffff`d30bf6e2 0000ffff`d30bf6f6
0000ffff`d30b8898  0000ffff`d30bf716 0000ffff`d30bf72e
0000ffff`d30b88a8  0000ffff`d30bf742 0000ffff`d30bf752
0000ffff`d30b88b8  0000ffff`d30bf760 0000ffff`d30bf783
0000ffff`d30b88c8  0000ffff`d30bf79e 0000ffff`d30bf7b1
0000ffff`d30b88d8  0000ffff`d30bf7ba 0000ffff`d30bfe72
0000ffff`d30b88e8  0000ffff`d30bfe8b 0000ffff`d30bfee5
0000ffff`d30b88f8  0000ffff`d30bfeff 0000ffff`d30bff10

0:001> da 0000ffff`d30bf6e2
0000ffff`d30bf6e2  "XDG_SESSION_ID=6850"

0:001> dpa 0000ffff`d30b8888
0000ffff`d30b8888  0000ffff`d30bf6e2 "XDG_SESSION_ID=6850"
0000ffff`d30b8890  0000ffff`d30bf6f6 "HOSTNAME=instance-20211109-2004"
0000ffff`d30b8898  0000ffff`d30bf716 "SELINUX_ROLE_REQUESTED="
0000ffff`d30b88a0  0000ffff`d30bf72e "TERM=xterm-256color"
0000ffff`d30b88a8  0000ffff`d30bf742 "SHELL=/bin/bash"
0000ffff`d30b88b0  0000ffff`d30bf752 "HISTSIZE=1000"
0000ffff`d30b88b8  0000ffff`d30bf760 "SSH_CLIENT=37.228.238.120 61099 22"
0000ffff`d30b88c0  0000ffff`d30bf783 "SELINUX_USE_CURRENT_RANGE="
0000ffff`d30b88c8  0000ffff`d30bf79e "SSH_TTY=/dev/pts/1"
0000ffff`d30b88d0  0000ffff`d30bf7b1 "USER=opc"
0000ffff`d30b88d8  0000ffff`d30bf7ba "LS_COLORS=rs=0:di=38;5;27:ln=38;5;51:mh=44;38;5;15:pi=4"
0000ffff`d30b88e0  0000ffff`d30bfe72 "MAIL=/var/spool/mail/opc"
0000ffff`d30b88e8  0000ffff`d30bfe8b "PATH=/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:"
0000ffff`d30b88f0  0000ffff`d30bfee5 "PWD=/home/opc/ALCDA2/App1"
0000ffff`d30b88f8  0000ffff`d30bfeff "LANG=en_US.UTF-8"
0000ffff`d30b8900  0000ffff`d30bff10 "SELINUX_LEVEL_REQUESTED="
```

17. Now we look at how to perform a memory search.

```
0:000> s 0`004c0000 0`004f0000 6
00000000`004c002a  06 9a 05 9b 04 9c 03 02-49 0a de dd dc db da d9  ........I.......
00000000`004c007e  06 9a 05 45 95 0a 96 09-46 9b 04 9c 03 6c 0a de  ...E....F....l..
00000000`004c012d  06 00 00 00 41 0e a0 01-9d 14 9e 13 41 0d 1d 46  ....A.......A..F
00000000`004c018d  06 9e 05 41 0d 1d 41 93-04 94 03 5b 0a de dd d4  ...A..A....[....
00000000`004c020d  06 9e 05 41 0d 1d 41 93-04 94 03 5b 0a de dd d4  ...A..A....[....
00000000`004c0285  06 9e 05 41 0d 1d 42 93-04 94 03 95 02 96 01 75  ...A..B........u
00000000`004c04fe  06 04 00 00 80 07 88 01-90 0b 00 b0 08 04 00 00  ................
00000000`004cfe78  06 00 00 00 00 00 00 00-50 01 3a cd fc ff 00 00  ........P.:.....
00000000`004d0048  06 00 00 00 00 00 00 00-40 f1 34 cb fc ff 00 00  ........@.4.....
00000000`004d1080  06 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
00000000`004d7e00  06 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
```

Note: It is possible to search through non-accessible regions as well; they are ignored:

```
0:000> s-q 0 Lffffff 6
00000000`0048b208  00000000`00000006 00000000`0000006f
00000000`0048be40  00000000`00000006 00000018`00000001
00000000`0048bf28  00000000`00000006 00000018`00000001
00000000`0048bfc0  00000000`00000006 00000018`00000001
00000000`00499f50  00000000`00000006 00000000`00498240
00000000`0049b728  00000000`00000006 00000000`00000002
00000000`004cfe78  00000000`00000006 0000fffc`cd3a0150
```

```
00000000`004d0048  00000000`00000006 0000fffc`cb34f140
00000000`004d1080  00000000`00000006 00000000`00000000
00000000`004d7e00  00000000`00000006 00000000`00000000
[...]

0:000> s-a 0000ffff`d30b88a8 L100000 "bin"
0000ffff`d30bf749  62 69 6e 2f 62 61 73 68-00 48 49 53 54 53 49 5a  bin/bash.HISTSIZ
0000ffff`d30bfe9b  62 69 6e 3a 2f 75 73 72-2f 62 69 6e 3a 2f 75 73  bin:/usr/bin:/us
0000ffff`d30bfea4  62 69 6e 3a 2f 75 73 72-2f 6c 6f 63 61 6c 2f 73  bin:/usr/local/s
0000ffff`d30bfeb4  62 69 6e 3a 2f 75 73 72-2f 73 62 69 6e 3a 2f 68  bin:/usr/sbin:/h
0000ffff`d30bfebe  62 69 6e 3a 2f 68 6f 6d-65 2f 6f 70 63 2f 2e 6c  bin:/home/opc/.l
0000ffff`d30bfed3  62 69 6e 3a 2f 68 6f 6d-65 2f 6f 70 63 2f 62 69  bin:/home/opc/bi
0000ffff`d30bfee1  62 69 6e 00 50 57 44 3d-2f 68 6f 6d 65 2f 6f 70  bin.PWD=/home/op
0000ffff`d30bffa5  62 69 6e 2f 6c 65 73 73-70 69 70 65 2e 73 68 20  bin/lesspipe.sh
```

Note: It is also possible to show all possible string fragments if any:

```
0:000> s-sa 0 Lfffffff
00000000`00400001  "ELF"
00000000`00400018  "D0@"
00000000`0040019c  "GNU"
00000000`004001bc  "GNU"
00000000`004001d1  "48y"
[...]
00000000`004a12b0  "weak version `"
00000000`004a12c0  "' not found (required by "
00000000`004a12e0  "version `"
00000000`004a12f0  "version lookup error"
00000000`004a1308  "cannot allocate version referenc"
00000000`004a1328  "e table"
00000000`004a1330  " of Verneed record"
00000000`004a1348  "RTLD_NEXT used in code not dynam"
00000000`004a1368  "ically loaded"
[...]
00000000`004d6588  "D?@"
00000000`004d7880  "@}M"
00000000`004d7d08  "xZM"
00000000`004d7d38  "peM"
00000000`01fa0700  "pnM"
00000000`01fa1680  "linux-vdso.so.1"
00000000`01fa16e0  "tls/atomics/"
```

15. Get the list of loaded modules:

```
0:001> lm
start             end                 module name
00000000`00400000 00000000`004e0000   App1     T (service symbols: ELF Export Symbols)
c:\alcda2\a64\app1\App1

0:001> lmv
start             end                 module name
00000000`00400000 00000000`004e0000   App1     T (service symbols: ELF Export Symbols)
c:\alcda2\a64\app1\App1
    Loaded symbol image file: App1
    Image path: /home/opc/ALCDA2/App1/App1
    Image name: App1
    Browse all global symbols  functions  data
    Timestamp:        unavailable (FFFFFFFE)
    CheckSum:         missing
```

```
    ImageSize:        000E0000
    Details:
0000fffc`cd3a0000 0000fffc`cd3b0000   linux_vdso_so T (service symbols: ELF In Memory Symbols)
    Loaded symbol image file: linux-vdso.so.1
    Image path: linux-vdso.so.1
    Image name: linux-vdso.so.1
    Browse all global symbols  functions  data
    Timestamp:        unavailable (FFFFFFFE)
    CheckSum:         missing
    ImageSize:        00010000
    Details:
```

Note: We don't see shared libraries except *vdso* (https://man7.org/linux/man-pages/man7/vdso.7.html) because they were statically linked. We also created the version of a dynamically linked *App1.shared* executable. If we load its core dump *App1.shared.core.22442* in the new instance of WinDbg Preview, we see the list of shared libraries:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App1\App1.shared.core.22442]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
.....
*** WARNING: Unable to verify timestamp for libc-2.17.so
libc_2_17!nanosleep+0x24:
0000ffff`0496dd64 d4000001 svc         #0

0:000> .sympath+ C:\ALCDA2\A64\App1
*** WARNING: Unable to verify timestamp for libc-2.17.so
Symbol search path is: srv*;C:\ALCDA2\A64\App1
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app1

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App1

0:000> .reload
....*** WARNING: Unable to verify timestamp for libc-2.17.so
.

************* Symbol Loading Error Summary **************
Module name            Error
libc-2.17              The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

```
0:000> lm
start             end                 module name
00000000`00400000 00000000`00430000   App1       (service symbols: ELF Export Symbols)
c:\alcda2\a64\app1\App1.shared
0000ffff`048c0000 0000ffff`04a50000   libc_2_17 T (service symbols: ELF In Memory Symbols)
0000ffff`04a50000 0000ffff`04a90000   libpthread_2_17   (deferred)
0000ffff`04ab0000 0000ffff`04ac0000   linux_vdso_so   (deferred)
0000ffff`04ac0000 0000ffff`04b00000   ld_2_17    (deferred)
```

16. Disassemble the *bar_one* function and follow the indirect *sleep* function call:

```
0:000> uf bar_one
Couldn't resolve error at 'bar_one'
```

It looks like we need to dump the stack trace to have symbols fully loaded:

```
0:000> k
*** WARNING: Unable to verify timestamp for App1.shared
*** WARNING: Unable to verify timestamp for libpthread-2.17.so
 # Child-SP          RetAddr               Call Site
00 0000ffff`048be750 0000ffff`0496da20     libc_2_17!nanosleep+0x24
01 0000ffff`048be790 00000000`00400738     libc_2_17!sleep+0x11c
02 0000ffff`048be990 00000000`0040074c     App1!bar_one+0x10
03 0000ffff`048be9a0 00000000`00400764     App1!foo_one+0xc
04 0000ffff`048be9b0 0000ffff`04a57d40     App1!thread_one+0x10
05 0000ffff`048be9d0 0000ffff`049a2d00     libpthread_2_17!_pthread_get_minstack+0x1394
06 0000ffff`048beb00 ffffffff`ffffffff     libc_2_17!clone+0x80
07 0000ffff`048beb00 00000000`00000000     0xffffffff`ffffffff

0:000> uf bar_one
App1!bar_one:
00000000`00400728 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`0040072c 910003fd mov         fp,sp
00000000`00400730 12800000 mov         w0,#-1
00000000`00400734 97ffff93 bl          App1!$x+0x30 (00000000`00400580)
00000000`00400738 a8c17bfd ldp         fp,lr,[sp],#0x10
00000000`0040073c d65f03c0 ret

0:000> u 00000000`00400580
App1!$x+0x30:
00000000`00400580 90000110 adrp        xip0,App1!+0x18 (00000000`00420000)
00000000`00400584 f9400611 ldr         xip1,[xip0,#8]
00000000`00400588 91002210 add         xip0,xip0,#8
00000000`0040058c d61f0220 br          xip1
00000000`00400590 90000110 adrp        xip0,App1!+0x18 (00000000`00420000)
00000000`00400594 f9400a11 ldr         xip1,[xip0,#0x10]
00000000`00400598 91004210 add         xip0,xip0,#0x10
00000000`0040059c d61f0220 br          xip1
```

Note: XIP0/XIP1 are mnemonics for X16/X17 registers used for inter-procedure-call.

```
0:000> dp 00000000`00420000 + 8
00000000`00420008  0000ffff`0496d904 0000ffff`04a57fd0
00000000`00420018  00000000`00400550 00000000`00400550
00000000`00420028  00000000`00000000 00000000`00000000
00000000`00420038  00000000`00000000 00000000`00000000
00000000`00420048  00000000`00000000 00000000`00000000
00000000`00420058  00000000`00000000 00000000`00000000
```

```
00000000`00420068  00000000`00000000 00000000`00000000
00000000`00420078  00000000`00000000 00000000`00000000
```

```
0:000> u 0000ffff`0496d904
libc_2_17!sleep:
0000ffff`0496d904 d106c3ff sub         sp,sp,#0x1B0
0000ffff`0496d908 a9bb7bfd stp         fp,lr,[sp,#-0x50]!
0000ffff`0496d90c 910003fd mov         fp,sp
0000ffff`0496d910 a90153f3 stp         x19,x20,[sp,#0x10]
0000ffff`0496d914 a9025bf5 stp         x21,x22,[sp,#0x20]
0000ffff`0496d918 a90363f7 stp         x23,x24,[sp,#0x30]
0000ffff`0496d91c f90023f9 str         x25,[sp,#0x40]
0000ffff`0496d920 34000e40 cbz         w0,libc_2_17!sleep+0x1e4 (0000ffff`0496dae8)
```

```
0:000> ln 0000ffff`0496d904
Browse module
Set bu breakpoint

(0000ffff`0496d904)   libc_2_17!sleep
Exact matches:
    libc_2_17!sleep = <no type information>
```

```
0:000> dps 00000000`00420000 + 8
00000000`00420008  0000ffff`0496d904 libc_2_17!sleep
00000000`00420010  0000ffff`04a57fd0 libpthread_2_17!pthread_create
00000000`00420018  00000000`00400550 App1!$x
00000000`00420020  00000000`00400550 App1!$x
00000000`00420028  00000000`00000000
00000000`00420030  00000000`00000000
00000000`00420038  00000000`00000000
00000000`00420040  00000000`00000000
00000000`00420048  00000000`00000000
00000000`00420050  00000000`00000000
00000000`00420058  00000000`00000000
00000000`00420060  00000000`00000000
00000000`00420068  00000000`00000000
00000000`00420070  00000000`00000000
```

17. *App1.shared.pmap.22442* also shows library memory regions:

```
22442:   ./App1.shared
0000000000400000     64K r-x-- App1.shared
0000000000410000     64K r---- App1.shared
0000000000420000     64K rw--- App1.shared
0000000036a80000    192K rw---   [ anon ]
0000ffff02070000     64K -----   [ anon ]
0000ffff02080000   8192K rw---   [ anon ]
0000ffff02880000     64K -----   [ anon ]
0000ffff02890000   8192K rw---   [ anon ]
0000ffff03090000     64K -----   [ anon ]
0000ffff030a0000   8192K rw---   [ anon ]
0000ffff038a0000     64K -----   [ anon ]
0000ffff038b0000   8192K rw---   [ anon ]
0000ffff040b0000     64K -----   [ anon ]
0000ffff040c0000   8192K rw---   [ anon ]
0000ffff048c0000   1472K r-x-- libc-2.17.so
0000ffff04a30000     64K r---- libc-2.17.so
0000ffff04a40000     64K rw--- libc-2.17.so
0000ffff04a50000    128K r-x-- libpthread-2.17.so
0000ffff04a70000     64K r---- libpthread-2.17.so
```

```
0000ffff04a80000     64K rw--- libpthread-2.17.so
0000ffff04aa0000     64K r----   [ anon ]
0000ffff04ab0000     64K r-x--   [ anon ]
0000ffff04ac0000    128K r-x-- ld-2.17.so
0000ffff04ae0000     64K r---- ld-2.17.so
0000ffff04af0000     64K rw--- ld-2.17.so
0000ffffe2fc0000    192K rw---   [ stack ]
 total            44096K
```

Note: We can also see shared library mappings in the output of the **!address** command:

```
0:000> !address

Mapping file section regions...
Mapping module regions...

        BaseAddress      EndAddress+1        RegionSize     Type       State                 Protect             Usage
----------------------------------------------------------------------------------------------------------------------
+        0`00000000        0`00400000        0`00400000                                                       <unknown>
+        0`00400000        0`00410000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ           Image      [App1;
"/home/opc/ALCDA2/App1/App1.shared"]
+        0`00410000        0`00420000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY               Image      [App1;
"/home/opc/ALCDA2/App1/App1.shared"]
+        0`00420000        0`00430000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            Image      [App1;
"/home/opc/ALCDA2/App1/App1.shared"]
+        0`00430000        0`36a80000        0`36650000                                                       <unknown>
+        0`36a80000        0`36ab0000        0`00030000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            <unknown>  [........Q.......]
+        0`36ab0000     ffff`02070000     fffe`cb5c0000                                                       <unknown>
+     ffff`02070000     ffff`02080000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY             <unknown>  [................]
+     ffff`02080000     ffff`02880000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            <unknown>  [................]
+     ffff`02880000     ffff`02890000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY             <unknown>  [................]
+     ffff`02890000     ffff`03090000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            <unknown>  [................]
+     ffff`03090000     ffff`030a0000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY             <unknown>  [................]
+     ffff`030a0000     ffff`038a0000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            <unknown>  [................]
+     ffff`038a0000     ffff`038b0000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY             <unknown>  [................]
+     ffff`038b0000     ffff`040b0000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            <unknown>  [................]
+     ffff`040b0000     ffff`040c0000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY             <unknown>  [................]
+     ffff`040c0000     ffff`048c0000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            <unknown>  [................]
+     ffff`048c0000     ffff`04a30000        0`00170000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ           Image      [libc_2_17; "/usr/lib64/libc-
2.17.so"]
+     ffff`04a30000     ffff`04a40000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY             Image      [libc_2_17; "/usr/lib64/libc-
2.17.so"]
+     ffff`04a40000     ffff`04a50000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            Image      [libc_2_17; "/usr/lib64/libc-
2.17.so"]
+     ffff`04a50000     ffff`04a70000        0`00020000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ           Image      [libpthread_2_17;
"/usr/lib64/libpthread-2.17.so"]
+     ffff`04a70000     ffff`04a80000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY             Image      [libpthread_2_17;
"/usr/lib64/libpthread-2.17.so"]
+     ffff`04a80000     ffff`04a90000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            Image      [libpthread_2_17;
"/usr/lib64/libpthread-2.17.so"]
+     ffff`04a90000     ffff`04ab0000        0`00020000                                                       <unknown>
+     ffff`04ab0000     ffff`04ac0000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ           Image      [linux_vdso_so; "linux-
vdso.so.1"]
+     ffff`04ac0000     ffff`04ae0000        0`00020000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ           Image      [ld_2_17; "/usr/lib64/ld-
2.17.so"]
+     ffff`04ae0000     ffff`04af0000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY             Image      [ld_2_17; "/usr/lib64/ld-
2.17.so"]
+     ffff`04af0000     ffff`04b00000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            Image      [ld_2_17; "/usr/lib64/ld-
2.17.so"]
+     ffff`04b00000     ffff`e2fc0000        0`de4c0000                                                       <unknown>
+     ffff`e2fc0000     ffff`e2ff0000        0`00030000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE            <unknown>  [................]
```

18. We close logging before exiting WinDbg Preview:

```
0:000> .logclose
Closing open log file 'C:\ALCDA2\A64\App1\App1.log'
```

We recommend exiting WinDbg Preview app or WinDbg after each exercise to avoid glitches.

Exercise A2D

- **Goal:** Learn how to identify exceptions, find problem threads and CPU instructions
- **Patterns:** NULL Pointer (Data); Active Thread (x64, GDB)
- \ALCDA-Dumps\Exercise-A2D-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A2D-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A2D-A64-WinDbg.pdf

Exercise A2D (x64, GDB)

Goal: Learn how to identify exceptions, find problem threads and CPU instructions.

Patterns: NULL Pointer (Data); Active Thread.

1. Load *core.App2D* dump file and *App2D* executable from the x64/App2D directory:

```
~/ALCDA2/x64/App2D$ gdb -c core.App2D -se App2D
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App2D...done.
[New LWP 3577]
[New LWP 3575]
[New LWP 3576]
[New LWP 3579]
[New LWP 3578]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App2D'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x0000000000401bbd in procA ()
[Current thread is 1 (Thread 0x7faf71659700 (LWP 3577))]
```

2. List all threads:

```
(gdb) info threads
  Id   Target Id                         Frame
* 1    Thread 0x7faf71659700 (LWP 3577) 0x0000000000401bbd in procA ()
  2    Thread 0xb97880 (LWP 3575)       0x00000000004442a1 in clone ()
  3    Thread 0x7faf71e5a700 (LWP 3576) 0x0000000000441a50 in nanosleep ()
  4    Thread 0x7faf70657700 (LWP 3579) 0x00000000004442a1 in clone ()
  5    Thread 0x7faf70e58700 (LWP 3578) 0x0000000000441a50 in nanosleep ()
```

3. The problem thread seems to be the current thread:

```
(gdb) thread 1
[Switching to thread 1 (Thread 0x7faf71659700 (LWP 3577))]
#0  0x0000000000401bbd in procA ()
```

```
(gdb) bt
#0  0x0000000000401bbd in procA ()
#1  0x0000000000401c3b in bar_two ()
#2  0x0000000000401c4c in foo_two ()
#3  0x0000000000401c65 in thread_two ()
#4  0x0000000000403113 in start_thread (arg=<optimized out>) at pthread_create.c:486
#5  0x00000000004442af in clone ()
```

4. Disassemble the problem instruction and check CPU register(s) details (NULL data pointer):

```
(gdb) x/i 0x0000000000401bbd
=> 0x401bbd <procA+16>: movl   $0x1,(%rax)

(gdb) info r $rax
rax            0x0  0

(gdb) x $rax
   0x0: Cannot access memory at address 0x0
```

5. List all thread stack traces and identify other anomalies, such as non-waiting active threads:

```
(gdb) thread apply all bt

Thread 5 (Thread 0x7faf70e58700 (LWP 3578)):
#0  0x0000000000441a50 in nanosleep ()
#1  0x00000000004419da in sleep ()
#2  0x0000000000401c7a in bar_three () at pthread_create.c:688
#3  0x0000000000401c8b in foo_three () at pthread_create.c:688
#4  0x0000000000401ca4 in thread_three () at pthread_create.c:688
#5  0x0000000000403113 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x00000000004442af in clone ()

Thread 4 (Thread 0x7faf70657700 (LWP 3579)):
#0  0x00000000004442a1 in clone ()
#1  0x0000000000403020 in ?? () at pthread_create.c:362
#2  0x00007faf70657700 in ?? ()
#3  0x0000000000000000 in ?? ()

Thread 3 (Thread 0x7faf71e5a700 (LWP 3576)):
#0  0x0000000000441a50 in nanosleep ()
#1  0x00000000004419da in sleep ()
#2  0x0000000000401bfc in bar_one () at pthread_create.c:688
#3  0x0000000000401c0d in foo_one () at pthread_create.c:688
#4  0x0000000000401c26 in thread_one () at pthread_create.c:688
#5  0x0000000000403113 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x00000000004442af in clone ()

Thread 2 (Thread 0xb97880 (LWP 3575)):
#0  0x00000000004442a1 in clone ()
#1  0x0000000000401f4f in create_thread (pd=pd@entry=0x7faf70657700,
attr=attr@entry=0x7fffc8d6bcf0, stopped_start=stopped_start@entry=0x7fffc8d6bcee,
stackaddr=stackaddr@entry=0x7faf70656e80,
    thread_ran=thread_ran@entry=0x7fffc8d6bcef) at
../sysdeps/unix/sysv/linux/createthread.c:101
#2  0x0000000000403986 in __pthread_create_2_1 (newthread=<optimized out>, attr=<optimized
out>, start_routine=<optimized out>, arg=<optimized out>) at pthread_create.c:826
#3  0x0000000000401dac in main () at pthread_create.c:688

Thread 1 (Thread 0x7faf71659700 (LWP 3577)):
#0  0x0000000000401bbd in procA () at pthread_create.c:688
```

```
#1  0x0000000000401c3b in bar_two () at pthread_create.c:688
#2  0x0000000000401c4c in foo_two () at pthread_create.c:688
#3  0x0000000000401c65 in thread_two () at pthread_create.c:688
#4  0x0000000000403113 in start_thread (arg=<optimized out>) at pthread_create.c:486
#5  0x00000000004442af in clone ()
```

6. Check the CPU instruction and the stack pointer of thread #4 for any signs of stack overflow (unaccessible stack addresses below the current stack pointer):

```
(gdb) thread 4
[Switching to thread 4 (Thread 0x7faf70657700 (LWP 3579))]
#0  0x00000000004442a1 in clone ()

(gdb) bt
#0  0x00000000004442a1 in clone ()
#1  0x0000000000403020 in ?? () at pthread_create.c:362
#2  0x00007faf70657700 in ?? ()
#3  0x0000000000000000 in ?? ()

(gdb) x/i 0x00000000004442a1
=> 0x4442a1 <clone+49>: test   %rax,%rax

(gdb) x/gx $rsp
0x7faf70656e70: 0x0000000000403020

(gdb) x/gx $rsp-8
0x7faf70656e68: 0x0000000000000000

(gdb) x/gx $rsp-0x10
0x7faf70656e60: 0x0000000000000000
```

7. Switch to thread #2 and verify that the *main* function was being engaged in thread creation (this may correlate with the last thread #4 caught in being created):

```
(gdb) thread 2
[Switching to thread 2 (Thread 0xb97880 (LWP 3575))]
#0  0x00000000004442a1 in clone ()

(gdb) bt
#0  0x00000000004442a1 in clone ()
#1  0x0000000000401f4f in create_thread (pd=pd@entry=0x7faf70657700,
attr=attr@entry=0x7fffc8d6bcf0, stopped_start=stopped_start@entry=0x7fffc8d6bcee,
stackaddr=stackaddr@entry=0x7faf70656e80,
    thread_ran=thread_ran@entry=0x7fffc8d6bcef) at
../sysdeps/unix/sysv/linux/createthread.c:101
#2  0x0000000000403986 in __pthread_create_2_1 (newthread=<optimized out>, attr=<optimized
out>, start_routine=<optimized out>, arg=<optimized out>) at pthread_create.c:826
#3  0x0000000000401dac in main () at pthread_create.c:688

(gdb) disassemble main
Dump of assembler code for function main:
   0x0000000000401d29 <+0>:     push   %rbp
   0x0000000000401d2a <+1>:     mov    %rsp,%rbp
   0x0000000000401d2d <+4>:     sub    $0x40,%rsp
   0x0000000000401d31 <+8>:     mov    %edi,-0x34(%rbp)
   0x0000000000401d34 <+11>:    mov    %rsi,-0x40(%rbp)
   0x0000000000401d38 <+15>:    lea    -0x8(%rbp),%rax
   0x0000000000401d3c <+19>:    mov    $0x0,%ecx
   0x0000000000401d41 <+24>:    lea    -0x138(%rip),%rdx        # 0x401c10 <thread_one>
```

```
   0x0000000000401d48 <+31>:    mov    $0x0,%esi
   0x0000000000401d4d <+36>:    mov    %rax,%rdi
   0x0000000000401d50 <+39>:    callq  0x403400 <__pthread_create_2_1>
   0x0000000000401d55 <+44>:    lea    -0x10(%rbp),%rax
   0x0000000000401d59 <+48>:    mov    $0x0,%ecx
   0x0000000000401d5e <+53>:    lea    -0x116(%rip),%rdx        # 0x401c4f <thread_two>
   0x0000000000401d65 <+60>:    mov    $0x0,%esi
   0x0000000000401d6a <+65>:    mov    %rax,%rdi
   0x0000000000401d6d <+68>:    callq  0x403400 <__pthread_create_2_1>
   0x0000000000401d72 <+73>:    lea    -0x18(%rbp),%rax
   0x0000000000401d76 <+77>:    mov    $0x0,%ecx
   0x0000000000401d7b <+82>:    lea    -0xf4(%rip),%rdx        # 0x401c8e <thread_three>
   0x0000000000401d82 <+89>:    mov    $0x0,%esi
   0x0000000000401d87 <+94>:    mov    %rax,%rdi
   0x0000000000401d8a <+97>:    callq  0x403400 <__pthread_create_2_1>
   0x0000000000401d8f <+102>:   lea    -0x20(%rbp),%rax
   0x0000000000401d93 <+106>:   mov    $0x0,%ecx
   0x0000000000401d98 <+111>:   lea    -0xd2(%rip),%rdx        # 0x401ccd <thread_four>
   0x0000000000401d9f <+118>:   mov    $0x0,%esi
   0x0000000000401da4 <+123>:   mov    %rax,%rdi
   0x0000000000401da7 <+126>:   callq  0x403400 <__pthread_create_2_1>
   0x0000000000401dac <+131>:   lea    -0x28(%rbp),%rax
   0x0000000000401db0 <+135>:   mov    $0x0,%ecx
   0x0000000000401db5 <+140>:   lea    -0xb0(%rip),%rdx        # 0x401d0c <thread_five>
   0x0000000000401dbc <+147>:   mov    $0x0,%esi
   0x0000000000401dc1 <+152>:   mov    %rax,%rdi
   0x0000000000401dc4 <+155>:   callq  0x403400 <__pthread_create_2_1>
   0x0000000000401dc9 <+160>:   mov    $0x3,%edi
   0x0000000000401dce <+165>:   callq  0x4419a0 <sleep>
   0x0000000000401dd3 <+170>:   mov    $0x0,%eax
   0x0000000000401dd8 <+175>:   leaveq
   0x0000000000401dd9 <+176>:   retq
End of assembler dump.
```

Exercise A2D (A64, GDB)

Goal: Learn how to identify exceptions, find problem threads and CPU instructions.

Patterns: NULL Pointer (Data).

1. Load *core.14554* dump file and *App2D* executable from the A64/App2D directory:

```
~/ALCDA2/A64/App2D$ gdb -c core.14554 -se App2D
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App2D...
(No debugging symbols found in App2D)

warning: Can't open file /home/opc/ALCDA2/App2D/App2D during file-backed mapping note
processing
[New LWP 14556]
[New LWP 14554]
[New LWP 14559]
[New LWP 14557]
[New LWP 14555]
[New LWP 14558]
Core was generated by `./App2D'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x00000000004031f8 in procA ()
[Current thread is 1 (LWP 14556)]
```

2. Set logging to a file in case of lengthy output from some commands and set color highlighting off:

```
(gdb) set logging file App2D.log

(gdb) set logging enabled on
Copying output to App2D.log.
Copying debug output to App2D.log.

(gdb) set style enabled off
```

3. List all threads:

```
(gdb) info threads
  Id   Target Id         Frame
* 1    LWP 14556         0x00000000004031f8 in procA ()
  2    LWP 14554         0x000000000040c9f4 in nanosleep ()
  3    LWP 14559         0x000000000040c9f4 in nanosleep ()
```

```
  4    LWP 14557         0x000000000040c9f4 in nanosleep ()
  5    LWP 14555         0x000000000040c9f4 in nanosleep ()
  6    LWP 14558         0x000000000040c9f4 in nanosleep ()
```

4. The problem thread seems to be the current thread:

```
(gdb) thread 1
[Switching to thread 1 (LWP 14556)]
#0  0x00000000004031f8 in procA ()

(gdb) bt
#0  0x00000000004031f8 in procA ()
#1  0x000000000040327c in bar_two ()
#2  0x0000000000403290 in foo_two ()
#3  0x00000000004032a8 in thread_two ()
#4  0x0000000000404c74 in start_thread ()
#5  0x0000000000429ba0 in thread_start ()
```

5. Disassemble the problem instruction and check CPU register(s) details (NULL data pointer):

```
(gdb) x/i 0x00000000004031f8
=> 0x4031f8 <procA+16>: str     w1, [x0]

(gdb) info r x0
x0             0x0                 0

(gdb) x $x0
   0x0: Cannot access memory at address 0x0
```

6. List all thread stack traces to see any other possible anomalies, such as non-waiting active threads:

```
(gdb) thread apply all bt

Thread 6 (LWP 14558):
#0  0x000000000040c9f4 in nanosleep ()
#1  0x0000000000424cf4 in sleep ()
#2  0x0000000000403214 in proc ()
#3  0x0000000000403308 in bar_four ()
#4  0x000000000040331c in foo_four ()
#5  0x0000000000403334 in thread_four ()
#6  0x0000000000404c74 in start_thread ()
#7  0x0000000000429ba0 in thread_start ()

Thread 5 (LWP 14555):
#0  0x000000000040c9f4 in nanosleep ()
#1  0x0000000000424cf4 in sleep ()
#2  0x0000000000403238 in bar_one ()
#3  0x000000000040324c in foo_one ()
#4  0x0000000000403264 in thread_one ()
#5  0x0000000000404c74 in start_thread ()
#6  0x0000000000429ba0 in thread_start ()

Thread 4 (LWP 14557):
#0  0x000000000040c9f4 in nanosleep ()
#1  0x0000000000424cf4 in sleep ()
#2  0x00000000004032c4 in bar_three ()
#3  0x00000000004032d8 in foo_three ()
#4  0x00000000004032f0 in thread_three ()
#5  0x0000000000404c74 in start_thread ()
#6  0x0000000000429ba0 in thread_start ()
```

```
Thread 3 (LWP 14559):
#0  0x000000000040c9f4 in nanosleep ()
#1  0x0000000000424cf4 in sleep ()
#2  0x0000000000403350 in bar_five ()
#3  0x0000000000403364 in foo_five ()
#4  0x000000000040337c in thread_five ()
#5  0x0000000000404c74 in start_thread ()
#6  0x0000000000429ba0 in thread_start ()

Thread 2 (LWP 14554):
#0  0x000000000040c9f4 in nanosleep ()
#1  0x0000000000424cf4 in sleep ()
#2  0x0000000000403418 in main ()

Thread 1 (LWP 14556):
#0  0x00000000004031f8 in procA ()
#1  0x000000000040327c in bar_two ()
#2  0x0000000000403290 in foo_two ()
#3  0x00000000004032a8 in thread_two ()
#4  0x0000000000404c74 in start_thread ()
#5  0x0000000000429ba0 in thread_start ()
```

Exercise A2D (A64, WinDbg Preview)

Goal: Learn how to identify exceptions, find problem threads and CPU instructions.

Patterns: NULL Pointer (Data).

1. Launch WinDbg Preview.

2. Load *core.14554* dump file from the A64\App2D folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App2D\core.14554]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
(38da.38dc): Signal SIGSEGV (Segmentation fault) code SEGV_MAPERR (Address not mapped to
object) at 0x0*** WARNING: Unable to verify timestamp for App2D
App2D+0x31f8:
00000000`004031f8 b9000001 str         w1,[x0]
```

3. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App2D\App2D.log
Opened log file 'C:\ALCDA2\A64\App2D\App2D.log'
```

4. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App2D\
Symbol search path is: srv*;C:\ALCDA2\A64\App2D\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app2d\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App2D\
*** WARNING: Unable to verify timestamp for App2D

0:000> .reload
..
*** WARNING: Unable to verify timestamp for App2D

************* Symbol Loading Error Summary **************
Module name            Error
```

```
App2D                  The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

5. The problem thread seems to be the current thread:

```
0:000> k
 # Child-SP          RetAddr               Call Site
00 0000fffe`1eeee810 00000000`0040327c     App2D!procA+0x10
01 0000fffe`1eeee820 00000000`00403290     App2D!bar_two+0xc
02 0000fffe`1eeee830 00000000`004032a8     App2D!foo_two+0xc
03 0000fffe`1eeee840 00000000`00404c74     App2D!thread_two+0x10
04 0000fffe`1eeee860 00000000`00429ba0     App2D!start_thread+0xb4
05 0000fffe`1eeee990 ffffffff`ffffffff     App2D!thread_start+0x30
06 0000fffe`1eeee990 00000000`00000000     0xffffffff`ffffffff
```

6. Check the problem instruction and CPU register(s) details (NULL data pointer):

```
0:000> r
 x0=0000000000000000   x1=0000000000000001   x2=0000fffe1eeef080   x3=3a2398bf2f00aa18
 x4=0000fffe1eeee860   x5=3a23674131ee4278   x6=0000fffe1eeef150   x7=0000000000000000
 x8=0000000000000063   x9=0000000000800000  x10=0000000000404bc0  x11=00000000003d0f00
x12=0000fffe1eeef080  x13=0000000000000000  x14=0000000000000000  x15=0000000000000000
x16=00000000004d0010  x17=0000000000424000  x18=0000000000000110  x19=0000fffe1eeef080
x20=0000000000000000  x21=00000000004d0000  x22=0000000000403298  x23=0000000000000000
x24=0000fffe1eeef770  x25=00000000301b06f0  x26=00000000004d7890  x27=0000000000010000
x28=0000000000810000   fp=0000fffe1eeee820   lr=000000000040327c   sp=0000fffe1eeee810
 pc=00000000004031f8  psr=20001000 --C- EL0
App2D!procA+0x10:
00000000`004031f8 b9000001 str         w1,[x0]
```

7. List all thread stack traces and check if there are other anomalies, such as non-waiting active threads:

```
0:000> ~*k

Unable to get thread data for thread 0
.  0  Id: 38da.38dc Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`1eeee810 00000000`0040327c     App2D!procA+0x10
01 0000fffe`1eeee820 00000000`00403290     App2D!bar_two+0xc
02 0000fffe`1eeee830 00000000`004032a8     App2D!foo_two+0xc
03 0000fffe`1eeee840 00000000`00404c74     App2D!thread_two+0x10
04 0000fffe`1eeee860 00000000`00429ba0     App2D!start_thread+0xb4
05 0000fffe`1eeee990 ffffffff`ffffffff     App2D!thread_start+0x30
06 0000fffe`1eeee990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 1
   1  Id: 38da.38da Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`dbdf3ec0 00000000`00424cf4     App2D!_libc_nanosleep+0x24
01 0000ffff`dbdf3f00 00000000`00403418     App2D!sleep+0x110
02 0000ffff`dbdf40f0 00000000`0040ec8c     App2D!main+0x90
03 0000ffff`dbdf4140 00000000`00403090     App2D!_libc_start_main+0x304
04 0000ffff`dbdf42a0 00000000`00000000     App2D!start+0x4c
```

```
Unable to get thread data for thread 2
   2  Id: 38da.38df Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`1d6be5f0 00000000`00424cf4     App2D!_libc_nanosleep+0x24
01 0000fffe`1d6be630 00000000`00403350     App2D!sleep+0x110
02 0000fffe`1d6be820 00000000`00403364     App2D!bar_five+0x10
03 0000fffe`1d6be830 00000000`0040337c     App2D!foo_five+0xc
04 0000fffe`1d6be840 00000000`00404c74     App2D!thread_five+0x10
05 0000fffe`1d6be860 00000000`00429ba0     App2D!start_thread+0xb4
06 0000fffe`1d6be990 ffffffff`ffffffff     App2D!thread_start+0x30
07 0000fffe`1d6be990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 3
   3  Id: 38da.38dd Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`1e6de5f0 00000000`00424cf4     App2D!_libc_nanosleep+0x24
01 0000fffe`1e6de630 00000000`004032c4     App2D!sleep+0x110
02 0000fffe`1e6de820 00000000`004032d8     App2D!bar_three+0x10
03 0000fffe`1e6de830 00000000`004032f0     App2D!foo_three+0xc
04 0000fffe`1e6de840 00000000`00404c74     App2D!thread_three+0x10
05 0000fffe`1e6de860 00000000`00429ba0     App2D!start_thread+0xb4
06 0000fffe`1e6de990 ffffffff`ffffffff     App2D!thread_start+0x30
07 0000fffe`1e6de990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 4
   4  Id: 38da.38db Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`1f6fe5f0 00000000`00424cf4     App2D!_libc_nanosleep+0x24
01 0000fffe`1f6fe630 00000000`00403238     App2D!sleep+0x110
02 0000fffe`1f6fe820 00000000`0040324c     App2D!bar_one+0x10
03 0000fffe`1f6fe830 00000000`00403264     App2D!foo_one+0xc
04 0000fffe`1f6fe840 00000000`00404c74     App2D!thread_one+0x10
05 0000fffe`1f6fe860 00000000`00429ba0     App2D!start_thread+0xb4
06 0000fffe`1f6fe990 ffffffff`ffffffff     App2D!thread_start+0x30
07 0000fffe`1f6fe990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 5
   5  Id: 38da.38de Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`1dece5d0 00000000`00424cf4     App2D!_libc_nanosleep+0x24
01 0000fffe`1dece610 00000000`00403214     App2D!sleep+0x110
02 0000fffe`1dece800 00000000`00403308     App2D!procB+0x10
03 0000fffe`1dece820 00000000`0040331c     App2D!bar_four+0xc
04 0000fffe`1dece830 00000000`00403334     App2D!foo_four+0xc
05 0000fffe`1dece840 00000000`00404c74     App2D!thread_four+0x10
06 0000fffe`1dece860 00000000`00429ba0     App2D!start_thread+0xb4
07 0000fffe`1dece990 ffffffff`ffffffff     App2D!thread_start+0x30
08 0000fffe`1dece990 00000000`00000000     0xffffffff`ffffffff
```

Note: There are no other active threads.

8. We close logging before exiting WinDbg Preview:

```
0:000> .logclose
Closing open log file 'C:\ALCDA2\A64\App2D\App2D.log'
```

Exercise A2C

- **Goal:** Learn how to identify exceptions, find problem threads and CPU instructions
- **Patterns:** NULL Pointer (Code); Missing Frame (WinDbg)
- \ALCDA-Dumps\Exercise-A2C-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A2C-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A2C-A64-WinDbg.pdf

Exercise A2C (x64, GDB)

Goal: Learn how to identify exceptions, find problem threads and CPU instructions.

Patterns: NULL Pointer (Code).

1. Load *core.App2C* dump file and *App2C* executable from the x64/App2C directory:

```
~/ALCDA2/x64/App2C$ gdb -c core.App2C -se App2C
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App2C...done.
[New LWP 3651]
[New LWP 3647]
[New LWP 3648]
[New LWP 3650]
[New LWP 3649]
[New LWP 3652]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App2C'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x0000000000000000 in ?? ()
[Current thread is 1 (Thread 0x7f7d259c4700 (LWP 3651))]
```

2. List all threads:

```
(gdb) info threads
  Id   Target Id                         Frame
* 1    Thread 0x7f7d259c4700 (LWP 3651) 0x0000000000000000 in ?? ()
  2    Thread 0x1bd4880 (LWP 3647)      0x0000000000441a60 in nanosleep ()
  3    Thread 0x7f7d271c7700 (LWP 3648) 0x0000000000441a60 in nanosleep ()
  4    Thread 0x7f7d261c5700 (LWP 3650) 0x0000000000441a60 in nanosleep ()
  5    Thread 0x7f7d269c6700 (LWP 3649) 0x0000000000441a60 in nanosleep ()
  6    Thread 0x7f7d251c3700 (LWP 3652) 0x0000000000441a60 in nanosleep ()
```

3. The problem thread seems to be the current thread:

```
(gdb) bt
#0  0x0000000000000000 in ?? ()
#1  0x0000000000401bf9 in proc ()
#2  0x0000000000401cc7 in bar_four ()
#3  0x0000000000401cd8 in foo_four ()
#4  0x0000000000401cf1 in thread_four ()
```

```
#5  0x0000000000403123 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x00000000004442bf in clone ()
```

Note: It looks like our GDB version prints the non-existent *proc* function instead of *procB*.

```
(gdb) disassemble proc
No symbol "proc" in current context.
```

4. Check the CPU instruction and a dereferenced pointer for any signs of a NULL pointer:

```
(gdb) disassemble procB
Dump of assembler code for function procB:
   0x0000000000401bd4 <+0>:     push   %rbp
   0x0000000000401bd5 <+1>:     mov    %rsp,%rbp
   0x0000000000401bd8 <+4>:     sub    $0x10,%rsp
   0x0000000000401bdc <+8>:     mov    $0x1,%edi
   0x0000000000401be1 <+13>:    callq  0x4419b0 <sleep>
   0x0000000000401be6 <+18>:    movq   $0x0,-0x8(%rbp)
   0x0000000000401bee <+26>:    mov    -0x8(%rbp),%rdx
   0x0000000000401bf2 <+30>:    mov    $0x0,%eax
   0x0000000000401bf7 <+35>:    callq  *%rdx
   0x0000000000401bf9 <+37>:    nop
   0x0000000000401bfa <+38>:    leaveq
   0x0000000000401bfb <+39>:    retq
End of assembler dump.

(gdb) info r rdx
rdx            0x0  0
```

5. List all thread stack traces to check for other anomalies, such as non-waiting active threads:

```
(gdb) thread apply all bt

Thread 6 (Thread 0x7f7d251c3700 (LWP 3652)):
#0  0x0000000000441a60 in nanosleep ()
#1  0x00000000004419ea in sleep ()
#2  0x0000000000401d06 in bar_five () at pthread_create.c:688
#3  0x0000000000401d17 in foo_five () at pthread_create.c:688
#4  0x0000000000401d30 in thread_five () at pthread_create.c:688
#5  0x0000000000403123 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x00000000004442bf in clone ()

Thread 5 (Thread 0x7f7d269c6700 (LWP 3649)):
#0  0x0000000000441a60 in nanosleep ()
#1  0x00000000004419ea in sleep ()
#2  0x0000000000401bbf in procA () at pthread_create.c:688
#3  0x0000000000401c49 in bar_two () at pthread_create.c:688
#4  0x0000000000401c5a in foo_two () at pthread_create.c:688
#5  0x0000000000401c73 in thread_two () at pthread_create.c:688
#6  0x0000000000403123 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x00000000004442bf in clone ()

Thread 4 (Thread 0x7f7d261c5700 (LWP 3650)):
#0  0x0000000000441a60 in nanosleep ()
#1  0x00000000004419ea in sleep ()
#2  0x0000000000401c88 in bar_three () at pthread_create.c:688
#3  0x0000000000401c99 in foo_three () at pthread_create.c:688
#4  0x0000000000401cb2 in thread_three () at pthread_create.c:688
#5  0x0000000000403123 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x00000000004442bf in clone ()
```

```
Thread 3 (Thread 0x7f7d271c7700 (LWP 3648)):
#0  0x0000000000441a60 in nanosleep ()
#1  0x00000000004419ea in sleep ()
#2  0x0000000000401c0a in bar_one () at pthread_create.c:688
#3  0x0000000000401c1b in foo_one () at pthread_create.c:688
#4  0x0000000000401c34 in thread_one () at pthread_create.c:688
#5  0x0000000000403123 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x00000000004442bf in clone ()

Thread 2 (Thread 0x1bd4880 (LWP 3647)):
#0  0x0000000000441a60 in nanosleep ()
#1  0x00000000004419ea in sleep ()
#2  0x0000000000401de1 in main () at pthread_create.c:688

Thread 1 (Thread 0x7f7d259c4700 (LWP 3651)):
#0  0x0000000000000000 in ?? ()
#1  0x0000000000401bf9 in proc () at pthread_create.c:688
#2  0x0000000000401cc7 in bar_four () at pthread_create.c:688
#3  0x0000000000401cd8 in foo_four () at pthread_create.c:688
#4  0x0000000000401cf1 in thread_four () at pthread_create.c:688
#5  0x0000000000403123 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x00000000004442bf in clone ()
```

Exercise A2C (A64, GDB)

Goal: Learn how to identify exceptions, find problem threads and CPU instructions.

Patterns: NULL Pointer (Code).

1. Load *core.24559* dump file and *App2C* executable from the A64/App2C directory:

```
~/ALCDA2/A64/App2C$ gdb -c core.24559 -se App2C
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App2C...
(No debugging symbols found in App2C)

warning: Can't open file /home/opc/ALCDA2/App2C/App2C during file-backed mapping note
processing
[New LWP 24563]
[New LWP 24559]
[New LWP 24560]
[New LWP 24561]
[New LWP 24564]
[New LWP 24562]
Core was generated by `./App2C'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x0000000000000000 in ?? ()
[Current thread is 1 (LWP 24563)]
```

2. Set logging to a file in case of lengthy output from some commands and set color highlighting off:

```
(gdb) set logging file App2C.log

(gdb) set logging enabled on
Copying output to App2C.log.
Copying debug output to App2C.log.

(gdb) set style enabled off
```

3. List all threads:

```
(gdb) info threads
  Id   Target Id         Frame
* 1    LWP 24563         0x0000000000000000 in ?? ()
  2    LWP 24559         0x000000000040c9f4 in nanosleep ()
  3    LWP 24560         0x000000000040c9f4 in nanosleep ()
```

```
  4    LWP 24561         0x000000000040c9f4 in nanosleep ()
  5    LWP 24564         0x000000000040c9f4 in nanosleep ()
  6    LWP 24562         0x000000000040c9f4 in nanosleep ()
```

4. The problem thread seems to be the current thread:

```
(gdb) bt
#0  0x0000000000000000 in ?? ()
#1  0x000000000040322c in proc ()
#2  0x0000000000403314 in bar_four ()
#3  0x0000000000403328 in foo_four ()
#4  0x0000000000403340 in thread_four ()
#5  0x0000000000404c74 in start_thread ()
#6  0x0000000000429ba0 in thread_start ()
```

Note: It looks like our GDB version prints the non-existent *proc* function instead of *procB*.

```
(gdb) disassemble proc
No symbol table is loaded.  Use the "file" command.
```

5. Check the CPU instruction and a dereferenced pointer for any signs of a NULL pointer:

```
(gdb) disassemble procB
Dump of assembler code for function procB:
   0x0000000000403210 <+0>:     stp     x29, x30, [sp, #-32]!
   0x0000000000403214 <+4>:     mov     x29, sp
   0x0000000000403218 <+8>:     mov     w0, #0x1                        // #1
   0x000000000040321c <+12>:    bl      0x424be4 <sleep>
   0x0000000000403220 <+16>:    str     xzr, [x29, #24]
   0x0000000000403224 <+20>:    ldr     x0, [x29, #24]
   0x0000000000403228 <+24>:    blr     x0
   0x000000000040322c <+28>:    ldp     x29, x30, [sp], #32
   0x0000000000403230 <+32>:    ret
End of assembler dump.

(gdb) info r x0
x0             0x0   0
```

6. List all thread stack traces to check for other anomalies, such as non-waiting active threads:

```
(gdb) thread apply all bt

Thread 6 (LWP 24562):
#0  0x000000000040c9f4 in nanosleep ()
#1  0x0000000000424cf4 in sleep ()
#2  0x00000000004032d0 in bar_three ()
#3  0x00000000004032e4 in foo_three ()
#4  0x00000000004032fc in thread_three ()
#5  0x0000000000404c74 in start_thread ()
#6  0x0000000000429ba0 in thread_start ()

Thread 5 (LWP 24564):
#0  0x000000000040c9f4 in nanosleep ()
#1  0x0000000000424cf4 in sleep ()
#2  0x000000000040335c in bar_five ()
#3  0x0000000000403370 in foo_five ()
#4  0x0000000000403388 in thread_five ()
#5  0x0000000000404c74 in start_thread ()
#6  0x0000000000429ba0 in thread_start ()
```

```
Thread 4 (LWP 24561):
#0  0x000000000040c9f4 in nanosleep ()
#1  0x0000000000424cf4 in sleep ()
#2  0x00000000004031f8 in procA ()
#3  0x0000000000403288 in bar_two ()
#4  0x000000000040329c in foo_two ()
#5  0x00000000004032b4 in thread_two ()
#6  0x0000000000404c74 in start_thread ()
#7  0x0000000000429ba0 in thread_start ()

Thread 3 (LWP 24560):
#0  0x000000000040c9f4 in nanosleep ()
#1  0x0000000000424cf4 in sleep ()
#2  0x0000000000403244 in bar_one ()
#3  0x0000000000403258 in foo_one ()
#4  0x0000000000403270 in thread_one ()
#5  0x0000000000404c74 in start_thread ()
#6  0x0000000000429ba0 in thread_start ()

Thread 2 (LWP 24559):
#0  0x000000000040c9f4 in nanosleep ()
#1  0x0000000000424cf4 in sleep ()
#2  0x0000000000403424 in main ()

Thread 1 (LWP 24563):
#0  0x0000000000000000 in ?? ()
#1  0x000000000040322c in proc ()
#2  0x0000000000403314 in bar_four ()
#3  0x0000000000403328 in foo_four ()
#4  0x0000000000403340 in thread_four ()
#5  0x0000000000404c74 in start_thread ()
#6  0x0000000000429ba0 in thread_start ()
```

Exercise A2C (A64, WinDbg Preview)

Goal: Learn how to identify exceptions, find problem threads and CPU instructions.

Patterns: NULL Pointer (Code); Missing Frame.

1. Launch WinDbg Preview.

2. Load *core.24559* dump file from the A64\App2C folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App2C\core.24559]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
(5fef.5ff3): Signal SIGSEGV (Segmentation fault) code SEGV_MAPERR (Address not mapped to
object) at 0x0*** WARNING: Unable to verify timestamp for App2C
00000000`00000000 ?? ???
```

3. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App2C\App2C.log
Opened log file 'C:\ALCDA2\A64\App2C\App2C.log'
```

4. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App2C\
Symbol search path is: srv*;C:\ALCDA2\A64\App2C\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app2c\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App2C\
*** WARNING: Unable to verify timestamp for App2C

0:000> .reload
..
*** WARNING: Unable to verify timestamp for App2C

************* Symbol Loading Error Summary **************
Module name            Error
App2C                  The system cannot find the file specified
```

```
You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

5. The problem thread seems to be the current thread:

```
0:000> k
 # Child-SP          RetAddr               Call Site
00 0000fffc`694ce800 00000000`0040322c     0x0
01 0000fffc`694ce800 00000000`00403314     App2C!procB+0x1c
02 0000fffc`694ce820 00000000`00403328     App2C!bar_four+0xc
03 0000fffc`694ce830 00000000`00403340     App2C!foo_four+0xc
04 0000fffc`694ce840 00000000`00404c74     App2C!thread_four+0x10
05 0000fffc`694ce860 00000000`00429ba0     App2C!start_thread+0xb4
06 0000fffc`694ce990 ffffffff`ffffffff     App2C!thread_start+0x30
07 0000fffc`694ce990 00000000`00000000     0xffffffff`ffffffff
```

6. Disassemble the return address backward:

```
0:000> ub 00000000`0040322c
App2C!procA+0x24:
00000000`0040320c d65f03c0 ret
App2C!procB:
00000000`00403210 a9be7bfd stp         fp,lr,[sp,#-0x20]!
00000000`00403214 910003fd mov         fp,sp
00000000`00403218 52800020 mov         w0,#1
00000000`0040321c 94008672 bl          App2C!sleep (00000000`00424be4)
00000000`00403220 f9000fbf str         xzr,[fp,#0x18]
00000000`00403224 f9400fa0 ldr         x0,[fp,#0x18]
00000000`00403228 d63f0000 blr         x0
```

Note: xzr generates 0, which is loaded into x0. If you use WinDbg from Debugging Tools for Windows, you may have a missing frame and a different return address:

```
0:000> k
 # Child-SP          RetAddr               Call Site
00 0000fffc`694ce800 00000000`00403314     0x0
01 0000fffc`694ce810 00000000`00403328     App2C!bar_four+0xc
02 0000fffc`694ce830 00000000`00403340     App2C!foo_four+0xc
03 0000fffc`694ce840 00000000`00404c74     App2C!thread_four+0x10
04 0000fffc`694ce860 00000000`00429ba0     App2C!start_thread+0xb4
05 0000fffc`694ce990 ffffffff`ffffffff     App2C!thread_start+0x30
06 0000fffc`694ce990 00000000`00000000     0xffffffff`ffffffff

0:000> ub 00000000`00403314
App2C!thread_three+0x8:
00000000`004032f4 f9000fa0 str         x0,[fp,#0x18]
00000000`004032f8 97fffff8 bl          App2C!foo_three (00000000`004032d8)
00000000`004032fc d2800000 mov         x0,#0
00000000`00403300 a8c27bfd ldp         fp,lr,[sp],#0x20
00000000`00403304 d65f03c0 ret
App2C!bar_four:
00000000`00403308 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`0040330c 910003fd mov         fp,sp
00000000`00403310 97ffffc0 bl          App2C!procB (00000000`00403210)
```

Note: Instead of a problem instruction, we see a procedure call. We disassemble *procB*, check the CPU instruction and a dereferenced pointer for any signs of a NULL pointer:

```
0:000> uf procB
App2C!procB:
00000000`00403210 a9be7bfd stp         fp,lr,[sp,#-0x20]!
00000000`00403214 910003fd mov         fp,sp
00000000`00403218 52800020 mov         w0,#1
00000000`0040321c 94008672 bl          App2C!sleep (00000000`00424be4)
00000000`00403220 f9000fbf str         xzr,[fp,#0x18]
00000000`00403224 f9400fa0 ldr         x0,[fp,#0x18]
00000000`00403228 d63f0000 blr         x0
00000000`0040322c a8c27bfd ldp         fp,lr,[sp],#0x20
00000000`00403230 d65f03c0 ret

0:000> r x0
x0=0000000000000000
```

Note: We see that 0 (the value of the xzr register) was stored in a stack location, then it was loaded into the x0 register. The fp register is an alias to the x29 register.

7. List all thread stack traces to check for other anomalies, such as non-waiting active threads:

```
0:000> ~*k

Unable to get thread data for thread 0
.  0  Id: 5fef.5ff3 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`694ce800 00000000`00403314     0x0
01 0000fffc`694ce810 00000000`00403328     App2C!bar_four+0xc
02 0000fffc`694ce830 00000000`00403340     App2C!foo_four+0xc
03 0000fffc`694ce840 00000000`00404c74     App2C!thread_four+0x10
04 0000fffc`694ce860 00000000`00429ba0     App2C!start_thread+0xb4
05 0000fffc`694ce990 ffffffff`ffffffff     App2C!thread_start+0x30
06 0000fffc`694ce990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 1
   1  Id: 5fef.5fef Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`c86b1620 00000000`00424cf4     App2C!_libc_nanosleep+0x24
01 0000ffff`c86b1660 00000000`00403424     App2C!sleep+0x110
02 0000ffff`c86b1850 00000000`0040ec8c     App2C!main+0x90
03 0000ffff`c86b18a0 00000000`00403090     App2C!_libc_start_main+0x304
04 0000ffff`c86b1a00 00000000`00000000     App2C!start+0x4c

Unable to get thread data for thread 2
   2  Id: 5fef.5ff0 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`6acfe5f0 00000000`00424cf4     App2C!_libc_nanosleep+0x24
01 0000fffc`6acfe630 00000000`00403244     App2C!sleep+0x110
02 0000fffc`6acfe820 00000000`00403258     App2C!bar_one+0x10
03 0000fffc`6acfe830 00000000`00403270     App2C!foo_one+0xc
04 0000fffc`6acfe840 00000000`00404c74     App2C!thread_one+0x10
05 0000fffc`6acfe860 00000000`00429ba0     App2C!start_thread+0xb4
06 0000fffc`6acfe990 ffffffff`ffffffff     App2C!thread_start+0x30
07 0000fffc`6acfe990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 3
```

```
   3  Id: 5fef.5ff1 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`6a4ee5d0 00000000`00424cf4     App2C!_libc_nanosleep+0x24
01 0000fffc`6a4ee610 00000000`004031f8     App2C!sleep+0x110
02 0000fffc`6a4ee800 00000000`00403288     App2C!procA+0x10
03 0000fffc`6a4ee820 00000000`0040329c     App2C!bar_two+0xc
04 0000fffc`6a4ee830 00000000`004032b4     App2C!foo_two+0xc
05 0000fffc`6a4ee840 00000000`00404c74     App2C!thread_two+0x10
06 0000fffc`6a4ee860 00000000`00429ba0     App2C!start_thread+0xb4
07 0000fffc`6a4ee990 ffffffff`ffffffff     App2C!thread_start+0x30
08 0000fffc`6a4ee990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 4
   4  Id: 5fef.5ff4 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`68cbe5f0 00000000`00424cf4     App2C!_libc_nanosleep+0x24
01 0000fffc`68cbe630 00000000`0040335c     App2C!sleep+0x110
02 0000fffc`68cbe820 00000000`00403370     App2C!bar_five+0x10
03 0000fffc`68cbe830 00000000`00403388     App2C!foo_five+0xc
04 0000fffc`68cbe840 00000000`00404c74     App2C!thread_five+0x10
05 0000fffc`68cbe860 00000000`00429ba0     App2C!start_thread+0xb4
06 0000fffc`68cbe990 ffffffff`ffffffff     App2C!thread_start+0x30
07 0000fffc`68cbe990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 5
   5  Id: 5fef.5ff2 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`69cde5f0 00000000`00424cf4     App2C!_libc_nanosleep+0x24
01 0000fffc`69cde630 00000000`004032d0     App2C!sleep+0x110
02 0000fffc`69cde820 00000000`004032e4     App2C!bar_three+0x10
03 0000fffc`69cde830 00000000`004032fc     App2C!foo_three+0xc
04 0000fffc`69cde840 00000000`00404c74     App2C!thread_three+0x10
05 0000fffc`69cde860 00000000`00429ba0     App2C!start_thread+0xb4
06 0000fffc`69cde990 ffffffff`ffffffff     App2C!thread_start+0x30
07 0000fffc`69cde990 00000000`00000000     0xffffffff`ffffffff
```

9. We close logging before exiting WinDbg Preview:

```
0:000> .logclose
Closing open log file 'C:\ALCDA2\A64\App2C\App2C.log'
```

Exercise A2S

- **Goal:** Learn how to use external debugging information
- \ALCDA-Dumps\Exercise-A2S-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A2S-A64-GDB.pdf

Exercise A2S (x64, GDB)

Goal: Learn how to use external debugging information.

1. Load *core.App2S* dump file and *App2S* executable from the x64/App2S directory:

```
~/ALCDA2/x64/App2S$ gdb -c core.App2S -se App2S
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App2S...(no debugging symbols found)...done.
[New LWP 3736]
[New LWP 3738]
[New LWP 3735]
[New LWP 3734]
[New LWP 3737]
[New LWP 3739]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App2S'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x0000000000401bcb in procA ()
[Current thread is 1 (Thread 0x7f30da538700 (LWP 3736))]
```

2. We check the current stack trace:

```
(gdb) bt
#0  0x0000000000401bcb in procA ()
#1  0x0000000000401c49 in bar_two ()
#2  0x0000000000401c5a in foo_two ()
#3  0x0000000000401c73 in thread_two ()
#4  0x0000000000403123 in start_thread ()
#5  0x00000000004442bf in clone ()
```

Note: We see that the problem happened in *procA,* but we want to locate it in the source code. The executable *App2S* was stripped from debugging symbols before its distribution to customers. Fortunately, the executable with debugging information was saved in a separate *App2S.debug* file.

3. We load the *App2S.debug* file with debugging symbols:

```
(gdb) symbol-file App2S.debug
Reading symbols from App2S.debug...done.
```

4. Now we get the stack trace with file numbers:

```
(gdb) bt
#0  0x0000000000401bcb in procA () at main.c:26
#1  0x0000000000401c49 in bar_two () at main.c:56
#2  0x0000000000401c5a in foo_two () at main.c:56
#3  0x0000000000401c73 in thread_two (arg=0x0) at main.c:56
#4  0x0000000000403123 in start_thread (arg=<optimized out>) at pthread_create.c:486
#5  0x00000000004442bf in clone ()
```

5. If we have the source file, we can list the exact location:

```
(gdb) list main.c:26
21      {
22          sleep(1);
23
24          int *p = NULL;
25
26          *p = 1;
27      }
28
29      void procB()
30      {
```

6. Alternatively, we can load the executable with debugging symbols from the start:

```
~/ALCDA2/x64/App2S$ gdb -c core.App2S -se App2S.debug
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App2S.debug...done.

warning: core file may not match specified executable file.
[New LWP 3736]
[New LWP 3738]
[New LWP 3735]
[New LWP 3734]
[New LWP 3737]
[New LWP 3739]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App2S'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x0000000000401bcb in procA () at main.c:26

warning: Source file is more recent than executable.
26          *p = 1;
```

```
[Current thread is 1 (Thread 0x7f30da538700 (LWP 3736))]

(gdb) bt
#0  0x0000000000401bcb in procA () at main.c:26
#1  0x0000000000401c49 in bar_two () at main.c:56
#2  0x0000000000401c5a in foo_two () at main.c:56
#3  0x0000000000401c73 in thread_two (arg=0x0) at main.c:56
#4  0x0000000000403123 in start_thread (arg=<optimized out>) at pthread_create.c:486
#5  0x00000000004442bf in clone ()
```

Note: We also see the warning that the source code is more recent (we modified some comments after compilation).

Exercise A2S (A64, GDB)

Goal: Learn how to use external debugging information.

1. Load *core._home_ubuntu_ALCDA2_A64_App2S_App2S.1001.3d452460-e216-4918-b09f-304672052efe.202652.172563749* dump file and *App2S* executable from the A64/App2S directory:

```
~/ALCDA2/A64/App2S$ gdb -c core._home_ubuntu_ALCDA2_A64_App2S_App2S.1001.3d452460-e216-4918-b09f-304672052efe.202652.172563749 -se App2S
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App2S...
(No debugging symbols found in App2S)
[New LWP 202654]
[New LWP 202657]
[New LWP 202652]
[New LWP 202653]
[New LWP 202655]
[New LWP 202656]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/aarch64-linux-gnu/libthread_db.so.1".
Core was generated by `./App2S'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x00000000004006f0 in procA ()
[Current thread is 1 (Thread 0xffff8a23a480 (LWP 202654))]
```

2. Set logging to a file in case of lengthy output from some commands and set color highlighting off:

```
(gdb) set logging file App2S.log

(gdb) set logging enabled on
Copying output to App2S.log.
Copying debug output to App2S.log.

(gdb) set style enabled off
```

3. We check the current stack trace:

```
(gdb) bt
#0  0x00000000004006f0 in procA ()
#1  0x000000000040077c in bar_two ()
#2  0x0000000000400790 in foo_two ()
#3  0x00000000004007a8 in thread_two ()
#4  0x000000000040eca4 in start_thread ()
```

```
#5  0x000000000044365c in thread_start ()
```

Note: We see that the problem happened in *procA,* but we want to locate it in the source code. The executable *App2S* was stripped from debugging symbols before its distribution to customers. Fortunately, the executable with debugging information was saved in a separate *App2S.debug* file.

4. We load the *App2S.debug* file with debugging symbols:

```
(gdb) symbol-file App2S.debug
Reading symbols from App2S.debug...
```

5. Now we get the stack trace with file numbers:

```
(gdb) bt
#0  0x00000000004006f0 in procA () at main.c:26
#1  0x000000000040077c in bar_two () at main.c:56
#2  0x0000000000400790 in foo_two () at main.c:56
#3  0x00000000004007a8 in thread_two (arg=0x0) at main.c:56
#4  0x000000000040eca4 in start_thread ()
#5  0x000000000044365c in thread_start ()
```

6. If we have the source file, we can list the exact location:

```
(gdb) list main.c:26
21      {
22          sleep(1);
23
24          int *p = NULL;
25
26          *p = 1;
27      }
28
29      void procB()
30      {
```

7. Alternatively, we can load the executable with debugging symbols from the start:

```
~/ALCDA2/A64/App2S$ gdb -c core._home_ubuntu_ALCDA2_A64_App2S_App2S.1001.3d452460-e216-4918-
b09f-304672052efe.202652.172563749 -se App2S.debug
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App2S.debug...
[New LWP 202654]
[New LWP 202657]
[New LWP 202652]
```

```
[New LWP 202653]
[New LWP 202655]
[New LWP 202656]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/aarch64-linux-gnu/libthread_db.so.1".
Core was generated by `./App2S'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x00000000004006f0 in procA () at main.c:26
26          *p = 1;
[Current thread is 1 (Thread 0xffff8a23a480 (LWP 202654))]

(gdb) set style enabled off

(gdb) bt
#0  0x00000000004006f0 in procA () at main.c:26
#1  0x000000000040077c in bar_two () at main.c:56
#2  0x0000000000400790 in foo_two () at main.c:56
#3  0x00000000004007a8 in thread_two (arg=0x0) at main.c:56
#4  0x000000000040eca4 in start_thread ()
#5  0x000000000044365c in thread_start ()
```

Exercise A3

- **Goal:** Learn how to identify spiking threads
- **Patterns:** Active Thread; Spiking Thread
- \ALCDA-Dumps\Exercise-A3-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A3-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A3-A64-WinDbg.pdf

Exercise A3 (x64, GDB)

Goal: Learn how to identify spiking threads.

Patterns: Active Thread; Spiking Thread.

1. The application **App3** was consuming 100% CPU (from **top** command output):

```
$ top
top - 13:19:10 up 23:14,  0 users,  load average: 0.74, 0.25, 0.09
Tasks:  10 total,   1 running,   9 sleeping,   0 stopped,   0 zombie
%Cpu(s): 12.5 us,  0.0 sy,  0.0 ni, 87.5 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   7912.4 total,   5556.1 free,    270.2 used,   2086.1 buff/cache
MiB Swap:   2048.0 total,   2048.0 free,      0.0 used.   7386.6 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 3975 coredump  20   0   42068      4      0 S 100.0   0.0   1:20.69 App3
    1 root      20   0    1744   1080   1016 S   0.0   0.0   0:00.00 init
  117 root      20   0    1764     84      0 S   0.0   0.0   0:06.16 init
 3121 coredump  20   0   42068    112    104 S   0.0   0.0   0:00.02 App1
 3343 root      20   0    1764     68      0 S   0.0   0.0   0:00.00 init
 3344 root      20   0    1764     84      0 S   0.0   0.0   0:03.54 init
 3345 coredump  20   0    6992   3852   3248 S   0.0   0.0   0:00.02 bash
 3349 coredump  20   0   26124   9476   6988 S   0.0   0.1   0:07.67 mc
 3351 coredump  20   0    7124   3872   3212 S   0.0   0.0   0:00.87 bash
 3940 coredump  20   0   10968   3524   3040 R   0.0   0.0   0:00.00 top
```

Its core dump was saved using **gcore**:

```
~/ALCDA2/x64/App3$ gcore -o App3.core 3975
[New LWP 3976]
[New LWP 3977]
[New LWP 3978]
[New LWP 3979]
[New LWP 3980]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
0x0000000000441a80 in nanosleep ()
Saved corefile App3.core.3975
[Inferior 1 (process 3975) detached]
```

2. Load *App3.core.3975* dump file and *App3* executable from the x64/App3 directory:

```
~/ALCDA2/x64/App3$ gdb -c App3.core.3975 -se App3
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.
```

```
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App3...done.
[New LWP 3975]
[New LWP 3976]
[New LWP 3977]
[New LWP 3978]
[New LWP 3979]
[New LWP 3980]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App3'.
#0  0x0000000000441a80 in nanosleep ()
[Current thread is 1 (Thread 0xbfc880 (LWP 3975))]
```

3. List all threads:

```
(gdb) info threads
  Id   Target Id                         Frame
* 1    Thread 0xbfc880 (LWP 3975)        0x0000000000441a80 in nanosleep ()
  2    Thread 0x7fc68ca9f700 (LWP 3976) 0x0000000000441a80 in nanosleep ()
  3    Thread 0x7fc68c29e700 (LWP 3977) 0x0000000000441a80 in nanosleep ()
  4    Thread 0x7fc68ba9d700 (LWP 3978) 0x0000000000441a80 in nanosleep ()
  5    Thread 0x7fc68b29c700 (LWP 3979) 0x0000000000441a80 in nanosleep ()
  6    Thread 0x7fc68aa9b700 (LWP 3980) 0x0000000000401e04 in __sqrt_finite ()
```

4. Switch to the active thread #6:

```
(gdb) thread 6
[Switching to thread 6 (Thread 0x7fc68aa9b700 (LWP 3980))]
#0  0x0000000000401e04 in __sqrt_finite ()

(gdb) bt
#0  0x0000000000401e04 in __sqrt_finite ()
#1  0x0000000000401bdc in proc ()
#2  0x0000000000401cf1 in bar_five ()
#3  0x0000000000401d02 in foo_five ()
#4  0x0000000000401d1b in thread_five ()
#5  0x0000000000403143 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x00000000004442df in clone ()
```

5. Disassemble the current instruction and check if it is normal:

```
(gdb) x/i 0x0000000000401e04
   0x401e04 <__sqrt_finite+4>:  retq

(gdb) disassemble __sqrt_finite
Dump of assembler code for function __sqrt_finite:
   0x0000000000401e00 <+0>:     sqrtsd %xmm0,%xmm0
   0x0000000000401e04 <+4>:     retq
End of assembler dump.
```

6. Disassemble the return address for the *proc* function (this GDB version shows proc instead of *procB* from the source code) to see an infinite loop:

```
(gdb) disassemble 0x0000000000401bdc
Dump of assembler code for function procB:
   0x0000000000401bbd <+0>:     push   %rbp
   0x0000000000401bbe <+1>:     mov    %rsp,%rbp
   0x0000000000401bc1 <+4>:     sub    $0x10,%rsp
   0x0000000000401bc5 <+8>:     movsd  0x9243b(%rip),%xmm0        # 0x494008
   0x0000000000401bcd <+16>:    movsd  %xmm0,-0x8(%rbp)
   0x0000000000401bd2 <+21>:    movsd  -0x8(%rbp),%xmm0
   0x0000000000401bd7 <+26>:    callq  0x401de0 <sqrtf64>
   0x0000000000401bdc <+31>:    movq   %xmm0,%rax
   0x0000000000401be1 <+36>:    mov    %rax,-0x8(%rbp)
   0x0000000000401be5 <+40>:    jmp    0x401bd2 <procB+21>
End of assembler dump.
```

Exercise A3 (A64, GDB)

Goal: Learn how to identify spiking threads.

Patterns: Active Thread; Spiking Thread.

1. The application **App3** was consuming 100% CPU (from **top** command output):

```
$ top
top - 19:59:39 up 31 days, 19:09,  1 user,  load average: 1.00, 0.72, 0.34
Tasks: 184 total,   1 running, 128 sleeping,   0 stopped,   0 zombie
%Cpu(s): 25.1 us,  0.0 sy,  0.0 ni, 74.8 id,  0.0 wa,  0.1 hi,  0.0 si,  0.0 st
KiB Mem : 23799872 total, 19518400 free,   816064 used,  3465408 buff/cache
KiB Swap:  8388544 total,  8388544 free,        0 used. 19342592 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
21335 opc       20   0   42752     64      0 S  99.7  0.0   6:08.44 App3
21348 opc       20   0  119360   8000   3776 R   0.3  0.0   0:00.37 top
    1 root      20   0  165888  14976   7488 S   0.0  0.1   5:44.10 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:02.24 kthreadd
    3 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 rcu_par_gp
    6 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/0:0H-kb
    8 root       0 -20       0      0      0 I   0.0  0.0   0:00.04 mm_percpu_wq
    9 root      20   0       0      0      0 S   0.0  0.0   0:27.66 ksoftirqd/0
   10 root      20   0       0      0      0 I   0.0  0.0   9:59.60 rcu_sched
   11 root      rt   0       0      0      0 S   0.0  0.0   0:31.70 migration/0
   13 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/0
   14 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/1
   15 root      rt   0       0      0      0 S   0.0  0.0   0:34.69 migration/1
   16 root      20   0       0      0      0 S   0.0  0.0   0:23.54 ksoftirqd/1
   18 root       0 -20       0      0      0 I   0.0  0.0   0:31.65 kworker/1:0H-kb
   19 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/1:1H
   20 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/2
   21 root      rt   0       0      0      0 S   0.0  0.0   0:31.60 migration/2
   22 root      20   0       0      0      0 S   0.0  0.0   0:23.26 ksoftirqd/2
   24 root       0 -20       0      0      0 I   0.0  0.0   0:25.78 kworker/2:0H-kb
   25 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/3
```

Its core dump was saved using **gcore**:

```
~/ALCDA2/A64/App3$ gcore -o App3.core 21335
[New LWP 21340]
[New LWP 21339]
[New LWP 21338]
[New LWP 21337]
[New LWP 21336]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
0x0000000000414364 in nanosleep ()
Saved corefile App3.core.21335
[Inferior 1 (process 21335) detached]
```

2. Load *App3.core.21335* dump file and *App3* executable from the A64/App3 directory:

```
~/ALCDA2/A64/App3$ gdb -c App3.core.21335 -se App3
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App3...
(No debugging symbols found in App3)

warning: Can't open file /home/opc/ALCDA2/App3/App3 during file-backed mapping note processing
[New LWP 21336]
[New LWP 21337]
[New LWP 21338]
[New LWP 21339]
[New LWP 21340]
[New LWP 21335]
Core was generated by `./App3'.
#0  0x0000000000414364 in nanosleep ()
[Current thread is 1 (LWP 21336)]
```

3. Set logging to a file in case of lengthy output from some commands and set color highlighting off:

```
(gdb) set logging file App3.log

(gdb) set logging enabled on
Copying output to App3.log.
Copying debug output to App3.log.

(gdb) set style enabled off
```

4. List all threads:

```
(gdb) info threads
  Id   Target Id         Frame
* 1    LWP 21336         0x0000000000414364 in nanosleep ()
  2    LWP 21337         0x0000000000414364 in nanosleep ()
  3    LWP 21338         0x0000000000414364 in nanosleep ()
  4    LWP 21339         0x0000000000414364 in nanosleep ()
  5    LWP 21340         0x0000000000408400 in __sqrt_finite ()
  6    LWP 21335         0x0000000000414364 in nanosleep ()
```

5. Switch to the active thread #5:

```
(gdb) thread 5
[Switching to thread 5 (LWP 21340)]
#0  0x0000000000408400 in __sqrt_finite ()
```

```
(gdb) bt
#0  0x0000000000408400 in __sqrt_finite ()
#1  0x0000000000403214 in proc ()
#2  0x0000000000403350 in bar_five ()
#3  0x0000000000403364 in foo_five ()
#4  0x000000000040337c in thread_five ()
#5  0x000000000040c5e4 in start_thread ()
#6  0x0000000000431920 in thread_start ()
```

6. Disassemble the current instruction and check if it is normal:

```
(gdb) x/i 0x0000000000408400
=> 0x408400 <__sqrt_finite+328>:        fcmpe   d1, d0

(gdb) disassemble __sqrt_finite
Dump of assembler code for function __sqrt_finite:
   0x00000000004082b8 <+0>:     fmov    x1, d0
   0x00000000004082bc <+4>:     asr     x0, x1, #32
   0x00000000004082c0 <+8>:     and     w2, w0, #0x1fffff
   0x00000000004082c4 <+12>:    orr     w3, w2, #0x3fe00000
   0x00000000004082c8 <+16>:    stp     x29, x30, [sp, #-16]!
   0x00000000004082cc <+20>:    sub     w4, w0, #0x100, lsl #12
   0x00000000004082d0 <+24>:    bfi     x1, x3, #32, #32
   0x00000000004082d4 <+28>:    mov     w3, #0x7fdfffff                 // #2145386495
   0x00000000004082d8 <+32>:    cmp     w4, w3
   0x00000000004082dc <+36>:    mov     x29, sp
   0x00000000004082e0 <+40>:    adrp    x4, 0x490000 <arena_thread_freeres+72>
   0x00000000004082e4 <+44>:    asr     w2, w2, #14
   0x00000000004082e8 <+48>:    add     x4, x4, #0xce0
   0x00000000004082ec <+52>:    mov     x3, #0x0                        // #0
   0x00000000004082f0 <+56>:    ldr     d2, [x4, w2, sxtw #3]
   0x00000000004082f4 <+60>:    fmov    d3, x1
   0x00000000004082f8 <+64>:    b.hi    0x40841c <__sqrt_finite+356>  // b.pmore
   0x00000000004082fc <+68>:    fmul    d0, d2, d3
   0x0000000000408300 <+72>:    fmul    d0, d0, d2
   0x0000000000408304 <+76>:    fmov    d1, #1.0000000000000000e+00
   0x0000000000408308 <+80>:    fsub    d0, d1, d0
   0x000000000040830c <+84>:    ldr     d1, 0x408498
   0x0000000000408310 <+88>:    ldr     d4, 0x4084a0
   0x0000000000408314 <+92>:    fmul    d1, d0, d1
   0x0000000000408318 <+96>:    fadd    d1, d1, d4
   0x000000000040831c <+100>:   ldr     d4, 0x4084a8
   0x0000000000408320 <+104>:   fmul    d1, d1, d0
   0x0000000000408324 <+108>:   fadd    d1, d1, d4
   0x0000000000408328 <+112>:   fmul    d0, d1, d0
   0x000000000040832c <+116>:   ldr     d1, 0x4084b0
   0x0000000000408330 <+120>:   and     w0, w0, #0x7fe00000
   0x0000000000408334 <+124>:   fadd    d0, d0, d1
   0x0000000000408338 <+128>:   fmul    d0, d0, d2
   0x000000000040833c <+132>:   ldr     d1, 0x4084b8
   0x0000000000408340 <+136>:   fmul    d2, d0, d3
   0x0000000000408344 <+140>:   fadd    d4, d2, d1
   0x0000000000408348 <+144>:   fsub    d4, d4, d1
   0x000000000040834c <+148>:   fsub    d5, d2, d4
   0x0000000000408350 <+152>:   fmul    d1, d4, d4
   0x0000000000408354 <+156>:   fadd    d4, d2, d4
   0x0000000000408358 <+160>:   fsub    d1, d3, d1
   0x000000000040835c <+164>:   fmul    d4, d5, d4
   0x0000000000408360 <+168>:   fmov    d5, #5.0000000000000000e-01
   0x0000000000408364 <+172>:   fmul    d0, d0, d5
```

```
   0x0000000000408368 <+176>:   fsub    d4, d1, d4
   0x000000000040836c <+180>:   fmul    d4, d0, d4
   0x0000000000408370 <+184>:   fadd    d1, d2, d4
   0x0000000000408374 <+188>:   fsub    d2, d2, d1
   0x0000000000408378 <+192>:   ldr     d0, 0x4084c0
--Type <RET> for more, q to quit, c to continue without paging--
   0x000000000040837c <+196>:   fadd    d4, d2, d4
   0x0000000000408380 <+200>:   fmul    d0, d4, d0
   0x0000000000408384 <+204>:   fadd    d0, d0, d1
   0x0000000000408388 <+208>:   mov     w1, #0x20000000                 // #536870912
   0x000000000040838c <+212>:   add     w0, w1, w0, lsr #1
   0x0000000000408390 <+216>:   fcmp    d0, d1
   0x0000000000408394 <+220>:   bfi     x3, x0, #32, #32
   0x0000000000408398 <+224>:   b.eq    0x40845c <__sqrt_finite+420>  // b.none
   0x000000000040839c <+228>:   fmov    d0, #1.500000000000000000e+00
   0x00000000004083a0 <+232>:   ldr     d2, 0x4084c8
   0x00000000004083a4 <+236>:   fmul    d4, d4, d0
   0x00000000004083a8 <+240>:   fadd    d0, d4, d1
   0x00000000004083ac <+244>:   fmul    d4, d1, d2
   0x00000000004083b0 <+248>:   fmul    d2, d0, d2
   0x00000000004083b4 <+252>:   fsub    d6, d1, d4
   0x00000000004083b8 <+256>:   fsub    d5, d0, d2
   0x00000000004083bc <+260>:   fadd    d4, d6, d4
   0x00000000004083c0 <+264>:   fadd    d2, d5, d2
   0x00000000004083c4 <+268>:   fsub    d6, d0, d2
   0x00000000004083c8 <+272>:   fmul    d16, d4, d2
   0x00000000004083cc <+276>:   fmul    d5, d1, d0
   0x00000000004083d0 <+280>:   fsub    d7, d1, d4
   0x00000000004083d4 <+284>:   fsub    d16, d16, d5
   0x00000000004083d8 <+288>:   fmul    d4, d4, d6
   0x00000000004083dc <+292>:   fmul    d2, d7, d2
   0x00000000004083e0 <+296>:   fadd    d4, d16, d4
   0x00000000004083e4 <+300>:   fadd    d2, d4, d2
   0x00000000004083e8 <+304>:   fmul    d7, d7, d6
   0x00000000004083ec <+308>:   fsub    d3, d5, d3
   0x00000000004083f0 <+312>:   fadd    d6, d2, d7
   0x00000000004083f4 <+316>:   fadd    d3, d3, d6
   0x00000000004083f8 <+320>:   fcmpe   d3, #0.0
   0x00000000004083fc <+324>:   b.mi    0x408484 <__sqrt_finite+460>  // b.first
=> 0x0000000000408400 <+328>:   fcmpe   d1, d0
   0x0000000000408404 <+332>:   b.gt    0x40840c <__sqrt_finite+340>
   0x0000000000408408 <+336>:   fmov    d0, d1
   0x000000000040840c <+340>:   fmov    d1, x3
   0x0000000000408410 <+344>:   fmul    d0, d0, d1
   0x0000000000408414 <+348>:   ldp     x29, x30, [sp], #16
   0x0000000000408418 <+352>:   ret
   0x000000000040841c <+356>:   and     w2, w0, #0x7ff00000
   0x0000000000408420 <+360>:   mov     w1, #0x7ff00000                 // #2146435072
   0x0000000000408424 <+364>:   cmp     w2, w1
   0x0000000000408428 <+368>:   b.eq    0x40846c <__sqrt_finite+436>  // b.none
   0x000000000040842c <+372>:   fcmp    d0, #0.0
   0x0000000000408430 <+376>:   b.eq    0x408414 <__sqrt_finite+348>  // b.none
   0x0000000000408434 <+380>:   tbnz    w0, #31, 0x408478 <__sqrt_finite+448>
   0x0000000000408438 <+384>:   adrp    x0, 0x491000 <inroot+800>
   0x000000000040843c <+388>:   ldr     d1, [x0, #224]
   0x0000000000408440 <+392>:   fmul    d0, d0, d1
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000408444 <+396>:   bl      0x4082b8 <__sqrt_finite>
   0x0000000000408448 <+400>:   adrp    x0, 0x491000 <inroot+800>
   0x000000000040844c <+404>:   ldr     d1, [x0, #232]
```

```
   0x0000000000408450 <+408>:   ldp     x29, x30, [sp], #16
   0x0000000000408454 <+412>:   fmul    d0, d1, d0
   0x0000000000408458 <+416>:   ret
   0x000000000040845c <+420>:   fmov    d2, x3
   0x0000000000408460 <+424>:   fmul    d0, d1, d2
   0x0000000000408464 <+428>:   ldp     x29, x30, [sp], #16
   0x0000000000408468 <+432>:   ret
   0x000000000040846c <+436>:   fmul    d1, d0, d0
   0x0000000000408470 <+440>:   fadd    d0, d1, d0
   0x0000000000408474 <+444>:   b       0x408414 <__sqrt_finite+348>
   0x0000000000408478 <+448>:   fsub    d0, d0, d0
   0x000000000040847c <+452>:   fdiv    d0, d0, d0
   0x0000000000408480 <+456>:   b       0x408414 <__sqrt_finite+348>
   0x0000000000408484 <+460>:   fcmpe   d1, d0
   0x0000000000408488 <+464>:   b.mi    0x40840c <__sqrt_finite+340>  // b.first
   0x000000000040848c <+468>:   fmov    d0, d1
   0x0000000000408490 <+472>:   b       0x40840c <__sqrt_finite+340>
End of assembler dump.
```

Note: The function is quite large compared to the x64 version, where there is a dedicated instruction.

7. Disassemble the return address for the *proc* function (this GDB version shows *proc* instead of *procB* from the source code) to see an infinite loop:

```
(gdb) disassemble 0x0000000000403214
Dump of assembler code for function procB:
   0x00000000004031fc <+0>:     stp     x29, x30, [sp, #-32]!
   0x0000000000403200 <+4>:     mov     x29, sp
   0x0000000000403204 <+8>:     ldr     x0, 0x403220
   0x0000000000403208 <+12>:    str     x0, [x29, #24]
   0x000000000040320c <+16>:    ldr     d0, [x29, #24]
   0x0000000000403210 <+20>:    bl      0x403424 <sqrt>
   0x0000000000403214 <+24>:    str     d0, [x29, #24]
   0x0000000000403218 <+28>:    b       0x40320c <procB+16>
End of assembler dump.
```

Exercise A3 (A64, WinDbg Preview)

Goal: Learn how to identify spiking threads.

Patterns: Active Thread; Spiking Thread.

1. The application **App3** was consuming 100% CPU (from **top** command output):

```
$ top
top - 19:59:39 up 31 days, 19:09,  1 user,  load average: 1.00, 0.72, 0.34
Tasks: 184 total,   1 running, 128 sleeping,   0 stopped,   0 zombie
%Cpu(s): 25.1 us,  0.0 sy,  0.0 ni, 74.8 id,  0.0 wa,  0.1 hi,  0.0 si,  0.0 st
KiB Mem : 23799872 total, 19518400 free,   816064 used,  3465408 buff/cache
KiB Swap:  8388544 total,  8388544 free,        0 used. 19342592 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
21335 opc       20   0   42752     64      0 S  99.7  0.0   6:08.44 App3
21348 opc       20   0  119360   8000   3776 R   0.3  0.0   0:00.37 top
    1 root      20   0  165888  14976   7488 S   0.0  0.1   5:44.10 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:02.24 kthreadd
    3 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 rcu_par_gp
    6 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/0:0H-kb
    8 root       0 -20       0      0      0 I   0.0  0.0   0:00.04 mm_percpu_wq
    9 root      20   0       0      0      0 S   0.0  0.0   0:27.66 ksoftirqd/0
   10 root      20   0       0      0      0 I   0.0  0.0   9:59.60 rcu_sched
   11 root      rt   0       0      0      0 S   0.0  0.0   0:31.70 migration/0
   13 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/0
   14 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/1
   15 root      rt   0       0      0      0 S   0.0  0.0   0:34.69 migration/1
   16 root      20   0       0      0      0 S   0.0  0.0   0:23.54 ksoftirqd/1
   18 root       0 -20       0      0      0 I   0.0  0.0   0:31.65 kworker/1:0H-kb
   19 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/1:1H
   20 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/2
   21 root      rt   0       0      0      0 S   0.0  0.0   0:31.60 migration/2
   22 root      20   0       0      0      0 S   0.0  0.0   0:23.26 ksoftirqd/2
   24 root       0 -20       0      0      0 I   0.0  0.0   0:25.78 kworker/2:0H-kb
   25 root      20   0       0      0      0 S   0.0  0.0   0:00.00 cpuhp/3
```

Its core dump was saved using **gcore**:

```
~/ALCDA2/A64/App3$ gcore -o App3.core 21335
[New LWP 21340]
[New LWP 21339]
[New LWP 21338]
[New LWP 21337]
[New LWP 21336]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
0x0000000000414364 in nanosleep ()
Saved corefile App3.core.21335
[Inferior 1 (process 21335) detached]
```

2. Launch WinDbg Preview.

3. Load *App3.core.21335* dump file from the A64\App3 folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App3\App3.core.21335]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
*** WARNING: Unable to verify timestamp for App3
App3+0x14364:
00000000`00414364 d4000001 svc         #0
```

4. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App3\App3.log
Opened log file 'C:\ALCDA2\A64\App3\App3.log'
```

5. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App3\
Symbol search path is: srv*;C:\ALCDA2\A64\App3\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app3\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App3\
*** WARNING: Unable to verify timestamp for App3

0:000> .reload
..
*** WARNING: Unable to verify timestamp for App3

************* Symbol Loading Error Summary **************
Module name            Error
App3                   The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

6. List all thread stack traces to identify active threads:

```
0:000> ~*k

Unable to get thread data for thread 0
.  0  Id: 5357.5358 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`8dbde5f0 00000000`0042ca74     App3!_libc_nanosleep+0x24
01 0000fffc`8dbde630 00000000`00403238     App3!sleep+0x110
02 0000fffc`8dbde820 00000000`0040324c     App3!bar_one+0x10
03 0000fffc`8dbde830 00000000`00403264     App3!foo_one+0xc
04 0000fffc`8dbde840 00000000`0040c5e4     App3!thread_one+0x10
05 0000fffc`8dbde860 00000000`00431920     App3!start_thread+0xb4
06 0000fffc`8dbde990 ffffffff`ffffffff     App3!thread_start+0x30
07 0000fffc`8dbde990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 1
   1  Id: 5357.5359 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`8d3ce5f0 00000000`0042ca74     App3!_libc_nanosleep+0x24
01 0000fffc`8d3ce630 00000000`00403280     App3!sleep+0x110
02 0000fffc`8d3ce820 00000000`00403294     App3!bar_two+0x10
03 0000fffc`8d3ce830 00000000`004032ac     App3!foo_two+0xc
04 0000fffc`8d3ce840 00000000`0040c5e4     App3!thread_two+0x10
05 0000fffc`8d3ce860 00000000`00431920     App3!start_thread+0xb4
06 0000fffc`8d3ce990 ffffffff`ffffffff     App3!thread_start+0x30
07 0000fffc`8d3ce990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 2
   2  Id: 5357.535a Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`8cbbe5e0 00000000`0042ca74     App3!_libc_nanosleep+0x24
01 0000fffc`8cbbe620 00000000`004031f8     App3!sleep+0x110
02 0000fffc`8cbbe810 00000000`004032c4     App3!procA+0x10
03 0000fffc`8cbbe820 00000000`004032d8     App3!bar_three+0xc
04 0000fffc`8cbbe830 00000000`004032f0     App3!foo_three+0xc
05 0000fffc`8cbbe840 00000000`0040c5e4     App3!thread_three+0x10
06 0000fffc`8cbbe860 00000000`00431920     App3!start_thread+0xb4
07 0000fffc`8cbbe990 ffffffff`ffffffff     App3!thread_start+0x30
08 0000fffc`8cbbe990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 3
   3  Id: 5357.535b Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`8c3ae5f0 00000000`0042ca74     App3!_libc_nanosleep+0x24
01 0000fffc`8c3ae630 00000000`0040330c     App3!sleep+0x110
02 0000fffc`8c3ae820 00000000`00403320     App3!bar_four+0x10
03 0000fffc`8c3ae830 00000000`00403338     App3!foo_four+0xc
04 0000fffc`8c3ae840 00000000`0040c5e4     App3!thread_four+0x10
05 0000fffc`8c3ae860 00000000`00431920     App3!start_thread+0xb4
06 0000fffc`8c3ae990 ffffffff`ffffffff     App3!thread_start+0x30
07 0000fffc`8c3ae990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 4
   4  Id: 5357.535c Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`8bb9e7f0 00000000`00403214     App3!_sqrt_finite+0x148
01 0000fffc`8bb9e800 00000000`00403350     App3!procB+0x18
02 0000fffc`8bb9e820 00000000`00403364     App3!bar_five+0xc
03 0000fffc`8bb9e830 00000000`0040337c     App3!foo_five+0xc
```

```
04 0000fffc`8bb9e840 00000000`0040c5e4     App3!thread_five+0x10
05 0000fffc`8bb9e860 00000000`00431920     App3!start_thread+0xb4
06 0000fffc`8bb9e990 ffffffff`ffffffff     App3!thread_start+0x30
07 0000fffc`8bb9e990 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 5
   5  Id: 5357.5357 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`d4ed8dd0 00000000`0042ca74     App3!_libc_nanosleep+0x24
01 0000ffff`d4ed8e10 00000000`00403418     App3!sleep+0x110
02 0000ffff`d4ed9000 00000000`004165fc     App3!main+0x90
03 0000ffff`d4ed9050 00000000`00403090     App3!_libc_start_main+0x304
04 0000ffff`d4ed91b0 00000000`00000000     App3!start+0x4c
```

7. Switch to the active thread #4:

```
0:000> ~4s
App3!_sqrt_finite+0x148:
00000000`00408400 1e602030 fcmpe       d1,d0

0:004> k
 # Child-SP          RetAddr               Call Site
00 0000fffc`8bb9e7f0 00000000`00403214     App3!_sqrt_finite+0x148
01 0000fffc`8bb9e800 00000000`00403350     App3!procB+0x18
02 0000fffc`8bb9e820 00000000`00403364     App3!bar_five+0xc
03 0000fffc`8bb9e830 00000000`0040337c     App3!foo_five+0xc
04 0000fffc`8bb9e840 00000000`0040c5e4     App3!thread_five+0x10
05 0000fffc`8bb9e860 00000000`00431920     App3!start_thread+0xb4
06 0000fffc`8bb9e990 ffffffff`ffffffff     App3!thread_start+0x30
07 0000fffc`8bb9e990 00000000`00000000     0xffffffff`ffffffff
```

Note: We see that the current instruction is normal, related to floating-point operations.

8. Disassemble the return address for the *procB* function to see an infinite loop:

```
0:004> uf 00000000`00403214
App3!procB+0x10:
00000000`0040320c fd400fa0 ldr         d0,[fp,#0x18]
00000000`00403210 94000085 bl          App3!sqrt (00000000`00403424)

App3!procB+0x18:
00000000`00403214 fd000fa0 str         d0,[fp,#0x18]
00000000`00403218 17fffffd b           App3!procB+0x10 (00000000`0040320c)  Branch
```

10. We close logging before exiting WinDbg Preview:

```
0:004> .logclose
Closing open log file 'C:\ALCDA2\A64\App3\App3.log'
```

Exercise A4

- **Goal:** Learn how to identify heap regions and heap corruption
- **Patterns:** Dynamic Memory Corruption (Process Heap); Regular Data
- \ALCDA-Dumps\Exercise-A4-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A4-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A4-A64-WinDbg.pdf

Exercise A4 (x64, GDB)

Goal: Learn how to identify heap regions and heap corruption.

Patterns: Dynamic Memory Corruption (Process Heap); Regular Data.

1. Load *core.App4* dump file and *App4* executable from the x64/App4 directory:

```
~/ALCDA2/x64/App4$ gdb -c core.App4 -se App4
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App4...done.
[New LWP 4304]
[New LWP 4303]
[New LWP 4302]
[New LWP 4301]
[New LWP 4305]
[New LWP 4306]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App4'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x000000000041a906 in malloc ()
[Current thread is 1 (Thread 0x7f8d66b3a700 (LWP 4304))]
```

2. List threads:

```
(gdb) info threads
* 1    Thread 0x7f8d66b3a700 (LWP 4304) 0x000000000041a906 in malloc ()
  2    Thread 0x7f8d6733b700 (LWP 4303) 0x0000000000441d00 in nanosleep ()
  3    Thread 0x7f8d67b3c700 (LWP 4302) 0x0000000000441d00 in nanosleep ()
  4    Thread 0x124f880 (LWP 4301)      0x0000000000441d00 in nanosleep ()
  5    Thread 0x7f8d66339700 (LWP 4305) 0x0000000000441d00 in nanosleep ()
  6    Thread 0x7f8d65b38700 (LWP 4306) 0x0000000000441d00 in nanosleep ()
```

3. The identified problem thread #1 is the current thread. List its stack trace:

```
(gdb) bt
#0  0x000000000041a906 in malloc ()
#1  0x0000000000401e24 in proc ()
#2  0x0000000000401f2d in bar_three ()
#3  0x0000000000401f3e in foo_three ()
#4  0x0000000000401f57 in thread_three ()
#5  0x00000000004033c3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000044455f in clone ()
```

4. We see that the segmentation fault happened internally in the *malloc* function when *proc* was allocating heap memory. Disassemble the *proc* function:

```
(gdb) disassemble proc
Dump of assembler code for function proc:
   0x0000000000401bad <+0>:     push   %rbp
   0x0000000000401bae <+1>:     mov    %rsp,%rbp
   0x0000000000401bb1 <+4>:     sub    $0x40,%rsp
   0x0000000000401bb5 <+8>:     mov    $0x1,%edi
   0x0000000000401bba <+13>:    callq  0x441c50 <sleep>
   0x0000000000401bbf <+18>:    mov    $0x100,%edi
   0x0000000000401bc4 <+23>:    callq  0x41a7c0 <malloc>
   0x0000000000401bc9 <+28>:    mov    %rax,-0x8(%rbp)
   0x0000000000401bcd <+32>:    mov    $0x100,%edi
   0x0000000000401bd2 <+37>:    callq  0x41a7c0 <malloc>
   0x0000000000401bd7 <+42>:    mov    %rax,-0x10(%rbp)
   0x0000000000401bdb <+46>:    mov    $0x100,%edi
   0x0000000000401be0 <+51>:    callq  0x41a7c0 <malloc>
   0x0000000000401be5 <+56>:    mov    %rax,-0x18(%rbp)
   0x0000000000401be9 <+60>:    mov    $0x100,%edi
   0x0000000000401bee <+65>:    callq  0x41a7c0 <malloc>
   0x0000000000401bf3 <+70>:    mov    %rax,-0x20(%rbp)
   0x0000000000401bf7 <+74>:    mov    $0x100,%edi
   0x0000000000401bfc <+79>:    callq  0x41a7c0 <malloc>
   0x0000000000401c01 <+84>:    mov    %rax,-0x28(%rbp)
   0x0000000000401c05 <+88>:    mov    $0x100,%edi
   0x0000000000401c0a <+93>:    callq  0x41a7c0 <malloc>
   0x0000000000401c0f <+98>:    mov    %rax,-0x30(%rbp)
   0x0000000000401c13 <+102>:   mov    $0x100,%edi
   0x0000000000401c18 <+107>:   callq  0x41a7c0 <malloc>
   0x0000000000401c1d <+112>:   mov    %rax,-0x38(%rbp)
   0x0000000000401c21 <+116>:   mov    -0x30(%rbp),%rax
   0x0000000000401c25 <+120>:   mov    %rax,%rdi
   0x0000000000401c28 <+123>:   callq  0x41ae00 <free>
   0x0000000000401c2d <+128>:   mov    -0x20(%rbp),%rax
   0x0000000000401c31 <+132>:   mov    %rax,%rdi
   0x0000000000401c34 <+135>:   callq  0x41ae00 <free>
   0x0000000000401c39 <+140>:   mov    -0x10(%rbp),%rax
   0x0000000000401c3d <+144>:   mov    %rax,%rdi
   0x0000000000401c40 <+147>:   callq  0x41ae00 <free>
   0x0000000000401c45 <+152>:   mov    -0x10(%rbp),%rax
   0x0000000000401c49 <+156>:   movabs $0x7243206f6c6c6548,%rdx
   0x0000000000401c53 <+166>:   movabs $0x6548202132687361,%rcx
   0x0000000000401c5d <+176>:   mov    %rdx,(%rax)
   0x0000000000401c60 <+179>:   mov    %rcx,0x8(%rax)
   0x0000000000401c64 <+183>:   movabs $0x73617243206f6c6c,%rsi
   0x0000000000401c6e <+193>:   movabs $0x6c6c654820213268,%rdi
   0x0000000000401c78 <+203>:   mov    %rsi,0x10(%rax)
   0x0000000000401c7c <+207>:   mov    %rdi,0x18(%rax)
```

```
   0x0000000000401c80 <+211>:   movabs $0x326873617243206f,%rdx
   0x0000000000401c8a <+221>:   movabs $0x206f6c6c65482021,%rcx
   0x0000000000401c94 <+231>:   mov    %rdx,0x20(%rax)
   0x0000000000401c98 <+235>:   mov    %rcx,0x28(%rax)
   0x0000000000401c9c <+239>:   movabs $0x2021326873617243,%rsi
   0x0000000000401ca6 <+249>:   movabs $0x7243206f6c6c6548,%rdi
   0x0000000000401cb0 <+259>:   mov    %rsi,0x30(%rax)
   0x0000000000401cb4 <+263>:   mov    %rdi,0x38(%rax)
   0x0000000000401cb8 <+267>:   movl   $0x32687361,0x40(%rax)
   0x0000000000401cbf <+274>:   movw   $0x21,0x44(%rax)
   0x0000000000401cc5 <+280>:   mov    -0x20(%rbp),%rax
   0x0000000000401cc9 <+284>:   movabs $0x7243206f6c6c6548,%rdx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401cd3 <+294>:   movabs $0x6548202134687361,%rcx
   0x0000000000401cdd <+304>:   mov    %rdx,(%rax)
   0x0000000000401ce0 <+307>:   mov    %rcx,0x8(%rax)
   0x0000000000401ce4 <+311>:   movabs $0x73617243206f6c6c,%rsi
   0x0000000000401cee <+321>:   movabs $0x6c6c654820213468,%rdi
   0x0000000000401cf8 <+331>:   mov    %rsi,0x10(%rax)
   0x0000000000401cfc <+335>:   mov    %rdi,0x18(%rax)
   0x0000000000401d00 <+339>:   movabs $0x346873617243206f,%rdx
   0x0000000000401d0a <+349>:   movabs $0x206f6c6c65482021,%rcx
   0x0000000000401d14 <+359>:   mov    %rdx,0x20(%rax)
   0x0000000000401d18 <+363>:   mov    %rcx,0x28(%rax)
   0x0000000000401d1c <+367>:   movabs $0x2021346873617243,%rsi
   0x0000000000401d26 <+377>:   movabs $0x7243206f6c6c6548,%rdi
   0x0000000000401d30 <+387>:   mov    %rsi,0x30(%rax)
   0x0000000000401d34 <+391>:   mov    %rdi,0x38(%rax)
   0x0000000000401d38 <+395>:   movabs $0x6548202134687361,%rdx
   0x0000000000401d42 <+405>:   movabs $0x73617243206f6c6c,%rcx
   0x0000000000401d4c <+415>:   mov    %rdx,0x40(%rax)
   0x0000000000401d50 <+419>:   mov    %rcx,0x48(%rax)
   0x0000000000401d54 <+423>:   movl   $0x213468,0x50(%rax)
   0x0000000000401d5b <+430>:   mov    -0x30(%rbp),%rax
   0x0000000000401d5f <+434>:   movabs $0x7243206f6c6c6548,%rsi
   0x0000000000401d69 <+444>:   movabs $0x6548202136687361,%rdi
   0x0000000000401d73 <+454>:   mov    %rsi,(%rax)
   0x0000000000401d76 <+457>:   mov    %rdi,0x8(%rax)
   0x0000000000401d7a <+461>:   movabs $0x73617243206f6c6c,%rdx
   0x0000000000401d84 <+471>:   movabs $0x6c6c654820213668,%rcx
   0x0000000000401d8e <+481>:   mov    %rdx,0x10(%rax)
   0x0000000000401d92 <+485>:   mov    %rcx,0x18(%rax)
   0x0000000000401d96 <+489>:   movabs $0x366873617243206f,%rsi
   0x0000000000401da0 <+499>:   movabs $0x206f6c6c65482021,%rdi
   0x0000000000401daa <+509>:   mov    %rsi,0x20(%rax)
   0x0000000000401dae <+513>:   mov    %rdi,0x28(%rax)
   0x0000000000401db2 <+517>:   movabs $0x2021366873617243,%rdx
   0x0000000000401dbc <+527>:   movabs $0x7243206f6c6c6548,%rcx
   0x0000000000401dc6 <+537>:   mov    %rdx,0x30(%rax)
   0x0000000000401dca <+541>:   mov    %rcx,0x38(%rax)
   0x0000000000401dce <+545>:   movabs $0x6548202136687361,%rsi
   0x0000000000401dd8 <+555>:   movabs $0x73617243206f6c6c,%rdi
   0x0000000000401de2 <+565>:   mov    %rsi,0x40(%rax)
   0x0000000000401de6 <+569>:   mov    %rdi,0x48(%rax)
   0x0000000000401dea <+573>:   movabs $0x6c6c654820213668,%rdx
   0x0000000000401df4 <+583>:   movabs $0x366873617243206f,%rcx
   0x0000000000401dfe <+593>:   mov    %rdx,0x50(%rax)
   0x0000000000401e02 <+597>:   mov    %rcx,0x58(%rax)
   0x0000000000401e06 <+601>:   movw   $0x21,0x60(%rax)
   0x0000000000401e0c <+607>:   mov    $0x100,%edi
```

```
   0x0000000000401e11 <+612>:   callq  0x41a7c0 <malloc>
   0x0000000000401e16 <+617>:   mov    %rax,-0x10(%rbp)
   0x0000000000401e1a <+621>:   mov    $0x100,%edi
   0x0000000000401e1f <+626>:   callq  0x41a7c0 <malloc>
   0x0000000000401e24 <+631>:   mov    %rax,-0x20(%rbp)
   0x0000000000401e28 <+635>:   mov    $0x100,%edi
   0x0000000000401e2d <+640>:   callq  0x41a7c0 <malloc>
   0x0000000000401e32 <+645>:   mov    %rax,-0x30(%rbp)
   0x0000000000401e36 <+649>:   mov    $0x12c,%edi
   0x0000000000401e3b <+654>:   callq  0x441c50 <sleep>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401e40 <+659>:   mov    -0x38(%rbp),%rax
   0x0000000000401e44 <+663>:   mov    %rax,%rdi
   0x0000000000401e47 <+666>:   callq  0x41ae00 <free>
   0x0000000000401e4c <+671>:   mov    -0x30(%rbp),%rax
   0x0000000000401e50 <+675>:   mov    %rax,%rdi
   0x0000000000401e53 <+678>:   callq  0x41ae00 <free>
   0x0000000000401e58 <+683>:   mov    -0x28(%rbp),%rax
   0x0000000000401e5c <+687>:   mov    %rax,%rdi
   0x0000000000401e5f <+690>:   callq  0x41ae00 <free>
   0x0000000000401e64 <+695>:   mov    -0x20(%rbp),%rax
   0x0000000000401e68 <+699>:   mov    %rax,%rdi
   0x0000000000401e6b <+702>:   callq  0x41ae00 <free>
   0x0000000000401e70 <+707>:   mov    -0x18(%rbp),%rax
   0x0000000000401e74 <+711>:   mov    %rax,%rdi
   0x0000000000401e77 <+714>:   callq  0x41ae00 <free>
   0x0000000000401e7c <+719>:   mov    -0x10(%rbp),%rax
   0x0000000000401e80 <+723>:   mov    %rax,%rdi
   0x0000000000401e83 <+726>:   callq  0x41ae00 <free>
   0x0000000000401e88 <+731>:   mov    -0x8(%rbp),%rax
   0x0000000000401e8c <+735>:   mov    %rax,%rdi
   0x0000000000401e8f <+738>:   callq  0x41ae00 <free>
   0x0000000000401e94 <+743>:   mov    $0xffffffff,%edi
   0x0000000000401e99 <+748>:   callq  0x441c50 <sleep>
   0x0000000000401e9e <+753>:   nop
   0x0000000000401e9f <+754>:   leaveq
   0x0000000000401ea0 <+755>:   retq
End of assembler dump.
```

Note: We see that before the problem *malloc* call, there were three buffer writes to memory addresses pointed to by values located at the following addresses: rbp-0x10, rbp-0x20, and rbp-0x30 (highlighted in red in disassembly). However, before buffer writes, there were *free* function calls with values located at the same addresses: rbp-0x30, rbp-0x20, and rbp-0x10 (highlighted in blue in disassembly). Therefore, we see “write after free” behavior.

5. We have the standard function prolog (highlighted in green in disassembly). Switch to stack frame #1 to check the addresses, their values, and memory contents they point to:

```
(gdb) frame 1
#1  0x0000000000401e24 in proc () at pthread_create.c:688
688     in pthread_create.c

(gdb) x/gx $rbp-0x10
0x7f8d66b39d60: 0x00007f8d60000c30

(gdb) x/s 0x00007f8d60000c30
0x7f8d60000c30: "Hello Cr"

(gdb) x/gx $rbp-0x20
0x7f8d66b39d50: 0x00007f8d60000e50
```

```
(gdb) x/s 0x00007f8d60000e50
0x7f8d60000e50: "Hello Crash4! Hello Crash4! Hello Crash4! Hello Crash4! Hello Crash4! Hello
Crash4!"
```

6. We know the addresses passed to heap management functions, for example, 0x00007f8d60000xxx. Find the heap region in the section list:

```
(gdb) maintenance info sections
Exec file:
    `/home/coredump/ALCDA2/x64/App4/App4', file type elf64-x86-64.
 [0]     0x00400200->0x00400220 at 0x00000200: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]     0x00400220->0x00400244 at 0x00000220: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]     0x00400248->0x004004d0 at 0x00000248: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]     0x00401000->0x00401017 at 0x00001000: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
 [4]     0x00401018->0x004010f0 at 0x00001018: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
 [5]     0x004010f0->0x004936c0 at 0x000010f0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]     0x004936c0->0x00494267 at 0x000936c0: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]     0x00494268->0x00494271 at 0x00094268: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [8]     0x00495000->0x004af73c at 0x00095000: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
 [9]     0x004af740->0x004bbb70 at 0x000af740: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]     0x004bbb70->0x004bbc1c at 0x000bbb70: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]     0x004bd0b0->0x004bd0d8 at 0x000bc0b0: .tdata ALLOC LOAD DATA HAS_CONTENTS
 [12]     0x004bd0d8->0x004bd120 at 0x000bc0d8: .tbss ALLOC
 [13]     0x004bd0d8->0x004bd0e0 at 0x000bc0d8: .preinit_array ALLOC LOAD DATA HAS_CONTENTS
 [14]     0x004bd0e0->0x004bd0f0 at 0x000bc0e0: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [15]     0x004bd0f0->0x004bd100 at 0x000bc0f0: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [16]     0x004bd100->0x004bfef4 at 0x000bc100: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [17]     0x004bfef8->0x004c0000 at 0x000beef8: .got ALLOC LOAD DATA HAS_CONTENTS
 [18]     0x004c0000->0x004c00f0 at 0x000bf000: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [19]     0x004c0100->0x004c1c30 at 0x000bf100: .data ALLOC LOAD DATA HAS_CONTENTS
 [20]     0x004c1c30->0x004c1c90 at 0x000c0c30: __libc_subfreeres ALLOC LOAD DATA HAS_CONTENTS
 [21]     0x004c1ca0->0x004c2408 at 0x000c0ca0: __libc_IO_vtables ALLOC LOAD DATA HAS_CONTENTS
 [22]     0x004c2408->0x004c2410 at 0x000c1408: __libc_atexit ALLOC LOAD DATA HAS_CONTENTS
 [23]     0x004c2420->0x004c8528 at 0x000c1410: .bss ALLOC
 [24]     0x004c8528->0x004c8558 at 0x000c1410: __libc_freeres_ptrs ALLOC
 [25]     0x00000000->0x00000038 at 0x000c1410: .comment READONLY HAS_CONTENTS
 [26]     0x00000000->0x00000420 at 0x000c1450: .debug_aranges READONLY HAS_CONTENTS
 [27]     0x00000000->0x000372ad at 0x000c1870: .debug_info READONLY HAS_CONTENTS
 [28]     0x00000000->0x000057e8 at 0x000f8b1d: .debug_abbrev READONLY HAS_CONTENTS
 [29]     0x00000000->0x0000aa2b at 0x000fe305: .debug_line READONLY HAS_CONTENTS
 [30]     0x00000000->0x00004d08 at 0x00108d30: .debug_str READONLY HAS_CONTENTS
 [31]     0x00000000->0x0000d4b8 at 0x0010da38: .debug_loc READONLY HAS_CONTENTS
 [32]     0x00000000->0x000024c0 at 0x0011aef0: .debug_ranges READONLY HAS_CONTENTS
Core file:
    `/home/coredump/ALCDA2/x64/App4/core.App4', file type elf64-x86-64.
 [0]     0x00000000->0x00002c60 at 0x00000510: note0 READONLY HAS_CONTENTS
 [1]     0x00000000->0x000000d8 at 0x00000594: .reg/4304 HAS_CONTENTS
 [2]     0x00000000->0x000000d8 at 0x00000594: .reg HAS_CONTENTS
 [3]     0x00000000->0x00000080 at 0x00000724: .note.linuxcore.siginfo/4304 HAS_CONTENTS
 [4]     0x00000000->0x00000080 at 0x00000724: .note.linuxcore.siginfo HAS_CONTENTS
 [5]     0x00000000->0x00000140 at 0x000007b8: .auxv HAS_CONTENTS
 [6]     0x00000000->0x00000100 at 0x0000090c: .note.linuxcore.file/4304 HAS_CONTENTS
 [7]     0x00000000->0x00000100 at 0x0000090c: .note.linuxcore.file HAS_CONTENTS
 [8]     0x00000000->0x00000200 at 0x00000a20: .reg2/4304 HAS_CONTENTS
 [9]     0x00000000->0x00000200 at 0x00000a20: .reg2 HAS_CONTENTS
 [10]     0x00000000->0x00000340 at 0x00000c34: .reg-xstate/4304 HAS_CONTENTS
 [11]     0x00000000->0x00000340 at 0x00000c34: .reg-xstate HAS_CONTENTS
 [12]     0x00000000->0x000000d8 at 0x00000ff8: .reg/4303 HAS_CONTENTS
 [13]     0x00000000->0x00000200 at 0x000010ec: .reg2/4303 HAS_CONTENTS
 [14]     0x00000000->0x00000340 at 0x00001300: .reg-xstate/4303 HAS_CONTENTS
 [15]     0x00000000->0x000000d8 at 0x000016c4: .reg/4302 HAS_CONTENTS
 [16]     0x00000000->0x00000200 at 0x000017b8: .reg2/4302 HAS_CONTENTS
 [17]     0x00000000->0x00000340 at 0x000019cc: .reg-xstate/4302 HAS_CONTENTS
 [18]     0x00000000->0x000000d8 at 0x00001d90: .reg/4301 HAS_CONTENTS
 [19]     0x00000000->0x00000200 at 0x00001e84: .reg2/4301 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
```

```
[20]     0x00000000->0x00000340 at 0x00002098: .reg-xstate/4301 HAS_CONTENTS
[21]     0x00000000->0x000000d8 at 0x0000245c: .reg/4305 HAS_CONTENTS
[22]     0x00000000->0x00000200 at 0x00002550: .reg2/4305 HAS_CONTENTS
[23]     0x00000000->0x00000340 at 0x00002764: .reg-xstate/4305 HAS_CONTENTS
[24]     0x00000000->0x000000d8 at 0x00002b28: .reg/4306 HAS_CONTENTS
[25]     0x00000000->0x00000200 at 0x00002c1c: .reg2/4306 HAS_CONTENTS
[26]     0x00000000->0x00000340 at 0x00002e30: .reg-xstate/4306 HAS_CONTENTS
[27]     0x00400000->0x00401000 at 0x00004000: load1 ALLOC LOAD READONLY HAS_CONTENTS
[28]     0x00401000->0x00401000 at 0x00005000: load2 ALLOC READONLY CODE
[29]     0x00495000->0x00495000 at 0x00005000: load3 ALLOC READONLY
[30]     0x004bd000->0x004c3000 at 0x00005000: load4 ALLOC LOAD HAS_CONTENTS
[31]     0x004c3000->0x004c9000 at 0x0000b000: load5 ALLOC LOAD HAS_CONTENTS
[32]     0x0124f000->0x01272000 at 0x00011000: load6 ALLOC LOAD HAS_CONTENTS
[33]     0x7f8d60000000->0x7f8d60021000 at 0x00034000: load7 ALLOC LOAD HAS_CONTENTS
[34]     0x7f8d60021000->0x7f8d60021000 at 0x00055000: load8 ALLOC READONLY
[35]     0x7f8d65338000->0x7f8d65338000 at 0x00055000: load9 ALLOC READONLY
[36]     0x7f8d65339000->0x7f8d65b39000 at 0x00055000: load10 ALLOC LOAD HAS_CONTENTS
[37]     0x7f8d65b39000->0x7f8d65b39000 at 0x00855000: load11 ALLOC READONLY
[38]     0x7f8d65b3a000->0x7f8d6633a000 at 0x00855000: load12 ALLOC LOAD HAS_CONTENTS
[39]     0x7f8d6633a000->0x7f8d6633a000 at 0x01055000: load13 ALLOC READONLY
[40]     0x7f8d6633b000->0x7f8d66b3b000 at 0x01055000: load14 ALLOC LOAD HAS_CONTENTS
[41]     0x7f8d66b3b000->0x7f8d66b3b000 at 0x01855000: load15 ALLOC READONLY
[42]     0x7f8d66b3c000->0x7f8d6733c000 at 0x01855000: load16 ALLOC LOAD HAS_CONTENTS
[43]     0x7f8d6733c000->0x7f8d6733c000 at 0x02055000: load17 ALLOC READONLY
[44]     0x7f8d6733d000->0x7f8d67b3d000 at 0x02055000: load18 ALLOC LOAD HAS_CONTENTS
[45]     0x7ffc80658000->0x7ffc80679000 at 0x02855000: load19 ALLOC LOAD HAS_CONTENTS
[46]     0x7ffc806ff000->0x7ffc80703000 at 0x02876000: load20 ALLOC LOAD READONLY HAS_CONTENTS
[47]     0x7ffc80703000->0x7ffc80704000 at 0x0287a000: load21 ALLOC LOAD READONLY CODE HAS_CONTENTS
```

7. Check the faulting instruction and the problem memory address:

```
(gdb) bt
#0  0x000000000041a906 in malloc ()
#1  0x0000000000401e24 in proc () at pthread_create.c:688
#2  0x0000000000401f2d in bar_three () at pthread_create.c:688
#3  0x0000000000401f3e in foo_three () at pthread_create.c:688
#4  0x0000000000401f57 in thread_three () at pthread_create.c:688
#5  0x00000000004033c3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000044455f in clone ()

(gdb) frame 0
#0  0x000000000041a906 in malloc ()

(gdb) x/i $rip
=> 0x41a906 <malloc+326>:       mov    (%rdx),%rsi

(gdb) x $rdx
   0x7243206f6c6c6548:  Cannot access memory at address 0x7243206f6c6c6548

(gdb) p (char[8])0x7243206f6c6c6548
$1 = "Hello Cr"
```

Note: We see that the “Hello Cr” fragment correlates with the “Hello Cr” buffer overwrite that we saw previously in step #5.

Exercise A4 (A64, GDB)

Goal: Learn how to identify heap regions and heap corruption.

Patterns: Dynamic Memory Corruption (Process Heap); Regular Data.

1. Load *core.8800* dump file and *App4* executable from the A64/App4 directory:

```
~/ALCDA2/A64/App4$ gdb -c core.8800 -se App4
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App4...
(No debugging symbols found in App4)

warning: Can't open file /home/opc/ALCDA2/App4/App4 during file-backed mapping note processing
[New LWP 8803]
[New LWP 8801]
[New LWP 8800]
[New LWP 8802]
[New LWP 8804]
[New LWP 8805]
Core was generated by `./App4'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x000000000041cbec in malloc ()
[Current thread is 1 (LWP 8803)]
```

2. Set logging to a file in case of lengthy output from some commands and set color highlighting off:

```
(gdb) set logging file App4.log

(gdb) set logging enabled on
Copying output to App4.log.
Copying debug output to App4.log.

(gdb) set style enabled off
```

3. List threads:

```
(gdb) info threads
* 1    LWP 8803          0x000000000041cbec in malloc ()
  2    LWP 8801          0x000000000040cb34 in nanosleep ()
  3    LWP 8800          0x000000000040cb34 in nanosleep ()
  4    LWP 8802          0x000000000040cb34 in nanosleep ()
  5    LWP 8804          0x000000000040cb34 in nanosleep ()
```

```
  6    LWP 8805          0x000000000040cb34 in nanosleep ()
```

4. The identified problem thread #1 is the current thread. List its stack trace:

```
(gdb) bt
#0  0x000000000041cbec in malloc ()
#1  0x0000000000403304 in proc ()
#2  0x0000000000403400 in bar_three ()
#3  0x0000000000403414 in foo_three ()
#4  0x000000000040342c in thread_three ()
#5  0x0000000000404db4 in start_thread ()
#6  0x0000000000429ce0 in thread_start ()
```

5. We see that the segmentation fault happened internally in the *malloc* function when *proc* was allocating heap memory. Disassemble the *proc* function:

```
(gdb) disassemble proc
Dump of assembler code for function proc:
   0x00000000004031e8 <+0>:     stp     x29, x30, [sp, #-80]!
   0x00000000004031ec <+4>:     mov     x29, sp
   0x00000000004031f0 <+8>:     mov     w0, #0x1                        // #1
   0x00000000004031f4 <+12>:    bl      0x424d24 <sleep>
   0x00000000004031f8 <+16>:    mov     x0, #0x100                      // #256
   0x00000000004031fc <+20>:    bl      0x41cb60 <malloc>
   0x0000000000403200 <+24>:    str     x0, [x29, #72]
   0x0000000000403204 <+28>:    mov     x0, #0x100                      // #256
   0x0000000000403208 <+32>:    bl      0x41cb60 <malloc>
   0x000000000040320c <+36>:    str     x0, [x29, #64]
   0x0000000000403210 <+40>:    mov     x0, #0x100                      // #256
   0x0000000000403214 <+44>:    bl      0x41cb60 <malloc>
   0x0000000000403218 <+48>:    str     x0, [x29, #56]
   0x000000000040321c <+52>:    mov     x0, #0x100                      // #256
   0x0000000000403220 <+56>:    bl      0x41cb60 <malloc>
   0x0000000000403224 <+60>:    str     x0, [x29, #48]
   0x0000000000403228 <+64>:    mov     x0, #0x100                      // #256
   0x000000000040322c <+68>:    bl      0x41cb60 <malloc>
   0x0000000000403230 <+72>:    str     x0, [x29, #40]
   0x0000000000403234 <+76>:    mov     x0, #0x100                      // #256
   0x0000000000403238 <+80>:    bl      0x41cb60 <malloc>
   0x000000000040323c <+84>:    str     x0, [x29, #32]
   0x0000000000403240 <+88>:    mov     x0, #0x100                      // #256
   0x0000000000403244 <+92>:    bl      0x41cb60 <malloc>
   0x0000000000403248 <+96>:    str     x0, [x29, #24]
   0x000000000040324c <+100>:   ldr     x0, [x29, #32]
   0x0000000000403250 <+104>:   bl      0x41d698 <free>
   0x0000000000403254 <+108>:   ldr     x0, [x29, #48]
   0x0000000000403258 <+112>:   bl      0x41d698 <free>
   0x000000000040325c <+116>:   ldr     x0, [x29, #64]
   0x0000000000403260 <+120>:   bl      0x41d698 <free>
   0x0000000000403264 <+124>:   ldr     x0, [x29, #64]
   0x0000000000403268 <+128>:   adrp    x1, 0x489000 <arena_thread_freeres+280>
   0x000000000040326c <+132>:   add     x1, x1, #0x360
   0x0000000000403270 <+136>:   ldp     x2, x3, [x1]
   0x0000000000403274 <+140>:   stp     x2, x3, [x0]
   0x0000000000403278 <+144>:   ldp     x2, x3, [x1, #16]
   0x000000000040327c <+148>:   stp     x2, x3, [x0, #16]
   0x0000000000403280 <+152>:   ldp     x2, x3, [x1, #32]
   0x0000000000403284 <+156>:   stp     x2, x3, [x0, #32]
   0x0000000000403288 <+160>:   ldp     x2, x3, [x1, #48]
   0x000000000040328c <+164>:   stp     x2, x3, [x0, #48]
```

```
   0x0000000000403290 <+168>:   ldr     w2, [x1, #64]
   0x0000000000403294 <+172>:   str     w2, [x0, #64]
   0x0000000000403298 <+176>:   ldrh    w1, [x1, #68]
   0x000000000040329c <+180>:   strh    w1, [x0, #68]
   0x00000000004032a0 <+184>:   ldr     x0, [x29, #48]
   0x00000000004032a4 <+188>:   adrp    x1, 0x489000 <arena_thread_freeres+280>
   0x00000000004032a8 <+192>:   add     x1, x1, #0x3a8
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004032ac <+196>:   ldp     x2, x3, [x1]
   0x00000000004032b0 <+200>:   stp     x2, x3, [x0]
   0x00000000004032b4 <+204>:   ldp     x2, x3, [x1, #16]
   0x00000000004032b8 <+208>:   stp     x2, x3, [x0, #16]
   0x00000000004032bc <+212>:   ldp     x2, x3, [x1, #32]
   0x00000000004032c0 <+216>:   stp     x2, x3, [x0, #32]
   0x00000000004032c4 <+220>:   ldp     x2, x3, [x1, #48]
   0x00000000004032c8 <+224>:   stp     x2, x3, [x0, #48]
   0x00000000004032cc <+228>:   ldp     x2, x3, [x1, #64]
   0x00000000004032d0 <+232>:   stp     x2, x3, [x0, #64]
   0x00000000004032d4 <+236>:   ldr     w1, [x1, #80]
   0x00000000004032d8 <+240>:   str     w1, [x0, #80]
   0x00000000004032dc <+244>:   ldr     x0, [x29, #32]
   0x00000000004032e0 <+248>:   adrp    x1, 0x489000 <arena_thread_freeres+280>
   0x00000000004032e4 <+252>:   add     x1, x1, #0x400
   0x00000000004032e8 <+256>:   mov     x2, #0x62                       // #98
   0x00000000004032ec <+260>:   bl      0x400280
   0x00000000004032f0 <+264>:   mov     x0, #0x100                      // #256
   0x00000000004032f4 <+268>:   bl      0x41cb60 <malloc>
   0x00000000004032f8 <+272>:   str     x0, [x29, #64]
   0x00000000004032fc <+276>:   mov     x0, #0x100                      // #256
   0x0000000000403300 <+280>:   bl      0x41cb60 <malloc>
   0x0000000000403304 <+284>:   str     x0, [x29, #48]
   0x0000000000403308 <+288>:   mov     x0, #0x100                      // #256
   0x000000000040330c <+292>:   bl      0x41cb60 <malloc>
   0x0000000000403310 <+296>:   str     x0, [x29, #32]
   0x0000000000403314 <+300>:   mov     w0, #0x12c                      // #300
   0x0000000000403318 <+304>:   bl      0x424d24 <sleep>
   0x000000000040331c <+308>:   ldr     x0, [x29, #24]
   0x0000000000403320 <+312>:   bl      0x41d698 <free>
   0x0000000000403324 <+316>:   ldr     x0, [x29, #32]
   0x0000000000403328 <+320>:   bl      0x41d698 <free>
   0x000000000040332c <+324>:   ldr     x0, [x29, #40]
   0x0000000000403330 <+328>:   bl      0x41d698 <free>
   0x0000000000403334 <+332>:   ldr     x0, [x29, #48]
   0x0000000000403338 <+336>:   bl      0x41d698 <free>
   0x000000000040333c <+340>:   ldr     x0, [x29, #56]
   0x0000000000403340 <+344>:   bl      0x41d698 <free>
   0x0000000000403344 <+348>:   ldr     x0, [x29, #64]
   0x0000000000403348 <+352>:   bl      0x41d698 <free>
   0x000000000040334c <+356>:   ldr     x0, [x29, #72]
   0x0000000000403350 <+360>:   bl      0x41d698 <free>
   0x0000000000403354 <+364>:   mov     w0, #0xffffffff                 // #-1
   0x0000000000403358 <+368>:   bl      0x424d24 <sleep>
   0x000000000040335c <+372>:   ldp     x29, x30, [sp], #80
   0x0000000000403360 <+376>:   ret
End of assembler dump.
```

Note: We see that before the problem *malloc* call, there were three buffer writes to memory addresses pointed to by values located at the following addresses: x29+64, x29+48, and x29+32 (highlighted in red in disassembly).

However, before buffer writes, there were *free* function calls with values located at the same addresses: x29+64, x29+48, and x29+32 (highlighted in blue in disassembly). Therefore, we see “write after free” behavior.

6. We have the standard function prolog (highlighted in green in disassembly). Switch to stack frame #1 to check the addresses, their values, and memory contents they point to:

```
(gdb) frame 1
#1  0x0000000000403304 in proc ()

(gdb) x/gx $x29+32
0xfffc03e5e7f0: 0x0000fffbfc001070

(gdb) x/s 0x0000fffbfc001070
0xfffbfc001070: "Hello Crash6! Hello Crash6! Hello Crash6! Hello Crash6! Hello Crash6! Hello
Crash6! Hello Crash6!"

(gdb) x/gx $x29+48
0xfffc03e5e800: 0x0000fffbfc000e50

(gdb) x/s 0x0000fffbfc000e50
0xfffbfc000e50: "Hello Crash4! Hello Crash4! Hello Crash4! Hello Crash4! Hello Crash4! Hello
Crash4!"
```

7. We know the addresses passed to heap management functions, for example, 0x0000fffbfc000xxx. Find the heap region in the section list:

```
(gdb) maintenance info sections
Exec file: `/home/ubuntu/ALCDA2/A64/App4/App4', file type elf64-littleaarch64.
 [0]      0x00400190->0x004001b0 at 0x00000190: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]      0x004001b0->0x004001d4 at 0x000001b0: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]      0x004001d8->0x00400250 at 0x000001d8: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]      0x00400250->0x00400264 at 0x00000250: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
 [4]      0x00400270->0x004002c0 at 0x00000270: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
 [5]      0x004002c0->0x00487218 at 0x000002c0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]      0x00487218->0x00488ee8 at 0x00087218: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]      0x00488ee8->0x00489338 at 0x00088ee8: __libc_thread_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [8]      0x00489338->0x00489348 at 0x00089338: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [9]      0x00489350->0x004a193d at 0x00089350: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]     0x004a193d->0x004a193e at 0x000a193d: .stapsdt.base ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]     0x004a1940->0x004a2078 at 0x000a1940: __libc_IO_vtables ALLOC LOAD READONLY DATA HAS_CONTENTS
 [12]     0x004a2078->0x004a20e0 at 0x000a2078: __libc_subfreeres ALLOC LOAD READONLY DATA HAS_CONTENTS
 [13]     0x004a20e0->0x004a20e8 at 0x000a20e0: __libc_atexit ALLOC LOAD READONLY DATA HAS_CONTENTS
 [14]     0x004a20e8->0x004a20f8 at 0x000a20e8: __libc_thread_subfreeres ALLOC LOAD READONLY DATA HAS_CONTENTS
 [15]     0x004a20f8->0x004b0734 at 0x000a20f8: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [16]     0x004b0734->0x004b08f1 at 0x000b0734: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [17]     0x004cfb20->0x004cfb48 at 0x000bfb20: .tdata ALLOC LOAD DATA HAS_CONTENTS
 [18]     0x004cfb48->0x004cfb98 at 0x000bfb48: .tbss ALLOC
 [19]     0x004cfb48->0x004cfb50 at 0x000bfb48: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [20]     0x004cfb50->0x004cfb60 at 0x000bfb50: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [21]     0x004cfb60->0x004cfb68 at 0x000bfb60: .jcr ALLOC LOAD DATA HAS_CONTENTS
 [22]     0x004cfb68->0x004cff24 at 0x000bfb68: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [23]     0x004cff28->0x004cffe8 at 0x000bff28: .got ALLOC LOAD DATA HAS_CONTENTS
 [24]     0x004cffe8->0x004d0028 at 0x000bffe8: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [25]     0x004d0030->0x004d1580 at 0x000c0030: .data ALLOC LOAD DATA HAS_CONTENTS
 [26]     0x004d1580->0x004d8050 at 0x000c1580: .bss ALLOC
 [27]     0x004d8050->0x004d8088 at 0x000c1580: __libc_freeres_ptrs ALLOC
 [28]     0x00000000->0x00000031 at 0x000c1580: .comment READONLY HAS_CONTENTS
 [29]     0x00000000->0x00001cb0 at 0x000c15b4: .note.stapsdt READONLY HAS_CONTENTS
Core file: `/home/ubuntu/ALCDA2/A64/App4/core.8800', file type elf64-littleaarch64.
 [0]      0x00000000->0x00002838 at 0x00000468: note0 READONLY HAS_CONTENTS
 [1]      0x00000000->0x00000110 at 0x000004ec: .reg/8803 HAS_CONTENTS
 [2]      0x00000000->0x00000110 at 0x000004ec: .reg HAS_CONTENTS
 [3]      0x00000000->0x00000080 at 0x000006b4: .note.linuxcore.siginfo/8803 HAS_CONTENTS
 [4]      0x00000000->0x00000080 at 0x000006b4: .note.linuxcore.siginfo HAS_CONTENTS
 [5]      0x00000000->0x00000160 at 0x00000748: .auxv HAS_CONTENTS
 [6]      0x00000000->0x00000076 at 0x000008bc: .note.linuxcore.file/8803 HAS_CONTENTS
 [7]      0x00000000->0x00000076 at 0x000008bc: .note.linuxcore.file HAS_CONTENTS
```

```
 [8]     0x00000000->0x00000210 at 0x00000948: .reg2/8803 HAS_CONTENTS
 [9]     0x00000000->0x00000210 at 0x00000948: .reg2 HAS_CONTENTS
 [10]    0x00000000->0x00000008 at 0x00000b6c: .reg-aarch-tls/8803 HAS_CONTENTS
 [11]    0x00000000->0x00000008 at 0x00000b6c: .reg-aarch-tls HAS_CONTENTS
 [12]    0x00000000->0x00000108 at 0x00000b88: .reg-aarch-hw-break/8803 HAS_CONTENTS
 [13]    0x00000000->0x00000108 at 0x00000b88: .reg-aarch-hw-break HAS_CONTENTS
 [14]    0x00000000->0x00000108 at 0x00000ca4: .reg-aarch-hw-watch/8803 HAS_CONTENTS
 [15]    0x00000000->0x00000108 at 0x00000ca4: .reg-aarch-hw-watch HAS_CONTENTS
 [16]    0x00000000->0x00000110 at 0x00000e48: .reg/8801 HAS_CONTENTS
 [17]    0x00000000->0x00000210 at 0x00000f74: .reg2/8801 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
 [18]    0x00000000->0x00000008 at 0x00001198: .reg-aarch-tls/8801 HAS_CONTENTS
 [19]    0x00000000->0x00000108 at 0x000011b4: .reg-aarch-hw-break/8801 HAS_CONTENTS
 [20]    0x00000000->0x00000108 at 0x000012d0: .reg-aarch-hw-watch/8801 HAS_CONTENTS
 [21]    0x00000000->0x00000110 at 0x00001474: .reg/8800 HAS_CONTENTS
 [22]    0x00000000->0x00000210 at 0x000015a0: .reg2/8800 HAS_CONTENTS
 [23]    0x00000000->0x00000008 at 0x000017c4: .reg-aarch-tls/8800 HAS_CONTENTS
 [24]    0x00000000->0x00000108 at 0x000017e0: .reg-aarch-hw-break/8800 HAS_CONTENTS
 [25]    0x00000000->0x00000108 at 0x000018fc: .reg-aarch-hw-watch/8800 HAS_CONTENTS
 [26]    0x00000000->0x00000110 at 0x00001aa0: .reg/8802 HAS_CONTENTS
 [27]    0x00000000->0x00000210 at 0x00001bcc: .reg2/8802 HAS_CONTENTS
 [28]    0x00000000->0x00000008 at 0x00001df0: .reg-aarch-tls/8802 HAS_CONTENTS
 [29]    0x00000000->0x00000108 at 0x00001e0c: .reg-aarch-hw-break/8802 HAS_CONTENTS
 [30]    0x00000000->0x00000108 at 0x00001f28: .reg-aarch-hw-watch/8802 HAS_CONTENTS
 [31]    0x00000000->0x00000110 at 0x000020cc: .reg/8804 HAS_CONTENTS
 [32]    0x00000000->0x00000210 at 0x000021f8: .reg2/8804 HAS_CONTENTS
 [33]    0x00000000->0x00000008 at 0x0000241c: .reg-aarch-tls/8804 HAS_CONTENTS
 [34]    0x00000000->0x00000108 at 0x00002438: .reg-aarch-hw-break/8804 HAS_CONTENTS
 [35]    0x00000000->0x00000108 at 0x00002554: .reg-aarch-hw-watch/8804 HAS_CONTENTS
 [36]    0x00000000->0x00000110 at 0x000026f8: .reg/8805 HAS_CONTENTS
 [37]    0x00000000->0x00000210 at 0x00002824: .reg2/8805 HAS_CONTENTS
 [38]    0x00000000->0x00000008 at 0x00002a48: .reg-aarch-tls/8805 HAS_CONTENTS
 [39]    0x00000000->0x00000108 at 0x00002a64: .reg-aarch-hw-break/8805 HAS_CONTENTS
 [40]    0x00000000->0x00000108 at 0x00002b80: .reg-aarch-hw-watch/8805 HAS_CONTENTS
 [41]    0x00400000->0x00410000 at 0x00010000: load1a ALLOC LOAD READONLY CODE HAS_CONTENTS
 [42]    0x00410000->0x004c0000 at 0x00020000: load1b ALLOC READONLY CODE
 [43]    0x004c0000->0x004e0000 at 0x00020000: load2 ALLOC LOAD HAS_CONTENTS
 [44]    0x31db0000->0x31df0000 at 0x00040000: load3 ALLOC LOAD HAS_CONTENTS
 [45]    0xfffbfc000000->0xfffbfc030000 at 0x00080000: load4 ALLOC LOAD HAS_CONTENTS
 [46]    0xfffbfc030000->0xfffc00000000 at 0x000b0000: load5 ALLOC READONLY
 [47]    0xfffc02630000->0xfffc02640000 at 0x000b0000: load6 ALLOC LOAD READONLY HAS_CONTENTS
 [48]    0xfffc02640000->0xfffc02e40000 at 0x000c0000: load7 ALLOC LOAD HAS_CONTENTS
 [49]    0xfffc02e40000->0xfffc02e50000 at 0x008c0000: load8 ALLOC LOAD READONLY HAS_CONTENTS
 [50]    0xfffc02e50000->0xfffc03650000 at 0x008d0000: load9 ALLOC LOAD HAS_CONTENTS
 [51]    0xfffc03650000->0xfffc03660000 at 0x010d0000: load10 ALLOC LOAD READONLY HAS_CONTENTS
 [52]    0xfffc03660000->0xfffc03e60000 at 0x010e0000: load11 ALLOC LOAD HAS_CONTENTS
 [53]    0xfffc03e60000->0xfffc03e70000 at 0x018e0000: load12 ALLOC LOAD READONLY HAS_CONTENTS
 [54]    0xfffc03e70000->0xfffc04670000 at 0x018f0000: load13 ALLOC LOAD HAS_CONTENTS
 [55]    0xfffc04670000->0xfffc04680000 at 0x020f0000: load14 ALLOC LOAD READONLY HAS_CONTENTS
 [56]    0xfffc04680000->0xfffc04e80000 at 0x02100000: load15 ALLOC LOAD HAS_CONTENTS
 [57]    0xfffc04e80000->0xfffc04e90000 at 0x02900000: load16 ALLOC LOAD READONLY HAS_CONTENTS
 [58]    0xfffc04e90000->0xfffc04ea0000 at 0x02910000: load17 ALLOC LOAD READONLY CODE HAS_CONTENTS
 [59]    0xffffd31d0000->0xffffd3200000 at 0x02920000: load18 ALLOC LOAD HAS_CONTENTS
```

8. Check the faulting instruction and the problem memory address:

```
(gdb) bt
#0  0x0000000000041cbec in malloc ()
#1  0x0000000000403304 in proc ()
#2  0x0000000000403400 in bar_three ()
#3  0x0000000000403414 in foo_three ()
#4  0x000000000040342c in thread_three ()
#5  0x0000000000404db4 in start_thread ()
#6  0x0000000000429ce0 in thread_start ()

(gdb) frame 0
#0  0x0000000000041cbec in malloc ()

(gdb) x/i $pc
=> 0x41cbec <malloc+140>:       ldr     x1, [x4]
```

```
(gdb) x $x4
   0x7243206f6c6c6548:  Cannot access memory at address 0x7243206f6c6c6548

(gdb) p (char[8])0x7243206f6c6c6548
$1 = "Hello Cr"
```

Note: We see that the “Hello Cr” fragment correlates with the “Hello Cr” buffer overwrite that we saw previously in step #6.

Exercise A4 (A64, WinDbg Preview)

Goal: Learn how to identify heap regions and heap corruption.

Patterns: Dynamic Memory Corruption (Process Heap); Regular Data.

1. Launch WinDbg Preview.

2. Load *core.8800* dump file from the A64\App4 folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App4\core.8800]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
(2260.2263): Signal SIGSEGV (Segmentation fault) code SEGV_MAPERR (Address not mapped to
object) at 0x43206f6c6c6548*** WARNING: Unable to verify timestamp for App4
App4+0x1cbec:
00000000`0041cbec ?? ???
```

3. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App4\App4.log
Opened log file 'C:\ALCDA2\A64\App4\App4.log'
```

4. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App4\
Symbol search path is: srv*;C:\ALCDA2\A64\App4\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app4\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App4\
*** WARNING: Unable to verify timestamp for App4

0:000> .reload
..
*** WARNING: Unable to verify timestamp for App4

************* Symbol Loading Error Summary **************
Module name            Error
```

```
App4                   The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

5. List all threads and their first frame:

```
0:000> ~*k 1

Unable to get thread data for thread 0
.  0  Id: 2260.2263 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`03e5e790 00000000`00403304     App4!malloc+0x8c

Unable to get thread data for thread 1
   1  Id: 2260.2261 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`04e7e5f0 00000000`00424e34     App4!_libc_nanosleep+0x24

Unable to get thread data for thread 2
   2  Id: 2260.2260 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`d31fe360 00000000`00424e34     App4!_libc_nanosleep+0x24

Unable to get thread data for thread 3
   3  Id: 2260.2262 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`0466e5f0 00000000`00424e34     App4!_libc_nanosleep+0x24

Unable to get thread data for thread 4
   4  Id: 2260.2264 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`0364e5f0 00000000`00424e34     App4!_libc_nanosleep+0x24

Unable to get thread data for thread 5
   5  Id: 2260.2265 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`02e3e5f0 00000000`00424e34     App4!_libc_nanosleep+0x24
```

6. List the current thread stack trace:

```
0:000> k
 # Child-SP          RetAddr               Call Site
00 0000fffc`03e5e790 00000000`00403304     App4!malloc+0x8c
01 0000fffc`03e5e7d0 00000000`00403400     App4!proc+0x11c
02 0000fffc`03e5e820 00000000`00403414     App4!bar_three+0xc
03 0000fffc`03e5e830 00000000`0040342c     App4!foo_three+0xc
04 0000fffc`03e5e840 00000000`00404db4     App4!thread_three+0x10
05 0000fffc`03e5e860 00000000`00429ce0     App4!start_thread+0xb4
06 0000fffc`03e5e990 ffffffff`ffffffff     App4!thread_start+0x30
07 0000fffc`03e5e990 00000000`00000000     0xffffffff`ffffffff
```

7. We see that the segmentation fault happened internally in the *malloc* function when *proc* was allocating heap memory. Disassemble the *proc* function:

```
0:000> uf proc
App4!proc:
00000000`004031e8 a9bb7bfd stp         fp,lr,[sp,#-0x50]!
00000000`004031ec 910003fd mov         fp,sp
00000000`004031f0 52800020 mov         w0,#1
00000000`004031f4 940086cc bl          App4!sleep (00000000`00424d24)
00000000`004031f8 d2802000 mov         x0,#0x100
00000000`004031fc 94006659 bl          App4!malloc (00000000`0041cb60)
00000000`00403200 f90027a0 str         x0,[fp,#0x48]
00000000`00403204 d2802000 mov         x0,#0x100
00000000`00403208 94006656 bl          App4!malloc (00000000`0041cb60)
00000000`0040320c f90023a0 str         x0,[fp,#0x40]
00000000`00403210 d2802000 mov         x0,#0x100
00000000`00403214 94006653 bl          App4!malloc (00000000`0041cb60)
00000000`00403218 f9001fa0 str         x0,[fp,#0x38]
00000000`0040321c d2802000 mov         x0,#0x100
00000000`00403220 94006650 bl          App4!malloc (00000000`0041cb60)
00000000`00403224 f9001ba0 str         x0,[fp,#0x30]
00000000`00403228 d2802000 mov         x0,#0x100
00000000`0040322c 9400664d bl          App4!malloc (00000000`0041cb60)
00000000`00403230 f90017a0 str         x0,[fp,#0x28]
00000000`00403234 d2802000 mov         x0,#0x100
00000000`00403238 9400664a bl          App4!malloc (00000000`0041cb60)
00000000`0040323c f90013a0 str         x0,[fp,#0x20]
00000000`00403240 d2802000 mov         x0,#0x100
00000000`00403244 94006647 bl          App4!malloc (00000000`0041cb60)
00000000`00403248 f9000fa0 str         x0,[fp,#0x18]
00000000`0040324c f94013a0 ldr         x0,[fp,#0x20]
00000000`00403250 94006912 bl          App4!_cfree (00000000`0041d698)
00000000`00403254 f9401ba0 ldr         x0,[fp,#0x30]
00000000`00403258 94006910 bl          App4!_cfree (00000000`0041d698)
00000000`0040325c f94023a0 ldr         x0,[fp,#0x40]
00000000`00403260 9400690e bl          App4!_cfree (00000000`0041d698)
00000000`00403264 f94023a0 ldr         x0,[fp,#0x40]
00000000`00403268 d0000421 adrp        x1,App4!arena_thread_freeres+0x118 (00000000`00489000)
00000000`0040326c 910d8021 add         x1,x1,#0x360
00000000`00403270 a9400c22 ldp         x2,x3,[x1]
00000000`00403274 a9000c02 stp         x2,x3,[x0]
00000000`00403278 a9410c22 ldp         x2,x3,[x1,#0x10]
00000000`0040327c a9010c02 stp         x2,x3,[x0,#0x10]
00000000`00403280 a9420c22 ldp         x2,x3,[x1,#0x20]
00000000`00403284 a9020c02 stp         x2,x3,[x0,#0x20]
00000000`00403288 a9430c22 ldp         x2,x3,[x1,#0x30]
00000000`0040328c a9030c02 stp         x2,x3,[x0,#0x30]
00000000`00403290 b9404022 ldr         w2,[x1,#0x40]
00000000`00403294 b9004002 str         w2,[x0,#0x40]
00000000`00403298 79408821 ldrh        w1,[x1,#0x44]
00000000`0040329c 79008801 strh        w1,[x0,#0x44]
00000000`004032a0 f9401ba0 ldr         x0,[fp,#0x30]
00000000`004032a4 d0000421 adrp        x1,App4!arena_thread_freeres+0x118 (00000000`00489000)
00000000`004032a8 910ea021 add         x1,x1,#0x3A8
00000000`004032ac a9400c22 ldp         x2,x3,[x1]
00000000`004032b0 a9000c02 stp         x2,x3,[x0]
00000000`004032b4 a9410c22 ldp         x2,x3,[x1,#0x10]
00000000`004032b8 a9010c02 stp         x2,x3,[x0,#0x10]
00000000`004032bc a9420c22 ldp         x2,x3,[x1,#0x20]
```

```
00000000`004032c0 a9020c02 stp         x2,x3,[x0,#0x20]
00000000`004032c4 a9430c22 ldp         x2,x3,[x1,#0x30]
00000000`004032c8 a9030c02 stp         x2,x3,[x0,#0x30]
00000000`004032cc a9440c22 ldp         x2,x3,[x1,#0x40]
00000000`004032d0 a9040c02 stp         x2,x3,[x0,#0x40]
00000000`004032d4 b9405021 ldr         w1,[x1,#0x50]
00000000`004032d8 b9005001 str         w1,[x0,#0x50]
00000000`004032dc f94013a0 ldr         x0,[fp,#0x20]
00000000`004032e0 d0000421 adrp        x1,App4!arena_thread_freeres+0x118 (00000000`00489000)
00000000`004032e4 91100021 add         x1,x1,#0x400
00000000`004032e8 d2800c42 mov         x2,#0x62
00000000`004032ec 97fff3e5 bl          App4!+0x10 (00000000`00400280)
00000000`004032f0 d2802000 mov         x0,#0x100
00000000`004032f4 9400661b bl          App4!malloc (00000000`0041cb60)
00000000`004032f8 f90023a0 str         x0,[fp,#0x40]
00000000`004032fc d2802000 mov         x0,#0x100
00000000`00403300 94006618 bl          App4!malloc (00000000`0041cb60)
00000000`00403304 f9001ba0 str         x0,[fp,#0x30]
00000000`00403308 d2802000 mov         x0,#0x100
00000000`0040330c 94006615 bl          App4!malloc (00000000`0041cb60)
00000000`00403310 f90013a0 str         x0,[fp,#0x20]
00000000`00403314 52802580 mov         w0,#0x12C
00000000`00403318 94008683 bl          App4!sleep (00000000`00424d24)
00000000`0040331c f9400fa0 ldr         x0,[fp,#0x18]
00000000`00403320 940068de bl          App4!_cfree (00000000`0041d698)
00000000`00403324 f94013a0 ldr         x0,[fp,#0x20]
00000000`00403328 940068dc bl          App4!_cfree (00000000`0041d698)
00000000`0040332c f94017a0 ldr         x0,[fp,#0x28]
00000000`00403330 940068da bl          App4!_cfree (00000000`0041d698)
00000000`00403334 f9401ba0 ldr         x0,[fp,#0x30]
00000000`00403338 940068d8 bl          App4!_cfree (00000000`0041d698)
00000000`0040333c f9401fa0 ldr         x0,[fp,#0x38]
00000000`00403340 940068d6 bl          App4!_cfree (00000000`0041d698)
00000000`00403344 f94023a0 ldr         x0,[fp,#0x40]
00000000`00403348 940068d4 bl          App4!_cfree (00000000`0041d698)
00000000`0040334c f94027a0 ldr         x0,[fp,#0x48]
00000000`00403350 940068d2 bl          App4!_cfree (00000000`0041d698)
00000000`00403354 12800000 mov         w0,#-1
00000000`00403358 94008673 bl          App4!sleep (00000000`00424d24)
00000000`0040335c a8c57bfd ldp         fp,lr,[sp],#0x50
00000000`00403360 d65f03c0 ret
```

Note: We see that before the problem *malloc* call, there were three buffer writes to memory addresses pointed to by values located at the following addresses: fp+0x40, fp+0x30, and fp+0x20 (highlighted in red in disassembly). However, before buffer writes, there were *free* function calls with values located at the same addresses: fp+0x20, fp+0x30, and fp+0x40 (highlighted in blue in disassembly). Therefore, we see “write after free” behavior.

8. We have the standard function prolog (highlighted in green in disassembly). Switch to stack frame #1 to check the addresses, their values, and memory contents they point to:

```
0:000> .frame /c /r 1
01 0000fffc`03e5e7d0 00000000`00403400     App4!proc+0x11c
 x0=0000fffb00000000   x1=00000000004d0000   x2=0000fffbfc000948   x3=0000fffbfc001070
 x4=7243206f6c6c6548   x5=0000fffbfc0010d2   x6=6548202136687361   x7=73617243206f6c6c
 x8=6c6c654820213668   x9=366873617243206f  x10=206f6c6c65482021  x11=0021366873617243
x12=6548202136687361  x13=73617243206f6c6c  x14=0000000000000000  x15=0000000000000000
x16=00000000004d0008  x17=0000000000423350  x18=0000000000000078  x19=0000fffc03e5f080
x20=0000000000000000  x21=00000000004d0000  x22=000000000040341c  x23=0000000000000000
x24=0000fffc03e5f770  x25=0000000031db06f0  x26=00000000004d7890  x27=0000000000010000
x28=0000000000810000   fp=0000fffc03e5e7d0   lr=0000000000403304   sp=0000fffc03e5e7d0
 pc=0000000000403304  psr=80001000 N--- EL0
App4!proc+0x11c:
00000000`00403304 f9001ba0 str         x0,[fp,#0x30]

0:000> dp fp+0x30 L1
0000fffc`03e5e800  0000fffb`fc000e50

0:000> dp 0000fffb`fc000e50
0000fffb`fc000e50  7243206f`6c6c6548 65482021`34687361
0000fffb`fc000e60  73617243`206f6c6c 6c6c6548`20213468
0000fffb`fc000e70  34687361`7243206f 206f6c6c`65482021
0000fffb`fc000e80  20213468`73617243 7243206f`6c6c6548
0000fffb`fc000e90  65482021`34687361 73617243`206f6c6c
0000fffb`fc000ea0  00000000`00213468 00000000`00000000
0000fffb`fc000eb0  00000000`00000000 00000000`00000000
0000fffb`fc000ec0  00000000`00000000 00000000`00000000

0:000> da 0000fffb`fc000e50
0000fffb`fc000e50  "Hello Crash4! Hello Crash4! Hell"
0000fffb`fc000e70  "o Crash4! Hello Crash4! Hello Cr"
0000fffb`fc000e90  "ash4! Hello Crash4!"

0:000> dpa fp+0x30 L1
0000fffc`03e5e800  0000fffb`fc000e50 "Hello Crash4! Hello Crash4! Hello Crash4! Hello Crash4!"
```

9. We know the addresses passed to heap management functions, for example, 0000fffb`fc000xxx. Find the heap region in the section and module region list:

```
0:000> !address

Mapping file section regions...
Mapping module regions...

        BaseAddress      EndAddress+1        RegionSize     Type       State                 Protect             Usage
----------------------------------------------------------------------------------------------------------------------------
+        0`00000000        0`00400000        0`00400000                                                       <unknown>
+        0`00400000        0`00410000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ          Image      [App4;
"/home/opc/ALCDA2/App4/App4"]
+        0`00410000        0`004c0000        0`000b0000                                                       Image      [App4;
"/home/opc/ALCDA2/App4/App4"]
+        0`004c0000        0`004e0000        0`00020000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE              Image      [App4;
"/home/opc/ALCDA2/App4/App4"]
+        0`004e0000        0`31db0000        0`318d0000                                                       <unknown>
+        0`31db0000        0`31df0000        0`00040000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE              <unknown>  [................]
+        0`31df0000     fffb`fc000000     fffb`ca210000                                                       <unknown>
+     fffb`fc000000     fffb`fc030000        0`00030000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE              <unknown>  [................]
+     fffb`fc030000     fffc`02630000        0`06600000                                                       <unknown>
+     fffc`02630000     fffc`02640000        0`00010000 MEM_PRIVATE MEM_COMMIT                               <unknown>  [................]
+     fffc`02640000     fffc`02e40000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE              <unknown>  [................]
+     fffc`02e40000     fffc`02e50000        0`00010000 MEM_PRIVATE MEM_COMMIT                               <unknown>  [................]
+     fffc`02e50000     fffc`03650000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE              <unknown>  [................]
+     fffc`03650000     fffc`03660000        0`00010000 MEM_PRIVATE MEM_COMMIT                               <unknown>  [................]
+     fffc`03660000     fffc`03e60000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE              <unknown>  [................]
+     fffc`03e60000     fffc`03e70000        0`00010000 MEM_PRIVATE MEM_COMMIT                               <unknown>  [................]
+     fffc`03e70000     fffc`04670000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE              <unknown>  [................]
+     fffc`04670000     fffc`04680000        0`00010000 MEM_PRIVATE MEM_COMMIT                               <unknown>  [................]
+     fffc`04680000     fffc`04e80000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE              <unknown>  [................]
```

```
+     fffc`04e80000     fffc`04e90000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                    <unknown>  [..0.......rB....]
+     fffc`04e90000     fffc`04ea0000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ               Image      [linux_vdso_so; "linux-
vdso.so.1"]
+     fffc`04ea0000     ffff`d31d0000        3`ce330000                                                       <unknown>
+     ffff`d31d0000     ffff`d3200000        0`00030000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE               <unknown>  [..............]
```

10. Check the faulting instruction and the problem memory address:

```
0:000> .cxr
Resetting default scope

0:000> r
 x0=0000fffb00000000   x1=00000000004d0000   x2=0000fffbfc000948   x3=0000fffbfc001070
 x4=7243206f6c6c6548   x5=0000fffbfc0010d2   x6=6548202136687361   x7=73617243206f6c6c
 x8=6c6c654820213668   x9=366873617243206f  x10=206f6c6c65482021  x11=0021366873617243
x12=6548202136687361  x13=73617243206f6c6c  x14=0000000000000000  x15=0000000000000000
x16=00000000004d0008  x17=0000000000423350  x18=0000000000000078  x19=0000000000000100
x20=000000000000000f  x21=0000fffc03e5f770  x22=000000000040341c  x23=0000000000000000
x24=0000fffc03e5f770  x25=0000000031db06f0  x26=00000000004d7890  x27=0000000000010000
x28=0000000000810000   fp=0000fffc03e5e790   lr=0000000000403304   sp=0000fffc03e5e790
 pc=000000000041cbec  psr=80001000 N--- EL0
App4!malloc+0x8c:
00000000`0041cbec f9400081 ldr         x1,[x4]

0:000> dp x4
7243206f`6c6c6548  ????????`???????? ????????`????????
7243206f`6c6c6558  ????????`???????? ????????`????????
7243206f`6c6c6568  ????????`???????? ????????`????????
7243206f`6c6c6578  ????????`???????? ????????`????????
7243206f`6c6c6588  ????????`???????? ????????`????????
7243206f`6c6c6598  ????????`???????? ????????`????????
7243206f`6c6c65a8  ????????`???????? ????????`????????
7243206f`6c6c65b8  ????????`???????? ????????`????????

0:000> .formats 7243206f`6c6c6548
Evaluate expression:
  Hex:     7243206f`6c6c6548
  Decimal: 8233460206695900488
  Octal:   07110310067554330625 10
  Binary:  01110010 01000011 00100000 01101111 01101100 01101100 01100101 01001000
  Chars:   rC olleH
  Time:    Thu Oct 18 19:57:49.590 27691 (UTC + 0:00)
  Float:   low 1.14314e+027 high 3.86488e+030
  Double:  2.55074e+242
```

Note: We see that the “rC olleH” (“Hello Cr” in little-endian interpretation) fragment correlates with the “Hello Cr” buffer overwrite that we saw previously in step #8.

11. We close logging before exiting WinDbg Preview:

```
0:000> .logclose
Closing open log file 'C:\ALCDA2\A64\App4\App4.log'
```

Exercise A5

- **Goal:** Learn how to identify stack corruption
- **Patterns:** Local Buffer Overflow (User Space); Execution Residue (User Space)
- \ALCDA-Dumps\Exercise-A5-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A5-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A5-A64-WinDbg.pdf

Exercise A5 (x64, GDB)

Goal: Learn how to identify stack corruption.

Patterns: Local Buffer Overflow (User Space); Execution Residue (User Space).

1. Load *core.App5* dump file and *App5* executable from the x64/App5 directory:

```
~/ALCDA2/x64/App5$ gdb -c core.App5 -se App5
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App5...done.
[New LWP 4604]
[New LWP 4603]
[New LWP 4605]
[New LWP 4606]
[New LWP 4607]
[New LWP 4608]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App5'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x0000000000007265 in ?? ()
[Current thread is 1 (Thread 0x7fb71bd94700 (LWP 4604))]
```

2. List threads and show stack trace of the problem thread:

```
(gdb) info threads
  Id   Target Id                         Frame
* 1    Thread 0x7fb71bd94700 (LWP 4604) 0x0000000000007265 in ?? ()
  2    Thread 0x1a56880 (LWP 4603)      0x0000000000441b30 in nanosleep ()
  3    Thread 0x7fb71b593700 (LWP 4605) 0x0000000000441b30 in nanosleep ()
  4    Thread 0x7fb71ad92700 (LWP 4606) 0x0000000000441b30 in nanosleep ()
  5    Thread 0x7fb71a591700 (LWP 4607) 0x0000000000441b30 in nanosleep ()
  6    Thread 0x7fb719d90700 (LWP 4608) 0x0000000000441b30 in nanosleep ()

(gdb) bt
#0  0x0000000000007265 in ?? ()
#1  0x0000000000000000 in ?? ()
```

3. We don’t see expected stack trace frames as in a normal thread stack trace:

```
(gdb) thread apply 3 bt

Thread 3 (Thread 0x7fb71b593700 (LWP 4605)):
#0  0x0000000000441b30 in nanosleep ()
#1  0x0000000000441aba in sleep ()
#2  0x0000000000401d1a in bar_two ()
#3  0x0000000000401d2b in foo_two ()
#4  0x0000000000401d44 in thread_two ()
#5  0x00000000004031f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000044438f in clone ()
```

4. We are still in thread #1. Dump raw stack data around the current stack pointer and find an ASCII buffer around a return address:

```
(gdb) info registers rsp
sp             0x7fb71bd93d80      0x7fb71bd93d80

(gdb) x/100a $rsp-0x100
0x7fb71bd93c80: 0x0     0x441aba <sleep+58>
0x7fb71bd93c90: 0x1     0x0
0x7fb71bd93ca0: 0x0     0x35216b7a748f2c00
0x7fb71bd93cb0: 0x0     0x0
0x7fb71bd93cc0: 0x7fb71bd93d50  0x401bc3 <procB+22>
0x7fb71bd93cd0: 0x0     0x7fb71bd93d66
0x7fb71bd93ce0: 0x422077654e20794d      0x7542207265676769
0x7fb71bd93cf0: 0x72656666      0x0
0x7fb71bd93d00: 0x0     0x0
0x7fb71bd93d10: 0x0     0x0
0x7fb71bd93d20: 0x0     0x0
0x7fb71bd93d30: 0x0     0x0
0x7fb71bd93d40: 0x0     0x0
0x7fb71bd93d50: 0x7fb71bd93d70  0x401cca <procA+40>              ; rbp, retaddr
0x7fb71bd93d60: 0x794d000000000000      0x6769422077654e20
0x7fb71bd93d70: 0x6666754220726567      0x7265                  ; rbp, retaddr
0x7fb71bd93d80: 0x0     0x0
0x7fb71bd93d90: 0x0     0x0
0x7fb71bd93da0: 0x0     0x0
0x7fb71bd93db0: 0x0     0x0
0x7fb71bd93dc0: 0x0     0x7fb71bd90000
0x7fb71bd93dd0: 0x7fb71bd94700  0x6ca0e0818989a649
0x7fb71bd93de0: 0x7ffc1803c31e  0x7ffc1803c31f
0x7fb71bd93df0: 0x7fb71bd94700  0x0
0x7fb71bd93e00: 0x93ced733f209a649      0x6ca0e001eabba649
0x7fb71bd93e10: 0x0     0x0
0x7fb71bd93e20: 0x0     0x0
0x7fb71bd93e30: 0x0     0x0
0x7fb71bd93e40: 0x0     0x0
0x7fb71bd93e50: 0x0     0x35216b7a748f2c00
0x7fb71bd93e60: 0x0     0x7fb71bd94700
0x7fb71bd93e70: 0x7fb71bd94700  0x44438f <clone+63>
0x7fb71bd93e80: 0x0     0x0
0x7fb71bd93e90: 0x0     0x0
0x7fb71bd93ea0: 0x0     0x0
0x7fb71bd93eb0: 0x0     0x0
0x7fb71bd93ec0: 0x0     0x0
0x7fb71bd93ed0: 0x0     0x0
0x7fb71bd93ee0: 0x0     0x0
0x7fb71bd93ef0: 0x0     0x0
```

```
0x7fb71bd93f00: 0x0     0x0
0x7fb71bd93f10: 0x0     0x0
0x7fb71bd93f20: 0x0     0x0
0x7fb71bd93f30: 0x0     0x0
0x7fb71bd93f40: 0x0     0x0
0x7fb71bd93f50: 0x0     0x0
0x7fb71bd93f60: 0x0     0x0
0x7fb71bd93f70: 0x0     0x0
0x7fb71bd93f80: 0x0     0x0
0x7fb71bd93f90: 0x0     0x0

(gdb) x/s 0x7fb71bd93d60+8
0x7fb71bd93d68: " New Bigger Buffer"
```

Exercise A5 (A64, GDB)

Goal: Learn how to identify stack corruption.

Patterns: Local Buffer Overflow (User Space); Execution Residue (User Space).

1. Load *core.11157* dump file and *App5* executable from the A64/App5 directory:

```
~/ALCDA2/A64/App5$ gdb -c core.11157 -se App5
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App5...
(No debugging symbols found in App5)

warning: Can't open file /home/opc/ALCDA2/App5/App5 during file-backed mapping note processing
[New LWP 11158]
[New LWP 11160]
[New LWP 11162]
[New LWP 11157]
[New LWP 11159]
[New LWP 11161]
Core was generated by `./App5'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x0000000000000000 in ?? ()
[Current thread is 1 (LWP 11158)]
```

2. Set logging to a file in case of lengthy output from some commands and set color highlighting off:

```
(gdb) set logging file App5.log

(gdb) set logging enabled on
Copying output to App5.log.
Copying debug output to App5.log.

(gdb) set style enabled off
```

3. List threads and show stack trace of the problem thread:

```
(gdb) info threads
  Id   Target Id         Frame
* 1    LWP 11158         0x0000000000000000 in ?? ()
  2    LWP 11160         0x000000000040ca54 in nanosleep ()
  3    LWP 11162         0x000000000040ca54 in nanosleep ()
  4    LWP 11157         0x000000000040ca54 in nanosleep ()
```

```
  5    LWP 11159         0x000000000040ca54 in nanosleep ()
  6    LWP 11161         0x000000000040ca54 in nanosleep ()

(gdb) bt
#0  0x0000000000000000 in ?? ()
Backtrace stopped: previous frame identical to this frame (corrupt stack?)
```

4. We don't see expected stack trace frames as in a normal thread stack trace:

```
(gdb) thread apply 3 bt

Thread 3 (LWP 11162):
#0  0x000000000040ca54 in nanosleep ()
#1  0x0000000000424d74 in sleep ()
#2  0x00000000004033bc in bar_five ()
#3  0x00000000004033d0 in foo_five ()
#4  0x00000000004033e8 in thread_five ()
#5  0x0000000000404cd4 in start_thread ()
#6  0x0000000000429c20 in thread_start ()
```

5. We are still in thread #1. Dump raw stack data around the current stack pointer and find an ASCII buffer around a return address:

```
(gdb) info registers sp
sp             0xfffcbdfee830      0xfffcbdfee830

(gdb) x/100a $sp-0x100
0xfffcbdfee730: 0x0     0x0
0xfffcbdfee740: 0x0     0x0
0xfffcbdfee750: 0x0     0x0
0xfffcbdfee760: 0x0     0x0
0xfffcbdfee770: 0xfffcbdfee800  0x403288 <procA+44>        ; x29, lr
0xfffcbdfee780: 0x0     0xfffcbdfee810
0xfffcbdfee790: 0x0     0x422077654e20794d
0xfffcbdfee7a0: 0x7542207265676769      0x72656666
0xfffcbdfee7b0: 0x0     0x0
0xfffcbdfee7c0: 0x0     0x0
0xfffcbdfee7d0: 0x0     0x0
0xfffcbdfee7e0: 0x0     0x0
0xfffcbdfee7f0: 0x0     0x0
0xfffcbdfee800: 0xfffcbdfee820  0x40329c <bar_one+12>      ; x29, lr
0xfffcbdfee810: 0x422077654e20794d      0x7542207265676769
0xfffcbdfee820: 0x72656666      0x0                        ; x29, lr
0xfffcbdfee830: 0x0     0x0
0xfffcbdfee840: 0x0     0x0
0xfffcbdfee850: 0x0     0x0
0xfffcbdfee860: 0x0     0x0
0xfffcbdfee870: 0xfffc00000000  0x4d7890 <__default_pthread_attr>
0xfffcbdfee880: 0x4d0000        0x0
0xfffcbdfee890: 0xfffcbdfef49c  0xfffcbdfef080
0xfffcbdfee8a0: 0x0     0x0
0xfffcbdfee8b0: 0xfffcbdfef080  0x4d7890 <__default_pthread_attr>
0xfffcbdfee8c0: 0x4d0000        0x4032b8 <thread_one>
0xfffcbdfee8d0: 0x0     0xfffcbdfef770
0xfffcbdfee8e0: 0x3ea606f0      0x4d7890 <__default_pthread_attr>
0xfffcbdfee8f0: 0x10000 0x810000
0xfffcbdfee900: 0xfffcbdfee860  0xa8d4758adeef427
0xfffcbdfee910: 0x0     0xa8db8a4105050e7
0xfffcbdfee920: 0x0     0x0
0xfffcbdfee930: 0x0     0x0
```

```
0xfffcbdfee940: 0x0     0x0
0xfffcbdfee950: 0x0     0x0
0xfffcbdfee960: 0x0     0x0
0xfffcbdfee970: 0x0     0x0
0xfffcbdfee980: 0x0     0x0
0xfffcbdfee990: 0x0     0x0
0xfffcbdfee9a0: 0x0     0x0
0xfffcbdfee9b0: 0x0     0x0
0xfffcbdfee9c0: 0x0     0x0
0xfffcbdfee9d0: 0x0     0x0
0xfffcbdfee9e0: 0x0     0x0
0xfffcbdfee9f0: 0x0     0x0
0xfffcbdfeea00: 0x0     0x0
0xfffcbdfeea10: 0x0     0x0
0xfffcbdfeea20: 0x0     0x0
0xfffcbdfeea30: 0x0     0x0
0xfffcbdfeea40: 0x0     0x0

(gdb) x/s 0xfffcbdfee820
0xfffcbdfee820: "ffer"

(gdb) x/s 0xfffcbdfee810
0xfffcbdfee810: "My New Bigger Buffer"
```

Exercise A5 (A64, WinDbg Preview)

Goal: Learn how to identify stack corruption.

Patterns: Local Buffer Overflow (User Space); Execution Residue (User Space).

1. Launch WinDbg Preview.

2. Load *core.11157* dump file from the A64\App5 folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App5\core.11157]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
(2b95.2b96): Signal SIGSEGV (Segmentation fault) code SEGV_MAPERR (Address not mapped to
object) at 0x000000000`00000000 ?? ???
```

3. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App5\App5.log
Opened log file 'C:\ALCDA2\A64\App5\App5.log'
```

4. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App5\
Symbol search path is: srv*;C:\ALCDA2\A64\App5\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app5\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App5\

0:000> .reload
..
```

5. List threads and show stack trace of the problem thread:

```
0:000> ~*k 1

Unable to get thread data for thread 0
.  0  Id: 2b95.2b96 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`bdfee830 00000000`00000000     0x0

Unable to get thread data for thread 1
   1  Id: 2b95.2b98 Suspend: 0 Teb: 00000000`00000000 Unfrozen
Unable to load image /home/opc/ALCDA2/App5/App5, Win32 error 0n2
*** WARNING: Unable to verify timestamp for App5
 # Child-SP          RetAddr               Call Site
00 0000fffc`bcfce5f0 00000000`00424d74     App5!_libc_nanosleep+0x24

Unable to get thread data for thread 2
   2  Id: 2b95.2b9a Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`bbfae5f0 00000000`00424d74     App5!_libc_nanosleep+0x24

Unable to get thread data for thread 3
   3  Id: 2b95.2b95 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`ca3bae40 00000000`00424d74     App5!_libc_nanosleep+0x24

Unable to get thread data for thread 4
   4  Id: 2b95.2b97 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`bd7de5f0 00000000`00424d74     App5!_libc_nanosleep+0x24

Unable to get thread data for thread 5
   5  Id: 2b95.2b99 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffc`bc7be5f0 00000000`00424d74     App5!_libc_nanosleep+0x24

0:000> k
 # Child-SP          RetAddr               Call Site
00 0000fffc`bdfee830 00000000`00000000     0x0
```

6. We don’t see expected stack trace frames as in a normal thread stack trace:

```
0:000> ~3k
 # Child-SP          RetAddr               Call Site
00 0000ffff`ca3bae40 00000000`00424d74     App5!_libc_nanosleep+0x24
01 0000ffff`ca3bae80 00000000`00403484     App5!sleep+0x110
02 0000ffff`ca3bb070 00000000`0040ecec     App5!main+0x90
03 0000ffff`ca3bb0c0 00000000`00403090     App5!_libc_start_main+0x304
04 0000ffff`ca3bb220 00000000`00000000     App5!start+0x4c
```

7. We are still in thread #0. Dump raw stack data around the current stack pointer and find an ASCII buffer around a return address:

```
0:000> r sp
sp=0000fffcbdfee830

0:000> r lr
lr=0000000000000000
```

```
0:000> dps sp-100 sp+100
0000fffc`bdfee730  00000000`00000000
0000fffc`bdfee738  00000000`00000000
0000fffc`bdfee740  00000000`00000000
0000fffc`bdfee748  00000000`00000000
0000fffc`bdfee750  00000000`00000000
0000fffc`bdfee758  00000000`00000000
0000fffc`bdfee760  00000000`00000000
0000fffc`bdfee768  00000000`00000000
0000fffc`bdfee770  0000fffc`bdfee800                      ; fp
0000fffc`bdfee778  00000000`00403288 App5!procA+0x2c      ; lr
0000fffc`bdfee780  00000000`00000000
0000fffc`bdfee788  0000fffc`bdfee810
0000fffc`bdfee790  00000000`00000000
0000fffc`bdfee798  42207765`4e20794d
0000fffc`bdfee7a0  75422072`65676769
0000fffc`bdfee7a8  00000000`72656666
0000fffc`bdfee7b0  00000000`00000000
0000fffc`bdfee7b8  00000000`00000000
0000fffc`bdfee7c0  00000000`00000000
0000fffc`bdfee7c8  00000000`00000000
0000fffc`bdfee7d0  00000000`00000000
0000fffc`bdfee7d8  00000000`00000000
0000fffc`bdfee7e0  00000000`00000000
0000fffc`bdfee7e8  00000000`00000000
0000fffc`bdfee7f0  00000000`00000000
0000fffc`bdfee7f8  00000000`00000000
0000fffc`bdfee800  0000fffc`bdfee820                      ; fp
0000fffc`bdfee808  00000000`0040329c App5!bar_one+0xc     ; lr
0000fffc`bdfee810  42207765`4e20794d
0000fffc`bdfee818  75422072`65676769
0000fffc`bdfee820  00000000`72656666                      ; fp
0000fffc`bdfee828  00000000`00000000                      ; lr
0000fffc`bdfee830  00000000`00000000
0000fffc`bdfee838  00000000`00000000
0000fffc`bdfee840  00000000`00000000
0000fffc`bdfee848  00000000`00000000
0000fffc`bdfee850  00000000`00000000
0000fffc`bdfee858  00000000`00000000
0000fffc`bdfee860  00000000`00000000
0000fffc`bdfee868  00000000`00000000
0000fffc`bdfee870  0000fffc`00000000
0000fffc`bdfee878  00000000`004d7890 App5!_default_pthread_attr
0000fffc`bdfee880  00000000`004d0000 App5!+0x18
0000fffc`bdfee888  00000000`00000000
0000fffc`bdfee890  0000fffc`bdfef49c
0000fffc`bdfee898  0000fffc`bdfef080
0000fffc`bdfee8a0  00000000`00000000
0000fffc`bdfee8a8  00000000`00000000
0000fffc`bdfee8b0  0000fffc`bdfef080
0000fffc`bdfee8b8  00000000`004d7890 App5!_default_pthread_attr
0000fffc`bdfee8c0  00000000`004d0000 App5!+0x18
0000fffc`bdfee8c8  00000000`004032b8 App5!thread_one
0000fffc`bdfee8d0  00000000`00000000
0000fffc`bdfee8d8  0000fffc`bdfef770
0000fffc`bdfee8e0  00000000`3ea606f0
0000fffc`bdfee8e8  00000000`004d7890 App5!_default_pthread_attr
0000fffc`bdfee8f0  00000000`00010000
0000fffc`bdfee8f8  00000000`00810000
0000fffc`bdfee900  0000fffc`bdfee860
```

```
0000fffc`bdfee908  0a8d4758`adeef427
0000fffc`bdfee910  00000000`00000000
0000fffc`bdfee918  0a8db8a4`105050e7
0000fffc`bdfee920  00000000`00000000
0000fffc`bdfee928  00000000`00000000
0000fffc`bdfee930  00000000`00000000

0:000> da 0000fffc`bdfee810
0000fffc`bdfee810  "My New Bigger Buffer"
```

Note: We are also able to reconstruct the past stack trace:

```
0000fffc`bdfee778  00000000`00403288 App5!procA+0x2c       ; lr
0000fffc`bdfee808  00000000`0040329c App5!bar_one+0xc      ; lr
```

8. We close logging before exiting WinDbg Preview:

```
0:000> .logclose
Closing open log file 'C:\ALCDA2\A64\App5\App5.log'
```

Exercise A6

- **Goal:** Learn how to identify stack overflow, stack boundaries, reconstruct stack trace
- **Patterns:** Stack Overflow (User Mode)
- \ALCDA-Dumps\Exercise-A6-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A6-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A6-A64-WinDbg.pdf

Exercise A6 (x64, GDB)

Goal: Learn how to identify stack overflow, stack boundaries, reconstruct stack trace.

Patterns: Stack Overflow (User Mode).

1. Load *core.App6* dump file and *App6* executable from the x64/App6 directory:

```
~/ALCDA2/x64/App6$ gdb -c core.App6 -se App6
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App6...done.
[New LWP 4704]
[New LWP 4707]
[New LWP 4705]
[New LWP 4703]
[New LWP 4706]
[New LWP 4708]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App6'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x0000000000401bb8 in procF ()
[Current thread is 1 (Thread 0x7f91e6de8700 (LWP 4704))]
```

2. Set logging to a file in case of lengthy output from some commands:

```
(gdb) set logging on App6.log
Copying output to App6.log.
```

3. List threads:

```
(gdb) info threads
  Id   Target Id                         Frame
* 1    Thread 0x7f91e6de8700 (LWP 4704) 0x0000000000401bb8 in procF ()
  2    Thread 0x7f91e55e5700 (LWP 4707) 0x0000000000441ab0 in nanosleep ()
  3    Thread 0x7f91e65e7700 (LWP 4705) 0x0000000000441ab0 in nanosleep ()
  4    Thread 0xec2880 (LWP 4703)       0x0000000000441ab0 in nanosleep ()
  5    Thread 0x7f91e5de6700 (LWP 4706) 0x0000000000441ab0 in nanosleep ()
  6    Thread 0x7f91e4de4700 (LWP 4708) 0x0000000000441ab0 in nanosleep ()
```

4. If we try to print the problem stack trace, we get an endless number of frames, so we quit:

```
(gdb) bt
#0  0x0000000000401bb8 in procF ()
#1  0x0000000000401c05 in procF ()
#2  0x0000000000401c05 in procF ()
#3  0x0000000000401c05 in procF ()
#4  0x0000000000401c05 in procF ()
#5  0x0000000000401c05 in procF ()
#6  0x0000000000401c05 in procF ()
#7  0x0000000000401c05 in procF ()
#8  0x0000000000401c05 in procF ()
#9  0x0000000000401c05 in procF ()
#10 0x0000000000401c05 in procF ()
#11 0x0000000000401c05 in procF ()
#12 0x0000000000401c05 in procF ()
#13 0x0000000000401c05 in procF ()
#14 0x0000000000401c05 in procF ()
#15 0x0000000000401c05 in procF ()
#16 0x0000000000401c05 in procF ()
#17 0x0000000000401c05 in procF ()
#18 0x0000000000401c05 in procF ()
#19 0x0000000000401c05 in procF ()
#20 0x0000000000401c05 in procF ()
#21 0x0000000000401c05 in procF ()
#22 0x0000000000401c05 in procF ()
#23 0x0000000000401c05 in procF ()
#24 0x0000000000401c05 in procF ()
#25 0x0000000000401c05 in procF ()
#26 0x0000000000401c05 in procF ()
#27 0x0000000000401c05 in procF ()
#28 0x0000000000401c05 in procF ()
#29 0x0000000000401c05 in procF ()
#30 0x0000000000401c05 in procF ()
#31 0x0000000000401c05 in procF ()
#32 0x0000000000401c05 in procF ()
#33 0x0000000000401c05 in procF ()
#34 0x0000000000401c05 in procF ()
#35 0x0000000000401c05 in procF ()
#36 0x0000000000401c05 in procF ()
#37 0x0000000000401c05 in procF ()
#38 0x0000000000401c05 in procF ()
#39 0x0000000000401c05 in procF ()
#40 0x0000000000401c05 in procF ()
#41 0x0000000000401c05 in procF ()
#42 0x0000000000401c05 in procF ()
#43 0x0000000000401c05 in procF ()
#44 0x0000000000401c05 in procF ()
#45 0x0000000000401c05 in procF ()
#46 0x0000000000401c05 in procF ()
#47 0x0000000000401c05 in procF ()
#48 0x0000000000401c05 in procF ()
#49 0x0000000000401c05 in procF ()
#50 0x0000000000401c05 in procF ()
#51 0x0000000000401c05 in procF ()
#52 0x0000000000401c05 in procF ()
#53 0x0000000000401c05 in procF ()
#54 0x0000000000401c05 in procF ()
#55 0x0000000000401c05 in procF ()
#56 0x0000000000401c05 in procF ()
```

```
--Type <RET> for more, q to quit, c to continue without paging--
#57 0x0000000000401c05 in procF ()
#58 0x0000000000401c05 in procF ()
#59 0x0000000000401c05 in procF ()
#60 0x0000000000401c05 in procF ()
#61 0x0000000000401c05 in procF ()
#62 0x0000000000401c05 in procF ()
#63 0x0000000000401c05 in procF ()
#64 0x0000000000401c05 in procF ()
#65 0x0000000000401c05 in procF ()
#66 0x0000000000401c05 in procF ()
#67 0x0000000000401c05 in procF ()
#68 0x0000000000401c05 in procF ()
#69 0x0000000000401c05 in procF ()
#70 0x0000000000401c05 in procF ()
#71 0x0000000000401c05 in procF ()
#72 0x0000000000401c05 in procF ()
#73 0x0000000000401c05 in procF ()
#74 0x0000000000401c05 in procF ()
#75 0x0000000000401c05 in procF ()
#76 0x0000000000401c05 in procF ()
#77 0x0000000000401c05 in procF ()
#78 0x0000000000401c05 in procF ()
#79 0x0000000000401c05 in procF ()
#80 0x0000000000401c05 in procF ()
#81 0x0000000000401c05 in procF ()
#82 0x0000000000401c05 in procF ()
#83 0x0000000000401c05 in procF ()
#84 0x0000000000401c05 in procF ()
#85 0x0000000000401c05 in procF ()
#86 0x0000000000401c05 in procF ()
#87 0x0000000000401c05 in procF ()
#88 0x0000000000401c05 in procF ()
#89 0x0000000000401c05 in procF ()
#90 0x0000000000401c05 in procF ()
#91 0x0000000000401c05 in procF ()
#92 0x0000000000401c05 in procF ()
#93 0x0000000000401c05 in procF ()
#94 0x0000000000401c05 in procF ()
#95 0x0000000000401c05 in procF ()
#96 0x0000000000401c05 in procF ()
#97 0x0000000000401c05 in procF ()
#98 0x0000000000401c05 in procF ()
#99 0x0000000000401c05 in procF ()
#100 0x0000000000401c05 in procF ()
#101 0x0000000000401c05 in procF ()
#102 0x0000000000401c05 in procF ()
#103 0x0000000000401c05 in procF ()
#104 0x0000000000401c05 in procF ()
#105 0x0000000000401c05 in procF ()
#106 0x0000000000401c05 in procF ()
#107 0x0000000000401c05 in procF ()
#108 0x0000000000401c05 in procF ()
#109 0x0000000000401c05 in procF ()
#110 0x0000000000401c05 in procF ()
#111 0x0000000000401c05 in procF ()
#112 0x0000000000401c05 in procF ()
#113 0x0000000000401c05 in procF ()
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
```

Note: It looks like a stack overflow.

5. Check if this is a stack overflow indeed. The stack region can be identified from *App6.pmap.4703* from the thread number. Since the problem thread has LWP 4704, it should be located just below the main stack region:

```
4703:   ./App6
0000000000400000      4K r---- App6
0000000000401000    592K r-x-- App6
0000000000495000    156K r---- App6
00000000004bd000     24K rw--- App6
00000000004c3000     24K rw---   [ anon ]
0000000000ec2000    140K rw---   [ anon ]
00007f91e45e4000      4K -----   [ anon ]
00007f91e45e5000   8192K rw---   [ anon ]
00007f91e4de5000      4K -----   [ anon ]
00007f91e4de6000   8192K rw---   [ anon ]
00007f91e55e6000      4K -----   [ anon ]
00007f91e55e7000   8192K rw---   [ anon ]
00007f91e5de7000      4K -----   [ anon ]
00007f91e5de8000   8192K rw---   [ anon ]
00007f91e65e8000      4K -----   [ anon ]
00007f91e65e9000   8192K rw---   [ anon ]
00007ffcec95d000    132K rw---   [ stack ]
00007ffcec9a9000     16K r----   [ anon ]
00007ffcec9ad000      4K r-x--   [ anon ]
 total            42072K
```

6. Check that manually based on the stack pointer value and section boundary addresses:

```
(gdb) x $rsp
0x7f91e65e8ef0: Cannot access memory at address 0x7f91e65e8ef0

(gdb) frame 1
#1  0x0000000000401c05 in procF ()

(gdb) x $rsp
0x7f91e65e9110: 0x00000000

(gdb) frame 2
#2  0x0000000000401c05 in procF ()

(gdb) x $rsp
0x7f91e65e9330: 0x00000000

(gdb) maintenance info sections

Exec file:
    `/home/coredump/ALCDA2/x64/App6/App6', file type elf64-x86-64.
 [0]     0x00400200->0x00400220 at 0x00000200: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]     0x00400220->0x00400244 at 0x00000220: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]     0x00400248->0x004004d0 at 0x00000248: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]     0x00401000->0x00401017 at 0x00001000: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
 [4]     0x00401018->0x004010f0 at 0x00001018: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
 [5]     0x004010f0->0x00493470 at 0x000010f0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]     0x00493470->0x00494017 at 0x00093470: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]     0x00494018->0x00494021 at 0x00094018: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [8]     0x00495000->0x004af73c at 0x00095000: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
 [9]     0x004af740->0x004bbb90 at 0x000af740: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]     0x004bbb90->0x004bbc3c at 0x000bbb90: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]     0x004bd0b0->0x004bd0d8 at 0x000bc0b0: .tdata ALLOC LOAD DATA HAS_CONTENTS
```

```
 [12]     0x004bd0d8->0x004bd120 at 0x000bc0d8: .tbss ALLOC
 [13]     0x004bd0d8->0x004bd0e0 at 0x000bc0d8: .preinit_array ALLOC LOAD DATA HAS_CONTENTS
 [14]     0x004bd0e0->0x004bd0f0 at 0x000bc0e0: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [15]     0x004bd0f0->0x004bd100 at 0x000bc0f0: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [16]     0x004bd100->0x004bfef4 at 0x000bc100: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [17]     0x004bfef8->0x004c0000 at 0x000beef8: .got ALLOC LOAD DATA HAS_CONTENTS
 [18]     0x004c0000->0x004c00f0 at 0x000bf000: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [19]     0x004c0100->0x004c1c30 at 0x000bf100: .data ALLOC LOAD DATA HAS_CONTENTS
 [20]     0x004c1c30->0x004c1c90 at 0x000c0c30: __libc_subfreeres ALLOC LOAD DATA HAS_CONTENTS
 [21]     0x004c1ca0->0x004c2408 at 0x000c0ca0: __libc_IO_vtables ALLOC LOAD DATA HAS_CONTENTS
 [22]     0x004c2408->0x004c2410 at 0x000c1408: __libc_atexit ALLOC LOAD DATA HAS_CONTENTS
 [23]     0x004c2420->0x004c8528 at 0x000c1410: .bss ALLOC
 [24]     0x004c8528->0x004c8558 at 0x000c1410: __libc_freeres_ptrs ALLOC
 [25]     0x00000000->0x00000038 at 0x000c1410: .comment READONLY HAS_CONTENTS
 [26]     0x00000000->0x00000420 at 0x000c1450: .debug_aranges READONLY HAS_CONTENTS
 [27]     0x00000000->0x000372ad at 0x000c1870: .debug_info READONLY HAS_CONTENTS
 [28]     0x00000000->0x000057e8 at 0x000f8b1d: .debug_abbrev READONLY HAS_CONTENTS
 [29]     0x00000000->0x0000aa2b at 0x000fe305: .debug_line READONLY HAS_CONTENTS
 [30]     0x00000000->0x00004d08 at 0x00108d30: .debug_str READONLY HAS_CONTENTS
 [31]     0x00000000->0x0000d4b8 at 0x0010da38: .debug_loc READONLY HAS_CONTENTS
 [32]     0x00000000->0x000024c0 at 0x0011aef0: .debug_ranges READONLY HAS_CONTENTS
Core file:
    `/home/coredump/ALCDA2/x64/App6/core.App6', file type elf64-x86-64.
 [0]     0x00000000->0x00002c60 at 0x000004a0: note0 READONLY HAS_CONTENTS
 [1]     0x00000000->0x000000d8 at 0x00000524: .reg/4704 HAS_CONTENTS
 [2]     0x00000000->0x000000d8 at 0x00000524: .reg HAS_CONTENTS
 [3]     0x00000000->0x00000080 at 0x000006b4: .note.linuxcore.siginfo/4704 HAS_CONTENTS
 [4]     0x00000000->0x00000080 at 0x000006b4: .note.linuxcore.siginfo HAS_CONTENTS
 [5]     0x00000000->0x00000140 at 0x00000748: .auxv HAS_CONTENTS
 [6]     0x00000000->0x00000100 at 0x0000089c: .note.linuxcore.file/4704 HAS_CONTENTS
 [7]     0x00000000->0x00000100 at 0x0000089c: .note.linuxcore.file HAS_CONTENTS
 [8]     0x00000000->0x00000200 at 0x000009b0: .reg2/4704 HAS_CONTENTS
 [9]     0x00000000->0x00000200 at 0x000009b0: .reg2 HAS_CONTENTS
 [10]     0x00000000->0x00000340 at 0x00000bc4: .reg-xstate/4704 HAS_CONTENTS
 [11]     0x00000000->0x00000340 at 0x00000bc4: .reg-xstate HAS_CONTENTS
 [12]     0x00000000->0x000000d8 at 0x00000f88: .reg/4707 HAS_CONTENTS
 [13]     0x00000000->0x00000200 at 0x0000107c: .reg2/4707 HAS_CONTENTS
 [14]     0x00000000->0x00000340 at 0x00001290: .reg-xstate/4707 HAS_CONTENTS
 [15]     0x00000000->0x000000d8 at 0x00001654: .reg/4705 HAS_CONTENTS
 [16]     0x00000000->0x00000200 at 0x00001748: .reg2/4705 HAS_CONTENTS
 [17]     0x00000000->0x00000340 at 0x0000195c: .reg-xstate/4705 HAS_CONTENTS
 [18]     0x00000000->0x000000d8 at 0x00001d20: .reg/4703 HAS_CONTENTS
 [19]     0x00000000->0x00000200 at 0x00001e14: .reg2/4703 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
 [20]     0x00000000->0x00000340 at 0x00002028: .reg-xstate/4703 HAS_CONTENTS
 [21]     0x00000000->0x000000d8 at 0x000023ec: .reg/4706 HAS_CONTENTS
 [22]     0x00000000->0x00000200 at 0x000024e0: .reg2/4706 HAS_CONTENTS
 [23]     0x00000000->0x00000340 at 0x000026f4: .reg-xstate/4706 HAS_CONTENTS
 [24]     0x00000000->0x000000d8 at 0x00002ab8: .reg/4708 HAS_CONTENTS
 [25]     0x00000000->0x00000200 at 0x00002bac: .reg2/4708 HAS_CONTENTS
 [26]     0x00000000->0x00000340 at 0x00002dc0: .reg-xstate/4708 HAS_CONTENTS
 [27]     0x00400000->0x00401000 at 0x00004000: load1 ALLOC LOAD READONLY HAS_CONTENTS
 [28]     0x00401000->0x00401000 at 0x00005000: load2 ALLOC READONLY CODE
 [29]     0x00495000->0x00495000 at 0x00005000: load3 ALLOC READONLY
 [30]     0x004bd000->0x004c3000 at 0x00005000: load4 ALLOC LOAD HAS_CONTENTS
 [31]     0x004c3000->0x004c9000 at 0x0000b000: load5 ALLOC LOAD HAS_CONTENTS
 [32]     0x00ec2000->0x00ee5000 at 0x00011000: load6 ALLOC LOAD HAS_CONTENTS
 [33]     0x7f91e45e4000->0x7f91e45e4000 at 0x00034000: load7 ALLOC READONLY
 [34]     0x7f91e45e5000->0x7f91e4de5000 at 0x00034000: load8 ALLOC LOAD HAS_CONTENTS
 [35]     0x7f91e4de5000->0x7f91e4de5000 at 0x00834000: load9 ALLOC READONLY
 [36]     0x7f91e4de6000->0x7f91e55e6000 at 0x00834000: load10 ALLOC LOAD HAS_CONTENTS
 [37]     0x7f91e55e6000->0x7f91e55e6000 at 0x01034000: load11 ALLOC READONLY
 [38]     0x7f91e55e7000->0x7f91e5de7000 at 0x01034000: load12 ALLOC LOAD HAS_CONTENTS
 [39]     0x7f91e5de7000->0x7f91e5de7000 at 0x01834000: load13 ALLOC READONLY
 [40]     0x7f91e5de8000->0x7f91e65e8000 at 0x01834000: load14 ALLOC LOAD HAS_CONTENTS
 [41]     0x7f91e65e8000->0x7f91e65e8000 at 0x02034000: load15 ALLOC READONLY
 [42]     0x7f91e65e9000->0x7f91e6de9000 at 0x02034000: load16 ALLOC LOAD HAS_CONTENTS
```

```
[43]     0x7ffcec95d000->0x7ffcec97e000 at 0x02834000: load17 ALLOC LOAD HAS_CONTENTS
[44]     0x7ffcec9a9000->0x7ffcec9ad000 at 0x02855000: load18 ALLOC LOAD READONLY HAS_CONTENTS
[45]     0x7ffcec9ad000->0x7ffcec9ae000 at 0x02859000: load19 ALLOC LOAD READONLY CODE HAS_CONTENTS
```

7. Dump the bottom of the raw stack to see execution residue, such as thread startup:

```
(gdb) x/1024a 0x7f91e6de9000-0x2000
0x7f91e6de7000: 0x0     0x0
0x7f91e6de7010: 0x0     0x0
0x7f91e6de7020: 0x0     0x0
0x7f91e6de7030: 0x0     0x0
0x7f91e6de7040: 0x0     0x0
0x7f91e6de7050: 0x0     0x0
0x7f91e6de7060: 0x0     0x0
0x7f91e6de7070: 0x0     0x0
0x7f91e6de7080: 0x0     0x0
0x7f91e6de7090: 0x0     0x0
0x7f91e6de70a0: 0x7f91e6de72c0  0x401c05 <procF+88>
0x7f91e6de70b0: 0x0     0x600000000
0x7f91e6de70c0: 0xffffffff      0x7
0x7f91e6de70d0: 0xffffffff      0x0
0x7f91e6de70e0: 0x0     0x0
0x7f91e6de70f0: 0x0     0x0
0x7f91e6de7100: 0x0     0x0
0x7f91e6de7110: 0x0     0x0
0x7f91e6de7120: 0x0     0x0
0x7f91e6de7130: 0x0     0x0
0x7f91e6de7140: 0x0     0x0
0x7f91e6de7150: 0x0     0x0
0x7f91e6de7160: 0x0     0x0
0x7f91e6de7170: 0x0     0x0
0x7f91e6de7180: 0x0     0x0
0x7f91e6de7190: 0x0     0x0
0x7f91e6de71a0: 0x0     0x0
0x7f91e6de71b0: 0x0     0x0
0x7f91e6de71c0: 0x0     0x0
0x7f91e6de71d0: 0x0     0x0
0x7f91e6de71e0: 0x0     0x0
0x7f91e6de71f0: 0x0     0x0
0x7f91e6de7200: 0x0     0x0
0x7f91e6de7210: 0x0     0x0
0x7f91e6de7220: 0x0     0x0
0x7f91e6de7230: 0x0     0x0
0x7f91e6de7240: 0x0     0x0
0x7f91e6de7250: 0x0     0x0
0x7f91e6de7260: 0x0     0x0
0x7f91e6de7270: 0x0     0x0
0x7f91e6de7280: 0x0     0x0
0x7f91e6de7290: 0x0     0x0
0x7f91e6de72a0: 0x0     0x0
0x7f91e6de72b0: 0x0     0x0
0x7f91e6de72c0: 0x7f91e6de74e0  0x401c05 <procF+88>
0x7f91e6de72d0: 0x0     0x500000000
0x7f91e6de72e0: 0xffffffff      0x6
0x7f91e6de72f0: 0xffffffff      0x0
0x7f91e6de7300: 0x0     0x0
0x7f91e6de7310: 0x0     0x0
0x7f91e6de7320: 0x0     0x0
0x7f91e6de7330: 0x0     0x0
0x7f91e6de7340: 0x0     0x0
```

```
0x7f91e6de7350: 0x0     0x0
0x7f91e6de7360: 0x0     0x0
0x7f91e6de7370: 0x0     0x0
0x7f91e6de7380: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f91e6de7390: 0x0     0x0
0x7f91e6de73a0: 0x0     0x0
0x7f91e6de73b0: 0x0     0x0
0x7f91e6de73c0: 0x0     0x0
0x7f91e6de73d0: 0x0     0x0
0x7f91e6de73e0: 0x0     0x0
0x7f91e6de73f0: 0x0     0x0
0x7f91e6de7400: 0x0     0x0
0x7f91e6de7410: 0x0     0x0
0x7f91e6de7420: 0x0     0x0
0x7f91e6de7430: 0x0     0x0
0x7f91e6de7440: 0x0     0x0
0x7f91e6de7450: 0x0     0x0
0x7f91e6de7460: 0x0     0x0
0x7f91e6de7470: 0x0     0x0
0x7f91e6de7480: 0x0     0x0
0x7f91e6de7490: 0x0     0x0
0x7f91e6de74a0: 0x0     0x0
0x7f91e6de74b0: 0x0     0x0
0x7f91e6de74c0: 0x0     0x0
0x7f91e6de74d0: 0x0     0x0
0x7f91e6de74e0: 0x7f91e6de7700  0x401c05 <procF+88>
0x7f91e6de74f0: 0x0     0x400000000
0x7f91e6de7500: 0xffffffff      0x5
0x7f91e6de7510: 0xffffffff      0x0
0x7f91e6de7520: 0x0     0x0
0x7f91e6de7530: 0x0     0x0
0x7f91e6de7540: 0x0     0x0
0x7f91e6de7550: 0x0     0x0
0x7f91e6de7560: 0x0     0x0
0x7f91e6de7570: 0x0     0x0
0x7f91e6de7580: 0x0     0x0
0x7f91e6de7590: 0x0     0x0
0x7f91e6de75a0: 0x0     0x0
0x7f91e6de75b0: 0x0     0x0
0x7f91e6de75c0: 0x0     0x0
0x7f91e6de75d0: 0x0     0x0
0x7f91e6de75e0: 0x0     0x0
0x7f91e6de75f0: 0x0     0x0
0x7f91e6de7600: 0x0     0x0
0x7f91e6de7610: 0x0     0x0
0x7f91e6de7620: 0x0     0x0
0x7f91e6de7630: 0x0     0x0
0x7f91e6de7640: 0x0     0x0
0x7f91e6de7650: 0x0     0x0
0x7f91e6de7660: 0x0     0x0
0x7f91e6de7670: 0x0     0x0
0x7f91e6de7680: 0x0     0x0
0x7f91e6de7690: 0x0     0x0
0x7f91e6de76a0: 0x0     0x0
0x7f91e6de76b0: 0x0     0x0
0x7f91e6de76c0: 0x0     0x0
0x7f91e6de76d0: 0x0     0x0
0x7f91e6de76e0: 0x0     0x0
0x7f91e6de76f0: 0x0     0x0
```

```
0x7f91e6de7700: 0x7f91e6de7920  0x401c05 <procF+88>
0x7f91e6de7710: 0x0     0x300000000
--Type <RET> for more, q to quit, c to continue without paging--
0x7f91e6de7720: 0xffffffff      0x4
0x7f91e6de7730: 0xffffffff      0x0
0x7f91e6de7740: 0x0     0x0
0x7f91e6de7750: 0x0     0x0
0x7f91e6de7760: 0x0     0x0
0x7f91e6de7770: 0x0     0x0
0x7f91e6de7780: 0x0     0x0
0x7f91e6de7790: 0x0     0x0
0x7f91e6de77a0: 0x0     0x0
0x7f91e6de77b0: 0x0     0x0
0x7f91e6de77c0: 0x0     0x0
0x7f91e6de77d0: 0x0     0x0
0x7f91e6de77e0: 0x0     0x0
0x7f91e6de77f0: 0x0     0x0
0x7f91e6de7800: 0x0     0x0
0x7f91e6de7810: 0x0     0x0
0x7f91e6de7820: 0x0     0x0
0x7f91e6de7830: 0x0     0x0
0x7f91e6de7840: 0x0     0x0
0x7f91e6de7850: 0x0     0x0
0x7f91e6de7860: 0x0     0x0
0x7f91e6de7870: 0x0     0x0
0x7f91e6de7880: 0x0     0x0
0x7f91e6de7890: 0x0     0x0
0x7f91e6de78a0: 0x0     0x0
0x7f91e6de78b0: 0x0     0x0
0x7f91e6de78c0: 0x0     0x0
0x7f91e6de78d0: 0x0     0x0
0x7f91e6de78e0: 0x0     0x0
0x7f91e6de78f0: 0x0     0x0
0x7f91e6de7900: 0x0     0x0
0x7f91e6de7910: 0x0     0x0
0x7f91e6de7920: 0x7f91e6de7b40  0x401c05 <procF+88>
0x7f91e6de7930: 0x0     0x200000000
0x7f91e6de7940: 0xffffffff      0x3
0x7f91e6de7950: 0xffffffff      0x0
0x7f91e6de7960: 0x0     0x0
0x7f91e6de7970: 0x0     0x0
0x7f91e6de7980: 0x0     0x0
0x7f91e6de7990: 0x0     0x0
0x7f91e6de79a0: 0x0     0x0
0x7f91e6de79b0: 0x0     0x0
0x7f91e6de79c0: 0x0     0x0
0x7f91e6de79d0: 0x0     0x0
0x7f91e6de79e0: 0x0     0x0
0x7f91e6de79f0: 0x0     0x0
0x7f91e6de7a00: 0x0     0x0
0x7f91e6de7a10: 0x0     0x0
0x7f91e6de7a20: 0x0     0x0
0x7f91e6de7a30: 0x0     0x0
0x7f91e6de7a40: 0x0     0x0
0x7f91e6de7a50: 0x0     0x0
0x7f91e6de7a60: 0x0     0x0
0x7f91e6de7a70: 0x0     0x0
0x7f91e6de7a80: 0x0     0x0
0x7f91e6de7a90: 0x0     0x0
0x7f91e6de7aa0: 0x0     0x0
```

```
--Type <RET> for more, q to quit, c to continue without paging--
0x7f91e6de7ab0: 0x0     0x0
0x7f91e6de7ac0: 0x0     0x0
0x7f91e6de7ad0: 0x0     0x0
0x7f91e6de7ae0: 0x0     0x0
0x7f91e6de7af0: 0x0     0x0
0x7f91e6de7b00: 0x0     0x0
0x7f91e6de7b10: 0x0     0x0
0x7f91e6de7b20: 0x0     0x0
0x7f91e6de7b30: 0x0     0x0
0x7f91e6de7b40: 0x7f91e6de7d60  0x401c05 <procF+88>
0x7f91e6de7b50: 0x0     0x100000000
0x7f91e6de7b60: 0xffffffff      0x2
0x7f91e6de7b70: 0xffffffff      0x0
0x7f91e6de7b80: 0x0     0x0
0x7f91e6de7b90: 0x0     0x0
0x7f91e6de7ba0: 0x0     0x0
0x7f91e6de7bb0: 0x0     0x0
0x7f91e6de7bc0: 0x0     0x0
0x7f91e6de7bd0: 0x0     0x0
0x7f91e6de7be0: 0x0     0x0
0x7f91e6de7bf0: 0x0     0x0
0x7f91e6de7c00: 0x0     0x0
0x7f91e6de7c10: 0x0     0x0
0x7f91e6de7c20: 0x0     0x0
0x7f91e6de7c30: 0x0     0x0
0x7f91e6de7c40: 0x0     0x0
0x7f91e6de7c50: 0x0     0x0
0x7f91e6de7c60: 0x0     0x0
0x7f91e6de7c70: 0x0     0x0
0x7f91e6de7c80: 0x0     0x0
0x7f91e6de7c90: 0x0     0x0
0x7f91e6de7ca0: 0x0     0x0
0x7f91e6de7cb0: 0x0     0x0
0x7f91e6de7cc0: 0x0     0x0
0x7f91e6de7cd0: 0x0     0x0
0x7f91e6de7ce0: 0x0     0x0
0x7f91e6de7cf0: 0x0     0x0
0x7f91e6de7d00: 0x0     0x0
0x7f91e6de7d10: 0x0     0x0
0x7f91e6de7d20: 0x0     0x0
0x7f91e6de7d30: 0x0     0x0
0x7f91e6de7d40: 0x0     0x0
0x7f91e6de7d50: 0x0     0x0
0x7f91e6de7d60: 0x7f91e6de7d70  0x401c16 <procE+14>
0x7f91e6de7d70: 0x7f91e6de7d80  0x401c31 <bar_one+24>
0x7f91e6de7d80: 0x7f91e6de7d90  0x401c42 <foo_one+14>
0x7f91e6de7d90: 0x7f91e6de7db0  0x401c5b <thread_one+22>
0x7f91e6de7da0: 0x0     0x0
0x7f91e6de7db0: 0x0     0x403173 <start_thread+243>
0x7f91e6de7dc0: 0x0     0x7f91e6de8700
0x7f91e6de7dd0: 0x7f91e6de8700  0x83fb3fb8616de639
0x7f91e6de7de0: 0x7ffcec97d22e  0x7ffcec97d22f
0x7f91e6de7df0: 0x7f91e6de8700  0x0
0x7f91e6de7e00: 0x7cd8f2049aede639      0x83fb3f38035fe639
0x7f91e6de7e10: 0x0     0x0
0x7f91e6de7e20: 0x0     0x0
0x7f91e6de7e30: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f91e6de7e40: 0x0     0x0
```

```
0x7f91e6de7e50: 0x0     0x40061a1f48adcb00
0x7f91e6de7e60: 0x0     0x7f91e6de8700
0x7f91e6de7e70: 0x7f91e6de8700  0x44430f <clone+63>
0x7f91e6de7e80: 0x0     0x0
0x7f91e6de7e90: 0x0     0x0
0x7f91e6de7ea0: 0x0     0x0
0x7f91e6de7eb0: 0x0     0x0
0x7f91e6de7ec0: 0x0     0x0
0x7f91e6de7ed0: 0x0     0x0
0x7f91e6de7ee0: 0x0     0x0
0x7f91e6de7ef0: 0x0     0x0
0x7f91e6de7f00: 0x0     0x0
0x7f91e6de7f10: 0x0     0x0
0x7f91e6de7f20: 0x0     0x0
0x7f91e6de7f30: 0x0     0x0
0x7f91e6de7f40: 0x0     0x0
0x7f91e6de7f50: 0x0     0x0
0x7f91e6de7f60: 0x0     0x0
0x7f91e6de7f70: 0x0     0x0
0x7f91e6de7f80: 0x0     0x0
0x7f91e6de7f90: 0x0     0x0
0x7f91e6de7fa0: 0x0     0x0
0x7f91e6de7fb0: 0x0     0x0
0x7f91e6de7fc0: 0x0     0x0
0x7f91e6de7fd0: 0x0     0x0
0x7f91e6de7fe0: 0x0     0x0
0x7f91e6de7ff0: 0x0     0x0
0x7f91e6de8000: 0x0     0x0
0x7f91e6de8010: 0x0     0x0
0x7f91e6de8020: 0x0     0x0
0x7f91e6de8030: 0x0     0x0
0x7f91e6de8040: 0x0     0x0
0x7f91e6de8050: 0x0     0x0
0x7f91e6de8060: 0x0     0x0
0x7f91e6de8070: 0x0     0x0
0x7f91e6de8080: 0x0     0x0
0x7f91e6de8090: 0x0     0x0
0x7f91e6de80a0: 0x0     0x0
0x7f91e6de80b0: 0x0     0x0
0x7f91e6de80c0: 0x0     0x0
0x7f91e6de80d0: 0x0     0x0
0x7f91e6de80e0: 0x0     0x0
0x7f91e6de80f0: 0x0     0x0
0x7f91e6de8100: 0x0     0x0
0x7f91e6de8110: 0x0     0x0
0x7f91e6de8120: 0x0     0x0
0x7f91e6de8130: 0x0     0x0
0x7f91e6de8140: 0x0     0x0
0x7f91e6de8150: 0x0     0x0
0x7f91e6de8160: 0x0     0x0
0x7f91e6de8170: 0x0     0x0
0x7f91e6de8180: 0x0     0x0
0x7f91e6de8190: 0x0     0x0
0x7f91e6de81a0: 0x0     0x0
0x7f91e6de81b0: 0x0     0x0
0x7f91e6de81c0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f91e6de81d0: 0x0     0x0
0x7f91e6de81e0: 0x0     0x0
0x7f91e6de81f0: 0x0     0x0
```

```
0x7f91e6de8200: 0x0     0x0
0x7f91e6de8210: 0x0     0x0
0x7f91e6de8220: 0x0     0x0
0x7f91e6de8230: 0x0     0x0
0x7f91e6de8240: 0x0     0x0
0x7f91e6de8250: 0x0     0x0
0x7f91e6de8260: 0x0     0x0
0x7f91e6de8270: 0x0     0x0
0x7f91e6de8280: 0x0     0x0
0x7f91e6de8290: 0x0     0x0
0x7f91e6de82a0: 0x0     0x0
0x7f91e6de82b0: 0x0     0x0
0x7f91e6de82c0: 0x0     0x0
0x7f91e6de82d0: 0x0     0x0
0x7f91e6de82e0: 0x0     0x0
0x7f91e6de82f0: 0x0     0x0
0x7f91e6de8300: 0x0     0x0
0x7f91e6de8310: 0x0     0x0
0x7f91e6de8320: 0x0     0x0
0x7f91e6de8330: 0x0     0x0
0x7f91e6de8340: 0x0     0x0
0x7f91e6de8350: 0x0     0x0
0x7f91e6de8360: 0x0     0x0
0x7f91e6de8370: 0x0     0x0
0x7f91e6de8380: 0x0     0x0
0x7f91e6de8390: 0x0     0x0
0x7f91e6de83a0: 0x0     0x0
0x7f91e6de83b0: 0x0     0x0
0x7f91e6de83c0: 0x0     0x0
0x7f91e6de83d0: 0x0     0x0
0x7f91e6de83e0: 0x0     0x0
0x7f91e6de83f0: 0x0     0x0
0x7f91e6de8400: 0x0     0x0
0x7f91e6de8410: 0x0     0x0
0x7f91e6de8420: 0x0     0x0
0x7f91e6de8430: 0x0     0x0
0x7f91e6de8440: 0x0     0x0
0x7f91e6de8450: 0x0     0x0
0x7f91e6de8460: 0x0     0x0
0x7f91e6de8470: 0x0     0x0
0x7f91e6de8480: 0x0     0x0
0x7f91e6de8490: 0x0     0x0
0x7f91e6de84a0: 0x0     0x0
0x7f91e6de84b0: 0x0     0x0
0x7f91e6de84c0: 0x0     0x0
0x7f91e6de84d0: 0x0     0x0
0x7f91e6de84e0: 0x0     0x0
0x7f91e6de84f0: 0x0     0x0
0x7f91e6de8500: 0x0     0x0
0x7f91e6de8510: 0x0     0x0
0x7f91e6de8520: 0x0     0x0
0x7f91e6de8530: 0x0     0x0
0x7f91e6de8540: 0x0     0x0
0x7f91e6de8550: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f91e6de8560: 0x0     0x0
0x7f91e6de8570: 0x0     0x0
0x7f91e6de8580: 0x0     0x0
0x7f91e6de8590: 0x0     0x0
0x7f91e6de85a0: 0x0     0x0
```

```
0x7f91e6de85b0: 0x0     0x0
0x7f91e6de85c0: 0x0     0x0
0x7f91e6de85d0: 0x0     0x0
0x7f91e6de85e0: 0x0     0x0
0x7f91e6de85f0: 0x0     0x0
0x7f91e6de8600: 0x0     0x0
0x7f91e6de8610: 0x0     0x0
0x7f91e6de8620: 0x0     0x0
0x7f91e6de8630: 0x0     0x0
0x7f91e6de8640: 0x0     0x0
0x7f91e6de8650: 0x0     0x0
0x7f91e6de8660: 0x0     0x0
0x7f91e6de8670: 0x0     0x0
0x7f91e6de8680: 0x0     0x0
0x7f91e6de8690: 0x7f91e6de8db8  0x4c1aa0 <_nl_global_locale>
0x7f91e6de86a0: 0x4c1aa0 <_nl_global_locale>    0x4c1ac0 <_nl_global_locale+32>
0x7f91e6de86b0: 0x4c1aa8 <_nl_global_locale+8>  0x0
0x7f91e6de86c0: 0x49bd00 <_nl_C_LC_CTYPE_tolower+512>   0x49c300 <_nl_C_LC_CTYPE_toupper+512>
0x7f91e6de86d0: 0x49cc00 <_nl_C_LC_CTYPE_class+256>     0x0
0x7f91e6de86e0: 0x0     0x0
0x7f91e6de86f0: 0x0     0x0
0x7f91e6de8700: 0x7f91e6de8700  0xec3b50
0x7f91e6de8710: 0x7f91e6de8700  0x1
0x7f91e6de8720: 0x0     0x40061a1f48adcb00
0x7f91e6de8730: 0xf31cc1fd9fdc30b6      0x0
0x7f91e6de8740: 0x0     0x0
0x7f91e6de8750: 0x0     0x0
0x7f91e6de8760: 0x0     0x0
0x7f91e6de8770: 0x0     0x0
0x7f91e6de8780: 0x0     0x0
0x7f91e6de8790: 0x0     0x0
0x7f91e6de87a0: 0x0     0x0
0x7f91e6de87b0: 0x0     0x0
0x7f91e6de87c0: 0x0     0x0
0x7f91e6de87d0: 0x0     0x0
0x7f91e6de87e0: 0x0     0x0
0x7f91e6de87f0: 0x0     0x0
0x7f91e6de8800: 0x0     0x0
0x7f91e6de8810: 0x0     0x0
0x7f91e6de8820: 0x0     0x0
0x7f91e6de8830: 0x0     0x0
0x7f91e6de8840: 0x0     0x0
0x7f91e6de8850: 0x0     0x0
0x7f91e6de8860: 0x0     0x0
0x7f91e6de8870: 0x0     0x0
0x7f91e6de8880: 0x0     0x0
0x7f91e6de8890: 0x0     0x0
0x7f91e6de88a0: 0x0     0x0
0x7f91e6de88b0: 0x0     0x0
0x7f91e6de88c0: 0x0     0x0
0x7f91e6de88d0: 0x0     0x0
0x7f91e6de88e0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f91e6de88f0: 0x0     0x0
0x7f91e6de8900: 0x0     0x0
0x7f91e6de8910: 0x0     0x0
0x7f91e6de8920: 0x0     0x0
0x7f91e6de8930: 0x0     0x0
0x7f91e6de8940: 0x0     0x0
0x7f91e6de8950: 0x0     0x0
```

```
0x7f91e6de8960: 0x0     0x0
0x7f91e6de8970: 0x0     0x0
0x7f91e6de8980: 0x0     0x0
0x7f91e6de8990: 0x0     0x0
0x7f91e6de89a0: 0x0     0x0
0x7f91e6de89b0: 0x0     0x0
0x7f91e6de89c0: 0x4c0120 <stack_used>   0x7f91e65e79c0
0x7f91e6de89d0: 0x1260  0x7f91e6de89e0
0x7f91e6de89e0: 0x7f91e6de89e0  0xfffffffffffffe0
0x7f91e6de89f0: 0x0     0x0
0x7f91e6de8a00: 0x7f91e6de7dd0  0x0
0x7f91e6de8a10: 0x0     0x0
0x7f91e6de8a20: 0x0     0x0
0x7f91e6de8a30: 0x0     0x0
0x7f91e6de8a40: 0x0     0x0
0x7f91e6de8a50: 0x0     0x0
0x7f91e6de8a60: 0x0     0x0
0x7f91e6de8a70: 0x0     0x0
0x7f91e6de8a80: 0x0     0x0
0x7f91e6de8a90: 0x0     0x0
0x7f91e6de8aa0: 0x0     0x0
0x7f91e6de8ab0: 0x0     0x0
0x7f91e6de8ac0: 0x0     0x0
0x7f91e6de8ad0: 0x0     0x0
0x7f91e6de8ae0: 0x0     0x0
0x7f91e6de8af0: 0x0     0x0
0x7f91e6de8b00: 0x0     0x0
0x7f91e6de8b10: 0x0     0x0
0x7f91e6de8b20: 0x0     0x0
0x7f91e6de8b30: 0x0     0x0
0x7f91e6de8b40: 0x0     0x0
0x7f91e6de8b50: 0x0     0x0
0x7f91e6de8b60: 0x0     0x0
0x7f91e6de8b70: 0x0     0x0
0x7f91e6de8b80: 0x0     0x0
0x7f91e6de8b90: 0x0     0x0
0x7f91e6de8ba0: 0x0     0x0
0x7f91e6de8bb0: 0x0     0x0
0x7f91e6de8bc0: 0x0     0x0
0x7f91e6de8bd0: 0x0     0x0
0x7f91e6de8be0: 0x0     0x0
0x7f91e6de8bf0: 0x0     0x0
0x7f91e6de8c00: 0x0     0x0
0x7f91e6de8c10: 0x7f91e6de8a10  0x0
0x7f91e6de8c20: 0x0     0x0
0x7f91e6de8c30: 0x0     0x0
0x7f91e6de8c40: 0x0     0x0
0x7f91e6de8c50: 0x0     0x0
0x7f91e6de8c60: 0x0     0x0
0x7f91e6de8c70: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f91e6de8c80: 0x0     0x0
0x7f91e6de8c90: 0x0     0x0
0x7f91e6de8ca0: 0x0     0x0
0x7f91e6de8cb0: 0x0     0x0
0x7f91e6de8cc0: 0x0     0x0
0x7f91e6de8cd0: 0x0     0x0
0x7f91e6de8ce0: 0x0     0x0
0x7f91e6de8cf0: 0x0     0x0
0x7f91e6de8d00: 0x0     0x0
```

```
0x7f91e6de8d10: 0x0     0x0
0x7f91e6de8d20: 0x16eaf938be3c7 0x0
0x7f91e6de8d30: 0x0     0x0
0x7f91e6de8d40: 0x401c45 <thread_one>   0x0
0x7f91e6de8d50: 0x0     0x0
0x7f91e6de8d60: 0x0     0x0
0x7f91e6de8d70: 0x0     0x0
0x7f91e6de8d80: 0x0     0x0
0x7f91e6de8d90: 0x7f91e65e8000  0x801000
0x7f91e6de8da0: 0x1000  0x1000
0x7f91e6de8db0: 0x0     0x0
0x7f91e6de8dc0: 0x0     0x0
0x7f91e6de8dd0: 0x0     0x0
0x7f91e6de8de0: 0x0     0x0
0x7f91e6de8df0: 0x0     0x0
0x7f91e6de8e00: 0x0     0x0
0x7f91e6de8e10: 0x0     0x0
0x7f91e6de8e20: 0x0     0x0
0x7f91e6de8e30: 0x0     0x0
0x7f91e6de8e40: 0x0     0x0
0x7f91e6de8e50: 0x0     0x0
0x7f91e6de8e60: 0x0     0x0
0x7f91e6de8e70: 0x0     0x0
0x7f91e6de8e80: 0x0     0x0
0x7f91e6de8e90: 0x0     0x0
0x7f91e6de8ea0: 0x0     0x0
0x7f91e6de8eb0: 0x0     0x0
0x7f91e6de8ec0: 0x0     0x0
0x7f91e6de8ed0: 0x0     0x0
0x7f91e6de8ee0: 0x0     0x0
0x7f91e6de8ef0: 0x0     0x0
0x7f91e6de8f00: 0x0     0x0
0x7f91e6de8f10: 0x0     0x0
0x7f91e6de8f20: 0x0     0x0
0x7f91e6de8f30: 0x0     0x0
0x7f91e6de8f40: 0x0     0x0
0x7f91e6de8f50: 0x0     0x0
0x7f91e6de8f60: 0x0     0x0
0x7f91e6de8f70: 0x0     0x0
0x7f91e6de8f80: 0x0     0x0
0x7f91e6de8f90: 0x0     0x0
0x7f91e6de8fa0: 0x0     0x0
0x7f91e6de8fb0: 0x0     0x0
0x7f91e6de8fc0: 0x0     0x0
0x7f91e6de8fd0: 0x0     0x0
0x7f91e6de8fe0: 0x0     0x0
0x7f91e6de8ff0: 0x0     0x0
```

8. See that the reconstruction of the stack trace is possible because of the standard function prologue and epilogue:

```
[...]
0x7f91e6de70a0: 0x7f91e6de72c0  0x401c05 <procF+88>
0x7f91e6de72c0: 0x7f91e6de74e0  0x401c05 <procF+88>
0x7f91e6de74e0: 0x7f91e6de7700  0x401c05 <procF+88>
0x7f91e6de7700: 0x7f91e6de7920  0x401c05 <procF+88>
0x7f91e6de7920: 0x7f91e6de7b40  0x401c05 <procF+88>
0x7f91e6de7b40: 0x7f91e6de7d60  0x401c05 <procF+88>
0x7f91e6de7d60: 0x7f91e6de7d70  0x401c16 <procE+14>
0x7f91e6de7d70: 0x7f91e6de7d80  0x401c31 <bar_one+24>
```

```
0x7f91e6de7d80: 0x7f91e6de7d90  0x401c42 <foo_one+14>
0x7f91e6de7d90: 0x7f91e6de7db0  0x401c5b <thread_one+22>
0x7f91e6de7db0: 0x0     0x403173 <start_thread+243>
```

```
(gdb) disass procF
Dump of assembler code for function procF:
   0x0000000000401bad <+0>:     push   %rbp
   0x0000000000401bae <+1>:     mov    %rsp,%rbp
   0x0000000000401bb1 <+4>:     sub    $0x210,%rsp
   0x0000000000401bb8 <+11>:    mov    %edi,-0x204(%rbp)
   0x0000000000401bbe <+17>:    lea    -0x200(%rbp),%rdx
   0x0000000000401bc5 <+24>:    mov    $0x0,%eax
   0x0000000000401bca <+29>:    mov    $0x40,%ecx
   0x0000000000401bcf <+34>:    mov    %rdx,%rdi
   0x0000000000401bd2 <+37>:    rep stos %rax,%es:(%rdi)
   0x0000000000401bd5 <+40>:    movl   $0xffffffff,-0x200(%rbp)
   0x0000000000401bdf <+50>:    mov    -0x204(%rbp),%eax
   0x0000000000401be5 <+56>:    add    $0x1,%eax
   0x0000000000401be8 <+59>:    mov    %eax,-0x1f8(%rbp)
   0x0000000000401bee <+65>:    movl   $0xffffffff,-0x1f0(%rbp)
   0x0000000000401bf8 <+75>:    mov    -0x1f8(%rbp),%eax
   0x0000000000401bfe <+81>:    mov    %eax,%edi
   0x0000000000401c00 <+83>:    callq  0x401bad <procF>
=> 0x0000000000401c05 <+88>:    nop
   0x0000000000401c06 <+89>:    leaveq
   0x0000000000401c07 <+90>:    retq
End of assembler dump.
```

9. Use the back trace command variant to get to the bottom of the stack trace:

```
(gdb) bt -20
#15398 0x0000000000401c05 in procF () at pthread_create.c:688
#15399 0x0000000000401c05 in procF () at pthread_create.c:688
#15400 0x0000000000401c05 in procF () at pthread_create.c:688
#15401 0x0000000000401c05 in procF () at pthread_create.c:688
#15402 0x0000000000401c05 in procF () at pthread_create.c:688
#15403 0x0000000000401c05 in procF () at pthread_create.c:688
#15404 0x0000000000401c05 in procF () at pthread_create.c:688
#15405 0x0000000000401c05 in procF () at pthread_create.c:688
#15406 0x0000000000401c05 in procF () at pthread_create.c:688
#15407 0x0000000000401c05 in procF () at pthread_create.c:688
#15408 0x0000000000401c05 in procF () at pthread_create.c:688
#15409 0x0000000000401c05 in procF () at pthread_create.c:688
#15410 0x0000000000401c05 in procF () at pthread_create.c:688
#15411 0x0000000000401c05 in procF () at pthread_create.c:688
#15412 0x0000000000401c16 in procE () at pthread_create.c:688
#15413 0x0000000000401c31 in bar_one () at pthread_create.c:688
#15414 0x0000000000401c42 in foo_one () at pthread_create.c:688
#15415 0x0000000000401c5b in thread_one () at pthread_create.c:688
#15416 0x0000000000403173 in start_thread (arg=<optimized out>) at pthread_create.c:486
#15417 0x000000000044430f in clone ()
```

Exercise A6 (A64, GDB)

Goal: Learn how to identify stack overflow, stack boundaries, reconstruct stack trace.

Patterns: Stack Overflow (User Mode).

1. Load *core.19393* dump file and *App6* executable from the A64/App6 directory:

```
~/ALCDA2/A64/App6$ gdb -c core.19393 -se App6
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App6...
(No debugging symbols found in App6)

warning: Can't open file /home/opc/ALCDA2/App6/App6 during file-backed mapping note processing
[New LWP 19394]
[New LWP 19393]
[New LWP 19398]
[New LWP 19397]
[New LWP 19396]
[New LWP 19395]
Core was generated by `./App6'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x00000000004031ec in procF ()
[Current thread is 1 (LWP 19394)]
```

2. Set logging to a file in case of lengthy output from some commands and set color highlighting off:

```
(gdb) set logging file App6.log

(gdb) set logging enabled on
Copying output to App6.log.
Copying debug output to App6.log.

(gdb) set style enabled off
```

3. List threads:

```
(gdb) info threads
  Id   Target Id         Frame
* 1    LWP 19394         0x00000000004031ec in procF ()
  2    LWP 19393         0x000000000040ca54 in nanosleep ()
  3    LWP 19398         0x000000000040ca54 in nanosleep ()
  4    LWP 19397         0x000000000040ca54 in nanosleep ()
```

```
  5    LWP 19396         0x000000000040ca58 in nanosleep ()
  6    LWP 19395         0x000000000040ca54 in nanosleep ()
```

4. If we try to print the problem stack trace, we get an endless number of frames, so we quit:

```
(gdb) bt
#0  0x00000000004031ec in procF ()
#1  0x0000000000403244 in procF ()
#2  0x0000000000403244 in procF ()
#3  0x0000000000403244 in procF ()
#4  0x0000000000403244 in procF ()
#5  0x0000000000403244 in procF ()
#6  0x0000000000403244 in procF ()
#7  0x0000000000403244 in procF ()
#8  0x0000000000403244 in procF ()
#9  0x0000000000403244 in procF ()
#10 0x0000000000403244 in procF ()
#11 0x0000000000403244 in procF ()
#12 0x0000000000403244 in procF ()
#13 0x0000000000403244 in procF ()
#14 0x0000000000403244 in procF ()
#15 0x0000000000403244 in procF ()
#16 0x0000000000403244 in procF ()
#17 0x0000000000403244 in procF ()
#18 0x0000000000403244 in procF ()
#19 0x0000000000403244 in procF ()
#20 0x0000000000403244 in procF ()
#21 0x0000000000403244 in procF ()
#22 0x0000000000403244 in procF ()
#23 0x0000000000403244 in procF ()
#24 0x0000000000403244 in procF ()
#25 0x0000000000403244 in procF ()
#26 0x0000000000403244 in procF ()
#27 0x0000000000403244 in procF ()
#28 0x0000000000403244 in procF ()
#29 0x0000000000403244 in procF ()
#30 0x0000000000403244 in procF ()
#31 0x0000000000403244 in procF ()
#32 0x0000000000403244 in procF ()
#33 0x0000000000403244 in procF ()
#34 0x0000000000403244 in procF ()
#35 0x0000000000403244 in procF ()
#36 0x0000000000403244 in procF ()
#37 0x0000000000403244 in procF ()
#38 0x0000000000403244 in procF ()
#39 0x0000000000403244 in procF ()
#40 0x0000000000403244 in procF ()
#41 0x0000000000403244 in procF ()
#42 0x0000000000403244 in procF ()
#43 0x0000000000403244 in procF ()
#44 0x0000000000403244 in procF ()
#45 0x0000000000403244 in procF ()
#46 0x0000000000403244 in procF ()
#47 0x0000000000403244 in procF ()
#48 0x0000000000403244 in procF ()
--Type <RET> for more, q to quit, c to continue without paging--
#49 0x0000000000403244 in procF ()
#50 0x0000000000403244 in procF ()
#51 0x0000000000403244 in procF ()
#52 0x0000000000403244 in procF ()
```

```
#53 0x0000000000403244 in procF ()
#54 0x0000000000403244 in procF ()
#55 0x0000000000403244 in procF ()
#56 0x0000000000403244 in procF ()
#57 0x0000000000403244 in procF ()
#58 0x0000000000403244 in procF ()
#59 0x0000000000403244 in procF ()
#60 0x0000000000403244 in procF ()
#61 0x0000000000403244 in procF ()
#62 0x0000000000403244 in procF ()
#63 0x0000000000403244 in procF ()
#64 0x0000000000403244 in procF ()
#65 0x0000000000403244 in procF ()
#66 0x0000000000403244 in procF ()
#67 0x0000000000403244 in procF ()
#68 0x0000000000403244 in procF ()
#69 0x0000000000403244 in procF ()
#70 0x0000000000403244 in procF ()
#71 0x0000000000403244 in procF ()
#72 0x0000000000403244 in procF ()
#73 0x0000000000403244 in procF ()
#74 0x0000000000403244 in procF ()
#75 0x0000000000403244 in procF ()
#76 0x0000000000403244 in procF ()
#77 0x0000000000403244 in procF ()
#78 0x0000000000403244 in procF ()
#79 0x0000000000403244 in procF ()
#80 0x0000000000403244 in procF ()
#81 0x0000000000403244 in procF ()
#82 0x0000000000403244 in procF ()
#83 0x0000000000403244 in procF ()
#84 0x0000000000403244 in procF ()
#85 0x0000000000403244 in procF ()
#86 0x0000000000403244 in procF ()
#87 0x0000000000403244 in procF ()
#88 0x0000000000403244 in procF ()
#89 0x0000000000403244 in procF ()
#90 0x0000000000403244 in procF ()
#91 0x0000000000403244 in procF ()
#92 0x0000000000403244 in procF ()
#93 0x0000000000403244 in procF ()
#94 0x0000000000403244 in procF ()
#95 0x0000000000403244 in procF ()
#96 0x0000000000403244 in procF ()
#97 0x0000000000403244 in procF ()
--Type <RET> for more, q to quit, c to continue without paging--
#98 0x0000000000403244 in procF ()
#99 0x0000000000403244 in procF ()
#100 0x0000000000403244 in procF ()
#101 0x0000000000403244 in procF ()
#102 0x0000000000403244 in procF ()
#103 0x0000000000403244 in procF ()
#104 0x0000000000403244 in procF ()
#105 0x0000000000403244 in procF ()
#106 0x0000000000403244 in procF ()
#107 0x0000000000403244 in procF ()
#108 0x0000000000403244 in procF ()
#109 0x0000000000403244 in procF ()
#110 0x0000000000403244 in procF ()
#111 0x0000000000403244 in procF ()
```

```
#112 0x0000000000403244 in procF ()
#113 0x0000000000403244 in procF ()
#114 0x0000000000403244 in procF ()
#115 0x0000000000403244 in procF ()
#116 0x0000000000403244 in procF ()
#117 0x0000000000403244 in procF ()
#118 0x0000000000403244 in procF ()
#119 0x0000000000403244 in procF ()
#120 0x0000000000403244 in procF ()
#121 0x0000000000403244 in procF ()
#122 0x0000000000403244 in procF ()
#123 0x0000000000403244 in procF ()
#124 0x0000000000403244 in procF ()
#125 0x0000000000403244 in procF ()
#126 0x0000000000403244 in procF ()
#127 0x0000000000403244 in procF ()
#128 0x0000000000403244 in procF ()
#129 0x0000000000403244 in procF ()
#130 0x0000000000403244 in procF ()
#131 0x0000000000403244 in procF ()
#132 0x0000000000403244 in procF ()
#133 0x0000000000403244 in procF ()
#134 0x0000000000403244 in procF ()
#135 0x0000000000403244 in procF ()
#136 0x0000000000403244 in procF ()
#137 0x0000000000403244 in procF ()
#138 0x0000000000403244 in procF ()
#139 0x0000000000403244 in procF ()
#140 0x0000000000403244 in procF ()
#141 0x0000000000403244 in procF ()
#142 0x0000000000403244 in procF ()
#143 0x0000000000403244 in procF ()
#144 0x0000000000403244 in procF ()
#145 0x0000000000403244 in procF ()
#146 0x0000000000403244 in procF ()
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
```

Note: It looks like a stack overflow.

5. Check if this is a stack overflow indeed. The stack region can be identified from *App6.pmap.19393* from the thread number. Since the problem thread has LWP 19394, it should be located just below the main stack region:

```
19393:   ./App6
0000000000400000    768K r-x-- App6
00000000004c0000    128K rw--- App6
0000000030aa0000    256K rw---   [ anon ]
0000ffff685c0000     64K -----   [ anon ]
0000ffff685d0000   8192K rw---   [ anon ]
0000ffff68dd0000     64K -----   [ anon ]
0000ffff68de0000   8192K rw---   [ anon ]
0000ffff695e0000     64K -----   [ anon ]
0000ffff695f0000   8192K rw---   [ anon ]
0000ffff69df0000     64K -----   [ anon ]
0000ffff69e00000   8192K rw---   [ anon ]
0000ffff6a600000     64K -----   [ anon ]
0000ffff6a610000   8192K rw---   [ anon ]
0000ffff6ae10000     64K r----   [ anon ]
0000ffff6ae20000     64K r-x--   [ anon ]
0000ffffe3b20000    192K rw---   [ stack ]
```

```
total            42752K
```

6. Check that manually based on the stack pointer value and section boundary addresses:

```
(gdb) x $sp
0xffff6a610000: 0x00000000

(gdb) x $sp-10
0xffff6a60fff6: 0x00000000

(gdb) frame 1
#1  0x0000000000403244 in procF ()

(gdb) x $sp
0xffff6a610210: 0x6a610430

(gdb) frame 2
#2  0x0000000000403244 in procF ()

(gdb) x $sp
0xffff6a610430: 0x6a610650

(gdb) maintenance info sections
Exec file: `/home/ubuntu/ALCDA2/A64/App6/App6', file type elf64-littleaarch64.
 [0]     0x00400190->0x004001b0 at 0x00000190: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]     0x004001b0->0x004001d4 at 0x000001b0: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]     0x004001d8->0x00400250 at 0x000001d8: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]     0x00400250->0x00400264 at 0x00000250: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
 [4]     0x00400270->0x004002c0 at 0x00000270: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
 [5]     0x004002c0->0x00487158 at 0x000002c0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]     0x00487158->0x00488e28 at 0x00087158: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]     0x00488e28->0x00489278 at 0x00088e28: __libc_thread_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [8]     0x00489278->0x00489288 at 0x00089278: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [9]     0x00489290->0x004a176d at 0x00089290: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]    0x004a176d->0x004a176e at 0x000a176d: .stapsdt.base ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]    0x004a1770->0x004a1ea8 at 0x000a1770: __libc_IO_vtables ALLOC LOAD READONLY DATA HAS_CONTENTS
 [12]    0x004a1ea8->0x004a1f10 at 0x000a1ea8: __libc_subfreeres ALLOC LOAD READONLY DATA HAS_CONTENTS
 [13]    0x004a1f10->0x004a1f18 at 0x000a1f10: __libc_atexit ALLOC LOAD READONLY DATA HAS_CONTENTS
 [14]    0x004a1f18->0x004a1f28 at 0x000a1f18: __libc_thread_subfreeres ALLOC LOAD READONLY DATA HAS_CONTENTS
 [15]    0x004a1f28->0x004b0594 at 0x000a1f28: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [16]    0x004b0594->0x004b0751 at 0x000b0594: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [17]    0x004cfb20->0x004cfb48 at 0x000bfb20: .tdata ALLOC LOAD DATA HAS_CONTENTS
 [18]    0x004cfb48->0x004cfb98 at 0x000bfb48: .tbss ALLOC
 [19]    0x004cfb48->0x004cfb50 at 0x000bfb48: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [20]    0x004cfb50->0x004cfb60 at 0x000bfb50: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [21]    0x004cfb60->0x004cfb68 at 0x000bfb60: .jcr ALLOC LOAD DATA HAS_CONTENTS
 [22]    0x004cfb68->0x004cff24 at 0x000bfb68: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [23]    0x004cff28->0x004cffe8 at 0x000bff28: .got ALLOC LOAD DATA HAS_CONTENTS
 [24]    0x004cffe8->0x004d0028 at 0x000bffe8: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [25]    0x004d0030->0x004d1580 at 0x000c0030: .data ALLOC LOAD DATA HAS_CONTENTS
 [26]    0x004d1580->0x004d8050 at 0x000c1580: .bss ALLOC
 [27]    0x004d8050->0x004d8088 at 0x000c1580: __libc_freeres_ptrs ALLOC
 [28]    0x00000000->0x00000031 at 0x000c1580: .comment READONLY HAS_CONTENTS
 [29]    0x00000000->0x00001cb0 at 0x000c15b4: .note.stapsdt READONLY HAS_CONTENTS
Core file: `/home/ubuntu/ALCDA2/A64/App6/core.19393', file type elf64-littleaarch64.
 [0]     0x00000000->0x00002838 at 0x000003f8: note0 READONLY HAS_CONTENTS
 [1]     0x00000000->0x00000110 at 0x0000047c: .reg/19394 HAS_CONTENTS
 [2]     0x00000000->0x00000110 at 0x0000047c: .reg HAS_CONTENTS
 [3]     0x00000000->0x00000080 at 0x00000644: .note.linuxcore.siginfo/19394 HAS_CONTENTS
 [4]     0x00000000->0x00000080 at 0x00000644: .note.linuxcore.siginfo HAS_CONTENTS
 [5]     0x00000000->0x00000160 at 0x000006d8: .auxv HAS_CONTENTS
 [6]     0x00000000->0x00000076 at 0x0000084c: .note.linuxcore.file/19394 HAS_CONTENTS
 [7]     0x00000000->0x00000076 at 0x0000084c: .note.linuxcore.file HAS_CONTENTS
 [8]     0x00000000->0x00000210 at 0x000008d8: .reg2/19394 HAS_CONTENTS
 [9]     0x00000000->0x00000210 at 0x000008d8: .reg2 HAS_CONTENTS
 [10]    0x00000000->0x00000008 at 0x00000afc: .reg-aarch-tls/19394 HAS_CONTENTS
 [11]    0x00000000->0x00000008 at 0x00000afc: .reg-aarch-tls HAS_CONTENTS
 [12]    0x00000000->0x00000108 at 0x00000b18: .reg-aarch-hw-break/19394 HAS_CONTENTS
 [13]    0x00000000->0x00000108 at 0x00000b18: .reg-aarch-hw-break HAS_CONTENTS
```

```
 [14]     0x00000000->0x00000108 at 0x00000c34: .reg-aarch-hw-watch/19394 HAS_CONTENTS
 [15]     0x00000000->0x00000108 at 0x00000c34: .reg-aarch-hw-watch HAS_CONTENTS
 [16]     0x00000000->0x00000110 at 0x00000dd8: .reg/19393 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
 [17]     0x00000000->0x00000210 at 0x00000f04: .reg2/19393 HAS_CONTENTS
 [18]     0x00000000->0x00000008 at 0x00001128: .reg-aarch-tls/19393 HAS_CONTENTS
 [19]     0x00000000->0x00000108 at 0x00001144: .reg-aarch-hw-break/19393 HAS_CONTENTS
 [20]     0x00000000->0x00000108 at 0x00001260: .reg-aarch-hw-watch/19393 HAS_CONTENTS
 [21]     0x00000000->0x00000110 at 0x00001404: .reg/19398 HAS_CONTENTS
 [22]     0x00000000->0x00000210 at 0x00001530: .reg2/19398 HAS_CONTENTS
 [23]     0x00000000->0x00000008 at 0x00001754: .reg-aarch-tls/19398 HAS_CONTENTS
 [24]     0x00000000->0x00000108 at 0x00001770: .reg-aarch-hw-break/19398 HAS_CONTENTS
 [25]     0x00000000->0x00000108 at 0x0000188c: .reg-aarch-hw-watch/19398 HAS_CONTENTS
 [26]     0x00000000->0x00000110 at 0x00001a30: .reg/19397 HAS_CONTENTS
 [27]     0x00000000->0x00000210 at 0x00001b5c: .reg2/19397 HAS_CONTENTS
 [28]     0x00000000->0x00000008 at 0x00001d80: .reg-aarch-tls/19397 HAS_CONTENTS
 [29]     0x00000000->0x00000108 at 0x00001d9c: .reg-aarch-hw-break/19397 HAS_CONTENTS
 [30]     0x00000000->0x00000108 at 0x00001eb8: .reg-aarch-hw-watch/19397 HAS_CONTENTS
 [31]     0x00000000->0x00000110 at 0x0000205c: .reg/19396 HAS_CONTENTS
 [32]     0x00000000->0x00000210 at 0x00002188: .reg2/19396 HAS_CONTENTS
 [33]     0x00000000->0x00000008 at 0x000023ac: .reg-aarch-tls/19396 HAS_CONTENTS
 [34]     0x00000000->0x00000108 at 0x000023c8: .reg-aarch-hw-break/19396 HAS_CONTENTS
 [35]     0x00000000->0x00000108 at 0x000024e4: .reg-aarch-hw-watch/19396 HAS_CONTENTS
 [36]     0x00000000->0x00000110 at 0x00002688: .reg/19395 HAS_CONTENTS
 [37]     0x00000000->0x00000210 at 0x000027b4: .reg2/19395 HAS_CONTENTS
 [38]     0x00000000->0x00000008 at 0x000029d8: .reg-aarch-tls/19395 HAS_CONTENTS
 [39]     0x00000000->0x00000108 at 0x000029f4: .reg-aarch-hw-break/19395 HAS_CONTENTS
 [40]     0x00000000->0x00000108 at 0x00002b10: .reg-aarch-hw-watch/19395 HAS_CONTENTS
 [41]     0x00400000->0x00410000 at 0x00010000: load1a ALLOC LOAD READONLY CODE HAS_CONTENTS
 [42]     0x00410000->0x004c0000 at 0x00020000: load1b ALLOC READONLY CODE
 [43]     0x004c0000->0x004e0000 at 0x00020000: load2 ALLOC LOAD HAS_CONTENTS
 [44]     0x30aa0000->0x30ae0000 at 0x00040000: load3 ALLOC LOAD HAS_CONTENTS
 [45]     0xffff685c0000->0xffff685d0000 at 0x00080000: load4 ALLOC LOAD READONLY HAS_CONTENTS
 [46]     0xffff685d0000->0xffff68dd0000 at 0x00090000: load5 ALLOC LOAD HAS_CONTENTS
 [47]     0xffff68dd0000->0xffff68de0000 at 0x00890000: load6 ALLOC LOAD READONLY HAS_CONTENTS
 [48]     0xffff68de0000->0xffff695e0000 at 0x008a0000: load7 ALLOC LOAD HAS_CONTENTS
 [49]     0xffff695e0000->0xffff695f0000 at 0x010a0000: load8 ALLOC LOAD READONLY HAS_CONTENTS
 [50]     0xffff695f0000->0xffff69df0000 at 0x010b0000: load9 ALLOC LOAD HAS_CONTENTS
 [51]     0xffff69df0000->0xffff69e00000 at 0x018b0000: load10 ALLOC LOAD READONLY HAS_CONTENTS
 [52]     0xffff69e00000->0xffff6a600000 at 0x018c0000: load11 ALLOC LOAD HAS_CONTENTS
 [53]     0xffff6a600000->0xffff6a610000 at 0x020c0000: load12 ALLOC LOAD READONLY HAS_CONTENTS
 [54]     0xffff6a610000->0xffff6ae10000 at 0x020d0000: load13 ALLOC LOAD HAS_CONTENTS
 [55]     0xffff6ae10000->0xffff6ae20000 at 0x028d0000: load14 ALLOC LOAD READONLY HAS_CONTENTS
 [56]     0xffff6ae20000->0xffff6ae30000 at 0x028e0000: load15 ALLOC LOAD READONLY CODE HAS_CONTENTS
 [57]     0xffffe3b20000->0xffffe3b50000 at 0x028f0000: load16 ALLOC LOAD HAS_CONTENTS
```

Note: The stack pointer points to the start of the stack region. The addresses below it should be inaccessible at runtime. However, the committed pages were included in the crash dump, and we see zeroes since GDB can read them.

7. Dump the bottom of the raw stack to see execution residue, such as thread startup:

```
(gdb) x/1024a 0xffff6ae10000-0x2000
0xffff6ae0e000: 0x0     0x0
0xffff6ae0e010: 0x0     0x0
0xffff6ae0e020: 0x0     0x0
0xffff6ae0e030: 0x0     0x0
0xffff6ae0e040: 0x0     0x0
0xffff6ae0e050: 0x0     0x0
0xffff6ae0e060: 0x0     0x0
0xffff6ae0e070: 0x0     0x0
0xffff6ae0e080: 0x0     0x0
0xffff6ae0e090: 0x0     0x0
0xffff6ae0e0a0: 0x0     0x0
0xffff6ae0e0b0: 0x0     0x0
0xffff6ae0e0c0: 0x0     0x0
0xffff6ae0e0d0: 0x0     0x0
0xffff6ae0e0e0: 0x0     0x0
```

```
0xffff6ae0e0f0: 0x0     0x0
0xffff6ae0e100: 0x0     0x0
0xffff6ae0e110: 0x0     0x0
0xffff6ae0e120: 0x0     0x0
0xffff6ae0e130: 0x0     0x0
0xffff6ae0e140: 0x0     0x0
0xffff6ae0e150: 0x0     0x0
0xffff6ae0e160: 0x0     0x0
0xffff6ae0e170: 0x0     0x0
0xffff6ae0e180: 0x0     0x0
0xffff6ae0e190: 0x0     0x0
0xffff6ae0e1a0: 0x0     0x0
0xffff6ae0e1b0: 0xffff6ae0e3d0  0x403244 <procF+92>
0xffff6ae0e1c0: 0x0     0x300000000
0xffff6ae0e1d0: 0xffffffff      0x4
0xffff6ae0e1e0: 0xffffffff      0x0
0xffff6ae0e1f0: 0x0     0x0
0xffff6ae0e200: 0x0     0x0
0xffff6ae0e210: 0x0     0x0
0xffff6ae0e220: 0x0     0x0
0xffff6ae0e230: 0x0     0x0
0xffff6ae0e240: 0x0     0x0
0xffff6ae0e250: 0x0     0x0
0xffff6ae0e260: 0x0     0x0
0xffff6ae0e270: 0x0     0x0
0xffff6ae0e280: 0x0     0x0
0xffff6ae0e290: 0x0     0x0
0xffff6ae0e2a0: 0x0     0x0
0xffff6ae0e2b0: 0x0     0x0
0xffff6ae0e2c0: 0x0     0x0
0xffff6ae0e2d0: 0x0     0x0
0xffff6ae0e2e0: 0x0     0x0
0xffff6ae0e2f0: 0x0     0x0
0xffff6ae0e300: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xffff6ae0e310: 0x0     0x0
0xffff6ae0e320: 0x0     0x0
0xffff6ae0e330: 0x0     0x0
0xffff6ae0e340: 0x0     0x0
0xffff6ae0e350: 0x0     0x0
0xffff6ae0e360: 0x0     0x0
0xffff6ae0e370: 0x0     0x0
0xffff6ae0e380: 0x0     0x0
0xffff6ae0e390: 0x0     0x0
0xffff6ae0e3a0: 0x0     0x0
0xffff6ae0e3b0: 0x0     0x0
0xffff6ae0e3c0: 0x0     0x0
0xffff6ae0e3d0: 0xffff6ae0e5f0  0x403244 <procF+92>
0xffff6ae0e3e0: 0x0     0x200000000
0xffff6ae0e3f0: 0xffffffff      0x3
0xffff6ae0e400: 0xffffffff      0x0
0xffff6ae0e410: 0x0     0x0
0xffff6ae0e420: 0x0     0x0
0xffff6ae0e430: 0x0     0x0
0xffff6ae0e440: 0x0     0x0
0xffff6ae0e450: 0x0     0x0
0xffff6ae0e460: 0x0     0x0
0xffff6ae0e470: 0x0     0x0
0xffff6ae0e480: 0x0     0x0
0xffff6ae0e490: 0x0     0x0
```

```
0xffff6ae0e4a0: 0x0     0x0
0xffff6ae0e4b0: 0x0     0x0
0xffff6ae0e4c0: 0x0     0x0
0xffff6ae0e4d0: 0x0     0x0
0xffff6ae0e4e0: 0x0     0x0
0xffff6ae0e4f0: 0x0     0x0
0xffff6ae0e500: 0x0     0x0
0xffff6ae0e510: 0x0     0x0
0xffff6ae0e520: 0x0     0x0
0xffff6ae0e530: 0x0     0x0
0xffff6ae0e540: 0x0     0x0
0xffff6ae0e550: 0x0     0x0
0xffff6ae0e560: 0x0     0x0
0xffff6ae0e570: 0x0     0x0
0xffff6ae0e580: 0x0     0x0
0xffff6ae0e590: 0x0     0x0
0xffff6ae0e5a0: 0x0     0x0
0xffff6ae0e5b0: 0x0     0x0
0xffff6ae0e5c0: 0x0     0x0
0xffff6ae0e5d0: 0x0     0x0
0xffff6ae0e5e0: 0x0     0x0
0xffff6ae0e5f0: 0xffff6ae0e810  0x403260 <procE+16>
0xffff6ae0e600: 0xffff6ae0e670  0x100000000
0xffff6ae0e610: 0xffffffff      0x2
--Type <RET> for more, q to quit, c to continue without paging--
0xffff6ae0e620: 0xffffffff      0x0
0xffff6ae0e630: 0x0     0x0
0xffff6ae0e640: 0x0     0x0
0xffff6ae0e650: 0x0     0x0
0xffff6ae0e660: 0x0     0x0
0xffff6ae0e670: 0x0     0x0
0xffff6ae0e680: 0x0     0x0
0xffff6ae0e690: 0x0     0x0
0xffff6ae0e6a0: 0x0     0x0
0xffff6ae0e6b0: 0x0     0x0
0xffff6ae0e6c0: 0x0     0x0
0xffff6ae0e6d0: 0x0     0x0
0xffff6ae0e6e0: 0x0     0x0
0xffff6ae0e6f0: 0x0     0x0
0xffff6ae0e700: 0x0     0x0
0xffff6ae0e710: 0x0     0x0
0xffff6ae0e720: 0x0     0x0
0xffff6ae0e730: 0x0     0x0
0xffff6ae0e740: 0x0     0x0
0xffff6ae0e750: 0x0     0x0
0xffff6ae0e760: 0x0     0x0
0xffff6ae0e770: 0x0     0x0
0xffff6ae0e780: 0x0     0x0
0xffff6ae0e790: 0x0     0x0
0xffff6ae0e7a0: 0x0     0x0
0xffff6ae0e7b0: 0x0     0x0
0xffff6ae0e7c0: 0x0     0x0
0xffff6ae0e7d0: 0x0     0x0
0xffff6ae0e7e0: 0x0     0x0
0xffff6ae0e7f0: 0x0     0x0
0xffff6ae0e800: 0x0     0x0
0xffff6ae0e810: 0xffff6ae0e820  0x40327c <bar_one+20>
0xffff6ae0e820: 0xffff6ae0e830  0x403290 <foo_one+12>
0xffff6ae0e830: 0xffff6ae0e840  0x4032a8 <thread_one+16>
0xffff6ae0e840: 0xffff6ae0e860  0x404cd4 <start_thread+180>
```

```
0xffff6ae0e850: 0xffff6ae0f080  0x0
0xffff6ae0e860: 0x0     0x429c20 <thread_start+48>
0xffff6ae0e870: 0xffff6ae0f080  0x4d7890 <__default_pthread_attr>
0xffff6ae0e880: 0x4d0000        0x0
0xffff6ae0e890: 0xffff6ae0f49c  0xffff6ae0f080
0xffff6ae0e8a0: 0x0     0x0
0xffff6ae0e8b0: 0xffff6ae0f080  0x4d7890 <__default_pthread_attr>
0xffff6ae0e8c0: 0x4d0000        0x403298 <thread_one>
0xffff6ae0e8d0: 0x0     0xffff6ae0f770
0xffff6ae0e8e0: 0x30aa06f0      0x4d7890 <__default_pthread_attr>
0xffff6ae0e8f0: 0x10000 0x810000
0xffff6ae0e900: 0xffff6ae0e860  0x5afbedf415cdf4fb
0xffff6ae0e910: 0x0     0x5afb120b7f6d503b
0xffff6ae0e920: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xffff6ae0e930: 0x0     0x0
0xffff6ae0e940: 0x0     0x0
0xffff6ae0e950: 0x0     0x0
0xffff6ae0e960: 0x0     0x0
0xffff6ae0e970: 0x0     0x0
0xffff6ae0e980: 0x0     0x0
0xffff6ae0e990: 0x0     0x0
0xffff6ae0e9a0: 0x0     0x0
0xffff6ae0e9b0: 0x0     0x0
0xffff6ae0e9c0: 0x0     0x0
0xffff6ae0e9d0: 0x0     0x0
0xffff6ae0e9e0: 0x0     0x0
0xffff6ae0e9f0: 0x0     0x0
0xffff6ae0ea00: 0x0     0x0
0xffff6ae0ea10: 0x0     0x0
0xffff6ae0ea20: 0x0     0x0
0xffff6ae0ea30: 0x0     0x0
0xffff6ae0ea40: 0x0     0x0
0xffff6ae0ea50: 0x0     0x0
0xffff6ae0ea60: 0x0     0x0
0xffff6ae0ea70: 0x0     0x0
0xffff6ae0ea80: 0x0     0x0
0xffff6ae0ea90: 0x0     0x0
0xffff6ae0eaa0: 0x0     0x0
0xffff6ae0eab0: 0x0     0x0
0xffff6ae0eac0: 0x0     0x0
0xffff6ae0ead0: 0x0     0x0
0xffff6ae0eae0: 0x0     0x0
0xffff6ae0eaf0: 0x0     0x0
0xffff6ae0eb00: 0x0     0x0
0xffff6ae0eb10: 0x0     0x0
0xffff6ae0eb20: 0x0     0x0
0xffff6ae0eb30: 0x0     0x0
0xffff6ae0eb40: 0x0     0x0
0xffff6ae0eb50: 0x0     0x0
0xffff6ae0eb60: 0x0     0x0
0xffff6ae0eb70: 0x0     0x0
0xffff6ae0eb80: 0x0     0x0
0xffff6ae0eb90: 0x0     0x0
0xffff6ae0eba0: 0x0     0x0
0xffff6ae0ebb0: 0x0     0x0
0xffff6ae0ebc0: 0x0     0x0
0xffff6ae0ebd0: 0x0     0x0
0xffff6ae0ebe0: 0x0     0x0
0xffff6ae0ebf0: 0x0     0x0
```

```
0xffff6ae0ec00: 0x0     0x0
0xffff6ae0ec10: 0x0     0x0
0xffff6ae0ec20: 0x0     0x0
0xffff6ae0ec30: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xffff6ae0ec40: 0x0     0x0
0xffff6ae0ec50: 0x0     0x0
0xffff6ae0ec60: 0x0     0x0
0xffff6ae0ec70: 0x0     0x0
0xffff6ae0ec80: 0x0     0x0
0xffff6ae0ec90: 0x0     0x0
0xffff6ae0eca0: 0x0     0x0
0xffff6ae0ecb0: 0x0     0x0
0xffff6ae0ecc0: 0x0     0x0
0xffff6ae0ecd0: 0x0     0x0
0xffff6ae0ece0: 0x0     0x0
0xffff6ae0ecf0: 0x0     0x0
0xffff6ae0ed00: 0x0     0x0
0xffff6ae0ed10: 0x0     0x0
0xffff6ae0ed20: 0x0     0x0
0xffff6ae0ed30: 0x0     0x0
0xffff6ae0ed40: 0x0     0x0
0xffff6ae0ed50: 0x0     0x0
0xffff6ae0ed60: 0x0     0x0
0xffff6ae0ed70: 0x0     0x0
0xffff6ae0ed80: 0x0     0x0
0xffff6ae0ed90: 0x0     0x0
0xffff6ae0eda0: 0x0     0x0
0xffff6ae0edb0: 0x0     0x0
0xffff6ae0edc0: 0x0     0x0
0xffff6ae0edd0: 0x0     0x0
0xffff6ae0ede0: 0x0     0x0
0xffff6ae0edf0: 0x0     0x0
0xffff6ae0ee00: 0x0     0x0
0xffff6ae0ee10: 0x0     0x0
0xffff6ae0ee20: 0x0     0x0
0xffff6ae0ee30: 0x0     0x0
0xffff6ae0ee40: 0x0     0x0
0xffff6ae0ee50: 0x0     0x0
0xffff6ae0ee60: 0x0     0x0
0xffff6ae0ee70: 0x0     0x0
0xffff6ae0ee80: 0x0     0x0
0xffff6ae0ee90: 0x0     0x0
0xffff6ae0eea0: 0x0     0x0
0xffff6ae0eeb0: 0x0     0x0
0xffff6ae0eec0: 0x0     0x0
0xffff6ae0eed0: 0x0     0x0
0xffff6ae0eee0: 0x0     0x0
0xffff6ae0eef0: 0x0     0x0
0xffff6ae0ef00: 0x0     0x0
0xffff6ae0ef10: 0x0     0x0
0xffff6ae0ef20: 0x0     0x0
0xffff6ae0ef30: 0x0     0x0
0xffff6ae0ef40: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xffff6ae0ef50: 0x0     0x0
0xffff6ae0ef60: 0x0     0x0
0xffff6ae0ef70: 0x0     0x0
0xffff6ae0ef80: 0x0     0x0
0xffff6ae0ef90: 0x0     0x0
```

```
0xffff6ae0efa0: 0x0     0x0
0xffff6ae0efb0: 0x0     0x0
0xffff6ae0efc0: 0x0     0x0
0xffff6ae0efd0: 0x0     0x0
0xffff6ae0efe0: 0x0     0x0
0xffff6ae0eff0: 0x0     0x0
0xffff6ae0f000: 0x0     0x0
0xffff6ae0f010: 0x0     0x0
0xffff6ae0f020: 0x0     0x0
0xffff6ae0f030: 0x0     0x0
0xffff6ae0f040: 0x0     0x0
0xffff6ae0f050: 0x0     0x0
0xffff6ae0f060: 0x0     0x0
0xffff6ae0f070: 0x0     0x0
0xffff6ae0f080: 0x1     0x0
0xffff6ae0f090: 0x0     0x0
0xffff6ae0f0a0: 0x0     0x0
0xffff6ae0f0b0: 0x0     0x0
0xffff6ae0f0c0: 0x0     0x0
0xffff6ae0f0d0: 0x0     0x0
0xffff6ae0f0e0: 0x0     0x0
0xffff6ae0f0f0: 0x0     0x0
0xffff6ae0f100: 0x0     0x0
0xffff6ae0f110: 0x0     0x0
0xffff6ae0f120: 0x0     0x0
0xffff6ae0f130: 0x0     0x0
0xffff6ae0f140: 0x4d0050 <stack_used>   0xffff6a5ff140
0xffff6ae0f150: 0x4bc100004bc2  0xffff6ae0f160
0xffff6ae0f160: 0xffff6ae0f160  0xfffffffffffffe0
0xffff6ae0f170: 0x0     0x0
0xffff6ae0f180: 0xffff6ae0e8b0  0x0
0xffff6ae0f190: 0x0     0x0
0xffff6ae0f1a0: 0x0     0x0
0xffff6ae0f1b0: 0x0     0x0
0xffff6ae0f1c0: 0x0     0x0
0xffff6ae0f1d0: 0x0     0x0
0xffff6ae0f1e0: 0x0     0x0
0xffff6ae0f1f0: 0x0     0x0
0xffff6ae0f200: 0x0     0x0
0xffff6ae0f210: 0x0     0x0
0xffff6ae0f220: 0x0     0x0
0xffff6ae0f230: 0x0     0x0
0xffff6ae0f240: 0x0     0x0
0xffff6ae0f250: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xffff6ae0f260: 0x0     0x0
0xffff6ae0f270: 0x0     0x0
0xffff6ae0f280: 0x0     0x0
0xffff6ae0f290: 0x0     0x0
0xffff6ae0f2a0: 0x0     0x0
0xffff6ae0f2b0: 0x0     0x0
0xffff6ae0f2c0: 0x0     0x0
0xffff6ae0f2d0: 0x0     0x0
0xffff6ae0f2e0: 0x0     0x0
0xffff6ae0f2f0: 0x0     0x0
0xffff6ae0f300: 0x0     0x0
0xffff6ae0f310: 0x0     0x0
0xffff6ae0f320: 0x0     0x0
0xffff6ae0f330: 0x0     0x0
0xffff6ae0f340: 0x0     0x0
```

```
0xffff6ae0f350: 0x0     0x0
0xffff6ae0f360: 0x0     0x0
0xffff6ae0f370: 0x0     0x0
0xffff6ae0f380: 0x0     0x0
0xffff6ae0f390: 0xffff6ae0f190  0x0
0xffff6ae0f3a0: 0x0     0x0
0xffff6ae0f3b0: 0x0     0x0
0xffff6ae0f3c0: 0x0     0x0
0xffff6ae0f3d0: 0x0     0x0
0xffff6ae0f3e0: 0x0     0x0
0xffff6ae0f3f0: 0x0     0x0
0xffff6ae0f400: 0x0     0x0
0xffff6ae0f410: 0x0     0x0
0xffff6ae0f420: 0x0     0x0
0xffff6ae0f430: 0x0     0x0
0xffff6ae0f440: 0x0     0x0
0xffff6ae0f450: 0x0     0x0
0xffff6ae0f460: 0x0     0x0
0xffff6ae0f470: 0x0     0x0
0xffff6ae0f480: 0x0     0x0
0xffff6ae0f490: 0x0     0x0
0xffff6ae0f4a0: 0x0     0x0
0xffff6ae0f4b0: 0x0     0x403298 <thread_one>
0xffff6ae0f4c0: 0x0     0x0
0xffff6ae0f4d0: 0x0     0x0
0xffff6ae0f4e0: 0x0     0x0
0xffff6ae0f4f0: 0x0     0x0
0xffff6ae0f500: 0x0     0x0
0xffff6ae0f510: 0xffff6a600000  0x810000
0xffff6ae0f520: 0x10000 0x10000
0xffff6ae0f530: 0x0     0x0
0xffff6ae0f540: 0x0     0x0
0xffff6ae0f550: 0x0     0x0
0xffff6ae0f560: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xffff6ae0f570: 0x0     0x0
0xffff6ae0f580: 0x0     0x0
0xffff6ae0f590: 0x0     0x0
0xffff6ae0f5a0: 0x0     0x0
0xffff6ae0f5b0: 0x0     0x0
0xffff6ae0f5c0: 0x0     0x0
0xffff6ae0f5d0: 0x0     0x0
0xffff6ae0f5e0: 0x0     0x0
0xffff6ae0f5f0: 0x0     0x0
0xffff6ae0f600: 0x0     0x0
0xffff6ae0f610: 0x0     0x0
0xffff6ae0f620: 0x0     0x0
0xffff6ae0f630: 0x0     0x0
0xffff6ae0f640: 0x0     0x0
0xffff6ae0f650: 0x0     0x0
0xffff6ae0f660: 0x0     0x0
0xffff6ae0f670: 0x0     0x0
0xffff6ae0f680: 0x0     0x0
0xffff6ae0f690: 0x0     0x0
0xffff6ae0f6a0: 0x0     0x0
0xffff6ae0f6b0: 0x0     0x0
0xffff6ae0f6c0: 0x0     0x0
0xffff6ae0f6d0: 0x0     0x0
0xffff6ae0f6e0: 0x0     0x0
0xffff6ae0f6f0: 0x0     0x0
```

```
0xffff6ae0f700: 0x0     0x0
0xffff6ae0f710: 0x0     0x0
0xffff6ae0f720: 0x0     0x0
0xffff6ae0f730: 0x0     0x0
0xffff6ae0f740: 0x0     0x0
0xffff6ae0f750: 0x0     0x0
0xffff6ae0f760: 0x0     0x0
0xffff6ae0f770: 0x30aa1d80      0x0
0xffff6ae0f780: 0xffff6ae0f538  0x4d13c0 <_nl_global_locale>
0xffff6ae0f790: 0x4d13c0 <_nl_global_locale>    0x4d13e0 <_nl_global_locale+32>
0xffff6ae0f7a0: 0x4d13c8 <_nl_global_locale+8>  0x0
0xffff6ae0f7b0: 0x48d280 <_nl_C_LC_CTYPE_class+256>     0x48c980 <_nl_C_LC_CTYPE_toupper+512>
0xffff6ae0f7c0: 0x48c380 <_nl_C_LC_CTYPE_tolower+512>   0x0
0xffff6ae0f7d0: 0x0     0x0
0xffff6ae0f7e0: 0x0     0x0
0xffff6ae0f7f0: 0x0     0x0
0xffff6ae0f800: 0x0     0x0
0xffff6ae0f810: 0x0     0x0
0xffff6ae0f820: 0x0     0x0
0xffff6ae0f830: 0x0     0x0
0xffff6ae0f840: 0x0     0x0
0xffff6ae0f850: 0x0     0x0
0xffff6ae0f860: 0x0     0x0
0xffff6ae0f870: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xffff6ae0f880: 0x0     0x0
0xffff6ae0f890: 0x0     0x0
0xffff6ae0f8a0: 0x0     0x0
0xffff6ae0f8b0: 0x0     0x0
0xffff6ae0f8c0: 0x0     0x0
0xffff6ae0f8d0: 0x0     0x0
0xffff6ae0f8e0: 0x0     0x0
0xffff6ae0f8f0: 0x0     0x0
0xffff6ae0f900: 0x0     0x0
0xffff6ae0f910: 0x0     0x0
0xffff6ae0f920: 0x0     0x0
0xffff6ae0f930: 0x0     0x0
0xffff6ae0f940: 0x0     0x0
0xffff6ae0f950: 0x0     0x0
0xffff6ae0f960: 0x0     0x0
0xffff6ae0f970: 0x0     0x0
0xffff6ae0f980: 0x0     0x0
0xffff6ae0f990: 0x0     0x0
0xffff6ae0f9a0: 0x0     0x0
0xffff6ae0f9b0: 0x0     0x0
0xffff6ae0f9c0: 0x0     0x0
0xffff6ae0f9d0: 0x0     0x0
0xffff6ae0f9e0: 0x0     0x0
0xffff6ae0f9f0: 0x0     0x0
0xffff6ae0fa00: 0x0     0x0
0xffff6ae0fa10: 0x0     0x0
0xffff6ae0fa20: 0x0     0x0
0xffff6ae0fa30: 0x0     0x0
0xffff6ae0fa40: 0x0     0x0
0xffff6ae0fa50: 0x0     0x0
0xffff6ae0fa60: 0x0     0x0
0xffff6ae0fa70: 0x0     0x0
0xffff6ae0fa80: 0x0     0x0
0xffff6ae0fa90: 0x0     0x0
0xffff6ae0faa0: 0x0     0x0
```

```
0xffff6ae0fab0: 0x0     0x0
0xffff6ae0fac0: 0x0     0x0
0xffff6ae0fad0: 0x0     0x0
0xffff6ae0fae0: 0x0     0x0
0xffff6ae0faf0: 0x0     0x0
0xffff6ae0fb00: 0x0     0x0
0xffff6ae0fb10: 0x0     0x0
0xffff6ae0fb20: 0x0     0x0
0xffff6ae0fb30: 0x0     0x0
0xffff6ae0fb40: 0x0     0x0
0xffff6ae0fb50: 0x0     0x0
0xffff6ae0fb60: 0x0     0x0
0xffff6ae0fb70: 0x0     0x0
0xffff6ae0fb80: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xffff6ae0fb90: 0x0     0x0
0xffff6ae0fba0: 0x0     0x0
0xffff6ae0fbb0: 0x0     0x0
0xffff6ae0fbc0: 0x0     0x0
0xffff6ae0fbd0: 0x0     0x0
0xffff6ae0fbe0: 0x0     0x0
0xffff6ae0fbf0: 0x0     0x0
0xffff6ae0fc00: 0x0     0x0
0xffff6ae0fc10: 0x0     0x0
0xffff6ae0fc20: 0x0     0x0
0xffff6ae0fc30: 0x0     0x0
0xffff6ae0fc40: 0x0     0x0
0xffff6ae0fc50: 0x0     0x0
0xffff6ae0fc60: 0x0     0x0
0xffff6ae0fc70: 0x0     0x0
0xffff6ae0fc80: 0x0     0x0
0xffff6ae0fc90: 0x0     0x0
0xffff6ae0fca0: 0x0     0x0
0xffff6ae0fcb0: 0x0     0x0
0xffff6ae0fcc0: 0x0     0x0
0xffff6ae0fcd0: 0x0     0x0
0xffff6ae0fce0: 0x0     0x0
0xffff6ae0fcf0: 0x0     0x0
0xffff6ae0fd00: 0x0     0x0
0xffff6ae0fd10: 0x0     0x0
0xffff6ae0fd20: 0x0     0x0
0xffff6ae0fd30: 0x0     0x0
0xffff6ae0fd40: 0x0     0x0
0xffff6ae0fd50: 0x0     0x0
0xffff6ae0fd60: 0x0     0x0
0xffff6ae0fd70: 0x0     0x0
0xffff6ae0fd80: 0x0     0x0
0xffff6ae0fd90: 0x0     0x0
0xffff6ae0fda0: 0x0     0x0
0xffff6ae0fdb0: 0x0     0x0
0xffff6ae0fdc0: 0x0     0x0
0xffff6ae0fdd0: 0x0     0x0
0xffff6ae0fde0: 0x0     0x0
0xffff6ae0fdf0: 0x0     0x0
0xffff6ae0fe00: 0x0     0x0
0xffff6ae0fe10: 0x0     0x0
0xffff6ae0fe20: 0x0     0x0
0xffff6ae0fe30: 0x0     0x0
0xffff6ae0fe40: 0x0     0x0
0xffff6ae0fe50: 0x0     0x0
```

```
0xffff6ae0fe60: 0x0     0x0
0xffff6ae0fe70: 0x0     0x0
0xffff6ae0fe80: 0x0     0x0
0xffff6ae0fe90: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xffff6ae0fea0: 0x0     0x0
0xffff6ae0feb0: 0x0     0x0
0xffff6ae0fec0: 0x0     0x0
0xffff6ae0fed0: 0x0     0x0
0xffff6ae0fee0: 0x0     0x0
0xffff6ae0fef0: 0x0     0x0
0xffff6ae0ff00: 0x0     0x0
0xffff6ae0ff10: 0x0     0x0
0xffff6ae0ff20: 0x0     0x0
0xffff6ae0ff30: 0x0     0x0
0xffff6ae0ff40: 0x0     0x0
0xffff6ae0ff50: 0x0     0x0
0xffff6ae0ff60: 0x0     0x0
0xffff6ae0ff70: 0x0     0x0
0xffff6ae0ff80: 0x0     0x0
0xffff6ae0ff90: 0x0     0x0
0xffff6ae0ffa0: 0x0     0x0
0xffff6ae0ffb0: 0x0     0x0
0xffff6ae0ffc0: 0x0     0x0
0xffff6ae0ffd0: 0x0     0x0
0xffff6ae0ffe0: 0x0     0x0
0xffff6ae0fff0: 0x0     0x0
```

8. See that the reconstruction of the stack trace is possible because of the standard function prologue and epilogue:

```
[...]
0xffff6ae0e1b0: 0xffff6ae0e3d0  0x403244 <procF+92>
0xffff6ae0e3d0: 0xffff6ae0e5f0  0x403244 <procF+92>
0xffff6ae0e5f0: 0xffff6ae0e810  0x403260 <procE+16>
0xffff6ae0e810: 0xffff6ae0e820  0x40327c <bar_one+20>
0xffff6ae0e820: 0xffff6ae0e830  0x403290 <foo_one+12>
0xffff6ae0e830: 0xffff6ae0e840  0x4032a8 <thread_one+16>
0xffff6ae0e840: 0xffff6ae0e860  0x404cd4 <start_thread+180>
```

```
(gdb) disass procF
Dump of assembler code for function procF:
   0x00000000004031e8 <+0>:     sub     sp, sp, #0x210
   0x00000000004031ec <+4>:     stp     x29, x30, [sp, #-16]!
   0x00000000004031f0 <+8>:     mov     x29, sp
   0x00000000004031f4 <+12>:    add     x1, x29, #0x1c
   0x00000000004031f8 <+16>:    str     w0, [x1]
   0x00000000004031fc <+20>:    add     x0, x29, #0x20
   0x0000000000403200 <+24>:    mov     x2, #0x200                      // #512
   0x0000000000403204 <+28>:    mov     w1, #0x0                        // #0
   0x0000000000403208 <+32>:    bl      0x400290
   0x000000000040320c <+36>:    add     x0, x29, #0x20
   0x0000000000403210 <+40>:    mov     w1, #0xffffffff                 // #-1
   0x0000000000403214 <+44>:    str     w1, [x0]
   0x0000000000403218 <+48>:    add     x0, x29, #0x1c
   0x000000000040321c <+52>:    ldr     w0, [x0]
   0x0000000000403220 <+56>:    add     w1, w0, #0x1
   0x0000000000403224 <+60>:    add     x0, x29, #0x20
   0x0000000000403228 <+64>:    str     w1, [x0, #8]
   0x000000000040322c <+68>:    add     x0, x29, #0x20
```

```
   0x0000000000403230 <+72>:    mov     w1, #0xffffffff                 // #-1
   0x0000000000403234 <+76>:    str     w1, [x0, #16]
   0x0000000000403238 <+80>:    add     x0, x29, #0x20
   0x000000000040323c <+84>:    ldr     w0, [x0, #8]
   0x0000000000403240 <+88>:    bl      0x4031e8 <procF>
=> 0x0000000000403244 <+92>:    ldp     x29, x30, [sp], #16
   0x0000000000403248 <+96>:    add     sp, sp, #0x210
   0x000000000040324c <+100>:   ret
End of assembler dump.
```

9. Use the back trace command variant to get to the bottom of the stack trace:

```
(gdb) bt -20
#15395 0x0000000000403244 in procF ()
#15396 0x0000000000403244 in procF ()
#15397 0x0000000000403244 in procF ()
#15398 0x0000000000403244 in procF ()
#15399 0x0000000000403244 in procF ()
#15400 0x0000000000403244 in procF ()
#15401 0x0000000000403244 in procF ()
#15402 0x0000000000403244 in procF ()
#15403 0x0000000000403244 in procF ()
#15404 0x0000000000403244 in procF ()
#15405 0x0000000000403244 in procF ()
#15406 0x0000000000403244 in procF ()
#15407 0x0000000000403244 in procF ()
#15408 0x0000000000403244 in procF ()
#15409 0x0000000000403260 in procE ()
#15410 0x000000000040327c in bar_one ()
#15411 0x0000000000403290 in foo_one ()
#15412 0x00000000004032a8 in thread_one ()
#15413 0x0000000000404cd4 in start_thread ()
#15414 0x0000000000429c20 in thread_start ()
```

Exercise A6 (A64, WinDbg Preview)

Goal: Learn how to identify stack overflow, stack boundaries, reconstruct stack trace.

Patterns: Stack Overflow (User Mode).

1. Launch WinDbg Preview.

2. Load *core.19393* dump file from the A64\App6 folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App6\core.19393]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
(4bc1.4bc2): Signal SIGSEGV (Segmentation fault) code SEGV_ACCERR (Invalid permissions for
mapped object) at 0xffff6a60fff0*** WARNING: Unable to verify timestamp for App6
App6+0x31ec:
00000000`004031ec a9bf7bfd stp         fp,lr,[sp,#-0x10]!
```

3. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App6\App6.log
Opened log file 'C:\ALCDA2\A64\App6\App6.log'
```

4. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App6\
Symbol search path is: srv*;C:\ALCDA2\A64\App6\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app6\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App6\
*** WARNING: Unable to verify timestamp for App6
```

```
0:000> .reload
..
*** WARNING: Unable to verify timestamp for App6

************* Symbol Loading Error Summary **************
Module name            Error
App6                   The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

5. List threads:

```
0:000> ~*k 1

Unable to get thread data for thread 0
.  0  Id: 4bc1.4bc2 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`6a610000 00000000`00403244     App6!procF+0x4

Unable to get thread data for thread 1
   1  Id: 4bc1.4bc1 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`e3b450f0 00000000`00424d74     App6!_libc_nanosleep+0x24

Unable to get thread data for thread 2
   2  Id: 4bc1.4bc6 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`68dce5f0 00000000`00424d74     App6!_libc_nanosleep+0x24

Unable to get thread data for thread 3
   3  Id: 4bc1.4bc5 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`695de5f0 00000000`00424d74     App6!_libc_nanosleep+0x24

Unable to get thread data for thread 4
   4  Id: 4bc1.4bc4 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`69dee5f0 00000000`00424d74     App6!_libc_nanosleep+0x28

Unable to get thread data for thread 5
   5  Id: 4bc1.4bc3 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`6a5fe5f0 00000000`00424d74     App6!_libc_nanosleep+0x24
```

6. If we try to print the problem stack trace, we get 256 frames before stopping:

```
0:000> k
 # Child-SP          RetAddr               Call Site
00 0000ffff`6a610000 00000000`00403244     App6!procF+0x4
01 0000ffff`6a610210 00000000`00403244     App6!procF+0x5c
02 0000ffff`6a610430 00000000`00403244     App6!procF+0x5c
03 0000ffff`6a610650 00000000`00403244     App6!procF+0x5c
04 0000ffff`6a610870 00000000`00403244     App6!procF+0x5c
05 0000ffff`6a610a90 00000000`00403244     App6!procF+0x5c
06 0000ffff`6a610cb0 00000000`00403244     App6!procF+0x5c
```

```
07 0000ffff`6a610ed0 00000000`00403244     App6!procF+0x5c
08 0000ffff`6a6110f0 00000000`00403244     App6!procF+0x5c
09 0000ffff`6a611310 00000000`00403244     App6!procF+0x5c
0a 0000ffff`6a611530 00000000`00403244     App6!procF+0x5c
0b 0000ffff`6a611750 00000000`00403244     App6!procF+0x5c
0c 0000ffff`6a611970 00000000`00403244     App6!procF+0x5c
0d 0000ffff`6a611b90 00000000`00403244     App6!procF+0x5c
0e 0000ffff`6a611db0 00000000`00403244     App6!procF+0x5c
0f 0000ffff`6a611fd0 00000000`00403244     App6!procF+0x5c
10 0000ffff`6a6121f0 00000000`00403244     App6!procF+0x5c
11 0000ffff`6a612410 00000000`00403244     App6!procF+0x5c
12 0000ffff`6a612630 00000000`00403244     App6!procF+0x5c
13 0000ffff`6a612850 00000000`00403244     App6!procF+0x5c
14 0000ffff`6a612a70 00000000`00403244     App6!procF+0x5c
15 0000ffff`6a612c90 00000000`00403244     App6!procF+0x5c
16 0000ffff`6a612eb0 00000000`00403244     App6!procF+0x5c
17 0000ffff`6a6130d0 00000000`00403244     App6!procF+0x5c
18 0000ffff`6a6132f0 00000000`00403244     App6!procF+0x5c
19 0000ffff`6a613510 00000000`00403244     App6!procF+0x5c
1a 0000ffff`6a613730 00000000`00403244     App6!procF+0x5c
1b 0000ffff`6a613950 00000000`00403244     App6!procF+0x5c
1c 0000ffff`6a613b70 00000000`00403244     App6!procF+0x5c
1d 0000ffff`6a613d90 00000000`00403244     App6!procF+0x5c
1e 0000ffff`6a613fb0 00000000`00403244     App6!procF+0x5c
1f 0000ffff`6a6141d0 00000000`00403244     App6!procF+0x5c
20 0000ffff`6a6143f0 00000000`00403244     App6!procF+0x5c
21 0000ffff`6a614610 00000000`00403244     App6!procF+0x5c
22 0000ffff`6a614830 00000000`00403244     App6!procF+0x5c
23 0000ffff`6a614a50 00000000`00403244     App6!procF+0x5c
24 0000ffff`6a614c70 00000000`00403244     App6!procF+0x5c
25 0000ffff`6a614e90 00000000`00403244     App6!procF+0x5c
26 0000ffff`6a6150b0 00000000`00403244     App6!procF+0x5c
27 0000ffff`6a6152d0 00000000`00403244     App6!procF+0x5c
28 0000ffff`6a6154f0 00000000`00403244     App6!procF+0x5c
29 0000ffff`6a615710 00000000`00403244     App6!procF+0x5c
2a 0000ffff`6a615930 00000000`00403244     App6!procF+0x5c
2b 0000ffff`6a615b50 00000000`00403244     App6!procF+0x5c
2c 0000ffff`6a615d70 00000000`00403244     App6!procF+0x5c
2d 0000ffff`6a615f90 00000000`00403244     App6!procF+0x5c
2e 0000ffff`6a6161b0 00000000`00403244     App6!procF+0x5c
2f 0000ffff`6a6163d0 00000000`00403244     App6!procF+0x5c
30 0000ffff`6a6165f0 00000000`00403244     App6!procF+0x5c
31 0000ffff`6a616810 00000000`00403244     App6!procF+0x5c
32 0000ffff`6a616a30 00000000`00403244     App6!procF+0x5c
33 0000ffff`6a616c50 00000000`00403244     App6!procF+0x5c
34 0000ffff`6a616e70 00000000`00403244     App6!procF+0x5c
35 0000ffff`6a617090 00000000`00403244     App6!procF+0x5c
36 0000ffff`6a6172b0 00000000`00403244     App6!procF+0x5c
37 0000ffff`6a6174d0 00000000`00403244     App6!procF+0x5c
38 0000ffff`6a6176f0 00000000`00403244     App6!procF+0x5c
39 0000ffff`6a617910 00000000`00403244     App6!procF+0x5c
3a 0000ffff`6a617b30 00000000`00403244     App6!procF+0x5c
3b 0000ffff`6a617d50 00000000`00403244     App6!procF+0x5c
3c 0000ffff`6a617f70 00000000`00403244     App6!procF+0x5c
3d 0000ffff`6a618190 00000000`00403244     App6!procF+0x5c
3e 0000ffff`6a6183b0 00000000`00403244     App6!procF+0x5c
3f 0000ffff`6a6185d0 00000000`00403244     App6!procF+0x5c
40 0000ffff`6a6187f0 00000000`00403244     App6!procF+0x5c
41 0000ffff`6a618a10 00000000`00403244     App6!procF+0x5c
42 0000ffff`6a618c30 00000000`00403244     App6!procF+0x5c
```

```
43 0000ffff`6a618e50 00000000`00403244     App6!procF+0x5c
44 0000ffff`6a619070 00000000`00403244     App6!procF+0x5c
45 0000ffff`6a619290 00000000`00403244     App6!procF+0x5c
46 0000ffff`6a6194b0 00000000`00403244     App6!procF+0x5c
47 0000ffff`6a6196d0 00000000`00403244     App6!procF+0x5c
48 0000ffff`6a6198f0 00000000`00403244     App6!procF+0x5c
49 0000ffff`6a619b10 00000000`00403244     App6!procF+0x5c
4a 0000ffff`6a619d30 00000000`00403244     App6!procF+0x5c
4b 0000ffff`6a619f50 00000000`00403244     App6!procF+0x5c
4c 0000ffff`6a61a170 00000000`00403244     App6!procF+0x5c
4d 0000ffff`6a61a390 00000000`00403244     App6!procF+0x5c
4e 0000ffff`6a61a5b0 00000000`00403244     App6!procF+0x5c
4f 0000ffff`6a61a7d0 00000000`00403244     App6!procF+0x5c
50 0000ffff`6a61a9f0 00000000`00403244     App6!procF+0x5c
51 0000ffff`6a61ac10 00000000`00403244     App6!procF+0x5c
52 0000ffff`6a61ae30 00000000`00403244     App6!procF+0x5c
53 0000ffff`6a61b050 00000000`00403244     App6!procF+0x5c
54 0000ffff`6a61b270 00000000`00403244     App6!procF+0x5c
55 0000ffff`6a61b490 00000000`00403244     App6!procF+0x5c
56 0000ffff`6a61b6b0 00000000`00403244     App6!procF+0x5c
57 0000ffff`6a61b8d0 00000000`00403244     App6!procF+0x5c
58 0000ffff`6a61baf0 00000000`00403244     App6!procF+0x5c
59 0000ffff`6a61bd10 00000000`00403244     App6!procF+0x5c
5a 0000ffff`6a61bf30 00000000`00403244     App6!procF+0x5c
5b 0000ffff`6a61c150 00000000`00403244     App6!procF+0x5c
5c 0000ffff`6a61c370 00000000`00403244     App6!procF+0x5c
5d 0000ffff`6a61c590 00000000`00403244     App6!procF+0x5c
5e 0000ffff`6a61c7b0 00000000`00403244     App6!procF+0x5c
5f 0000ffff`6a61c9d0 00000000`00403244     App6!procF+0x5c
60 0000ffff`6a61cbf0 00000000`00403244     App6!procF+0x5c
61 0000ffff`6a61ce10 00000000`00403244     App6!procF+0x5c
62 0000ffff`6a61d030 00000000`00403244     App6!procF+0x5c
63 0000ffff`6a61d250 00000000`00403244     App6!procF+0x5c
64 0000ffff`6a61d470 00000000`00403244     App6!procF+0x5c
65 0000ffff`6a61d690 00000000`00403244     App6!procF+0x5c
66 0000ffff`6a61d8b0 00000000`00403244     App6!procF+0x5c
67 0000ffff`6a61dad0 00000000`00403244     App6!procF+0x5c
68 0000ffff`6a61dcf0 00000000`00403244     App6!procF+0x5c
69 0000ffff`6a61df10 00000000`00403244     App6!procF+0x5c
6a 0000ffff`6a61e130 00000000`00403244     App6!procF+0x5c
6b 0000ffff`6a61e350 00000000`00403244     App6!procF+0x5c
6c 0000ffff`6a61e570 00000000`00403244     App6!procF+0x5c
6d 0000ffff`6a61e790 00000000`00403244     App6!procF+0x5c
6e 0000ffff`6a61e9b0 00000000`00403244     App6!procF+0x5c
6f 0000ffff`6a61ebd0 00000000`00403244     App6!procF+0x5c
70 0000ffff`6a61edf0 00000000`00403244     App6!procF+0x5c
71 0000ffff`6a61f010 00000000`00403244     App6!procF+0x5c
72 0000ffff`6a61f230 00000000`00403244     App6!procF+0x5c
73 0000ffff`6a61f450 00000000`00403244     App6!procF+0x5c
74 0000ffff`6a61f670 00000000`00403244     App6!procF+0x5c
75 0000ffff`6a61f890 00000000`00403244     App6!procF+0x5c
76 0000ffff`6a61fab0 00000000`00403244     App6!procF+0x5c
77 0000ffff`6a61fcd0 00000000`00403244     App6!procF+0x5c
78 0000ffff`6a61fef0 00000000`00403244     App6!procF+0x5c
79 0000ffff`6a620110 00000000`00403244     App6!procF+0x5c
7a 0000ffff`6a620330 00000000`00403244     App6!procF+0x5c
7b 0000ffff`6a620550 00000000`00403244     App6!procF+0x5c
7c 0000ffff`6a620770 00000000`00403244     App6!procF+0x5c
7d 0000ffff`6a620990 00000000`00403244     App6!procF+0x5c
7e 0000ffff`6a620bb0 00000000`00403244     App6!procF+0x5c
```

```
7f 0000ffff`6a620dd0 00000000`00403244     App6!procF+0x5c
80 0000ffff`6a620ff0 00000000`00403244     App6!procF+0x5c
81 0000ffff`6a621210 00000000`00403244     App6!procF+0x5c
82 0000ffff`6a621430 00000000`00403244     App6!procF+0x5c
83 0000ffff`6a621650 00000000`00403244     App6!procF+0x5c
84 0000ffff`6a621870 00000000`00403244     App6!procF+0x5c
85 0000ffff`6a621a90 00000000`00403244     App6!procF+0x5c
86 0000ffff`6a621cb0 00000000`00403244     App6!procF+0x5c
87 0000ffff`6a621ed0 00000000`00403244     App6!procF+0x5c
88 0000ffff`6a6220f0 00000000`00403244     App6!procF+0x5c
89 0000ffff`6a622310 00000000`00403244     App6!procF+0x5c
8a 0000ffff`6a622530 00000000`00403244     App6!procF+0x5c
8b 0000ffff`6a622750 00000000`00403244     App6!procF+0x5c
8c 0000ffff`6a622970 00000000`00403244     App6!procF+0x5c
8d 0000ffff`6a622b90 00000000`00403244     App6!procF+0x5c
8e 0000ffff`6a622db0 00000000`00403244     App6!procF+0x5c
8f 0000ffff`6a622fd0 00000000`00403244     App6!procF+0x5c
90 0000ffff`6a6231f0 00000000`00403244     App6!procF+0x5c
91 0000ffff`6a623410 00000000`00403244     App6!procF+0x5c
92 0000ffff`6a623630 00000000`00403244     App6!procF+0x5c
93 0000ffff`6a623850 00000000`00403244     App6!procF+0x5c
94 0000ffff`6a623a70 00000000`00403244     App6!procF+0x5c
95 0000ffff`6a623c90 00000000`00403244     App6!procF+0x5c
96 0000ffff`6a623eb0 00000000`00403244     App6!procF+0x5c
97 0000ffff`6a6240d0 00000000`00403244     App6!procF+0x5c
98 0000ffff`6a6242f0 00000000`00403244     App6!procF+0x5c
99 0000ffff`6a624510 00000000`00403244     App6!procF+0x5c
9a 0000ffff`6a624730 00000000`00403244     App6!procF+0x5c
9b 0000ffff`6a624950 00000000`00403244     App6!procF+0x5c
9c 0000ffff`6a624b70 00000000`00403244     App6!procF+0x5c
9d 0000ffff`6a624d90 00000000`00403244     App6!procF+0x5c
9e 0000ffff`6a624fb0 00000000`00403244     App6!procF+0x5c
9f 0000ffff`6a6251d0 00000000`00403244     App6!procF+0x5c
a0 0000ffff`6a6253f0 00000000`00403244     App6!procF+0x5c
a1 0000ffff`6a625610 00000000`00403244     App6!procF+0x5c
a2 0000ffff`6a625830 00000000`00403244     App6!procF+0x5c
a3 0000ffff`6a625a50 00000000`00403244     App6!procF+0x5c
a4 0000ffff`6a625c70 00000000`00403244     App6!procF+0x5c
a5 0000ffff`6a625e90 00000000`00403244     App6!procF+0x5c
a6 0000ffff`6a6260b0 00000000`00403244     App6!procF+0x5c
a7 0000ffff`6a6262d0 00000000`00403244     App6!procF+0x5c
a8 0000ffff`6a6264f0 00000000`00403244     App6!procF+0x5c
a9 0000ffff`6a626710 00000000`00403244     App6!procF+0x5c
aa 0000ffff`6a626930 00000000`00403244     App6!procF+0x5c
ab 0000ffff`6a626b50 00000000`00403244     App6!procF+0x5c
ac 0000ffff`6a626d70 00000000`00403244     App6!procF+0x5c
ad 0000ffff`6a626f90 00000000`00403244     App6!procF+0x5c
ae 0000ffff`6a6271b0 00000000`00403244     App6!procF+0x5c
af 0000ffff`6a6273d0 00000000`00403244     App6!procF+0x5c
b0 0000ffff`6a6275f0 00000000`00403244     App6!procF+0x5c
b1 0000ffff`6a627810 00000000`00403244     App6!procF+0x5c
b2 0000ffff`6a627a30 00000000`00403244     App6!procF+0x5c
b3 0000ffff`6a627c50 00000000`00403244     App6!procF+0x5c
b4 0000ffff`6a627e70 00000000`00403244     App6!procF+0x5c
b5 0000ffff`6a628090 00000000`00403244     App6!procF+0x5c
b6 0000ffff`6a6282b0 00000000`00403244     App6!procF+0x5c
b7 0000ffff`6a6284d0 00000000`00403244     App6!procF+0x5c
b8 0000ffff`6a6286f0 00000000`00403244     App6!procF+0x5c
b9 0000ffff`6a628910 00000000`00403244     App6!procF+0x5c
ba 0000ffff`6a628b30 00000000`00403244     App6!procF+0x5c
```

```
bb 0000ffff`6a628d50 00000000`00403244     App6!procF+0x5c
bc 0000ffff`6a628f70 00000000`00403244     App6!procF+0x5c
bd 0000ffff`6a629190 00000000`00403244     App6!procF+0x5c
be 0000ffff`6a6293b0 00000000`00403244     App6!procF+0x5c
bf 0000ffff`6a6295d0 00000000`00403244     App6!procF+0x5c
c0 0000ffff`6a6297f0 00000000`00403244     App6!procF+0x5c
c1 0000ffff`6a629a10 00000000`00403244     App6!procF+0x5c
c2 0000ffff`6a629c30 00000000`00403244     App6!procF+0x5c
c3 0000ffff`6a629e50 00000000`00403244     App6!procF+0x5c
c4 0000ffff`6a62a070 00000000`00403244     App6!procF+0x5c
c5 0000ffff`6a62a290 00000000`00403244     App6!procF+0x5c
c6 0000ffff`6a62a4b0 00000000`00403244     App6!procF+0x5c
c7 0000ffff`6a62a6d0 00000000`00403244     App6!procF+0x5c
c8 0000ffff`6a62a8f0 00000000`00403244     App6!procF+0x5c
c9 0000ffff`6a62ab10 00000000`00403244     App6!procF+0x5c
ca 0000ffff`6a62ad30 00000000`00403244     App6!procF+0x5c
cb 0000ffff`6a62af50 00000000`00403244     App6!procF+0x5c
cc 0000ffff`6a62b170 00000000`00403244     App6!procF+0x5c
cd 0000ffff`6a62b390 00000000`00403244     App6!procF+0x5c
ce 0000ffff`6a62b5b0 00000000`00403244     App6!procF+0x5c
cf 0000ffff`6a62b7d0 00000000`00403244     App6!procF+0x5c
d0 0000ffff`6a62b9f0 00000000`00403244     App6!procF+0x5c
d1 0000ffff`6a62bc10 00000000`00403244     App6!procF+0x5c
d2 0000ffff`6a62be30 00000000`00403244     App6!procF+0x5c
d3 0000ffff`6a62c050 00000000`00403244     App6!procF+0x5c
d4 0000ffff`6a62c270 00000000`00403244     App6!procF+0x5c
d5 0000ffff`6a62c490 00000000`00403244     App6!procF+0x5c
d6 0000ffff`6a62c6b0 00000000`00403244     App6!procF+0x5c
d7 0000ffff`6a62c8d0 00000000`00403244     App6!procF+0x5c
d8 0000ffff`6a62caf0 00000000`00403244     App6!procF+0x5c
d9 0000ffff`6a62cd10 00000000`00403244     App6!procF+0x5c
da 0000ffff`6a62cf30 00000000`00403244     App6!procF+0x5c
db 0000ffff`6a62d150 00000000`00403244     App6!procF+0x5c
dc 0000ffff`6a62d370 00000000`00403244     App6!procF+0x5c
dd 0000ffff`6a62d590 00000000`00403244     App6!procF+0x5c
de 0000ffff`6a62d7b0 00000000`00403244     App6!procF+0x5c
df 0000ffff`6a62d9d0 00000000`00403244     App6!procF+0x5c
e0 0000ffff`6a62dbf0 00000000`00403244     App6!procF+0x5c
e1 0000ffff`6a62de10 00000000`00403244     App6!procF+0x5c
e2 0000ffff`6a62e030 00000000`00403244     App6!procF+0x5c
e3 0000ffff`6a62e250 00000000`00403244     App6!procF+0x5c
e4 0000ffff`6a62e470 00000000`00403244     App6!procF+0x5c
e5 0000ffff`6a62e690 00000000`00403244     App6!procF+0x5c
e6 0000ffff`6a62e8b0 00000000`00403244     App6!procF+0x5c
e7 0000ffff`6a62ead0 00000000`00403244     App6!procF+0x5c
e8 0000ffff`6a62ecf0 00000000`00403244     App6!procF+0x5c
e9 0000ffff`6a62ef10 00000000`00403244     App6!procF+0x5c
ea 0000ffff`6a62f130 00000000`00403244     App6!procF+0x5c
eb 0000ffff`6a62f350 00000000`00403244     App6!procF+0x5c
ec 0000ffff`6a62f570 00000000`00403244     App6!procF+0x5c
ed 0000ffff`6a62f790 00000000`00403244     App6!procF+0x5c
ee 0000ffff`6a62f9b0 00000000`00403244     App6!procF+0x5c
ef 0000ffff`6a62fbd0 00000000`00403244     App6!procF+0x5c
f0 0000ffff`6a62fdf0 00000000`00403244     App6!procF+0x5c
f1 0000ffff`6a630010 00000000`00403244     App6!procF+0x5c
f2 0000ffff`6a630230 00000000`00403244     App6!procF+0x5c
f3 0000ffff`6a630450 00000000`00403244     App6!procF+0x5c
f4 0000ffff`6a630670 00000000`00403244     App6!procF+0x5c
f5 0000ffff`6a630890 00000000`00403244     App6!procF+0x5c
f6 0000ffff`6a630ab0 00000000`00403244     App6!procF+0x5c
```

```
f7 0000ffff`6a630cd0 00000000`00403244     App6!procF+0x5c
f8 0000ffff`6a630ef0 00000000`00403244     App6!procF+0x5c
f9 0000ffff`6a631110 00000000`00403244     App6!procF+0x5c
fa 0000ffff`6a631330 00000000`00403244     App6!procF+0x5c
fb 0000ffff`6a631550 00000000`00403244     App6!procF+0x5c
fc 0000ffff`6a631770 00000000`00403244     App6!procF+0x5c
fd 0000ffff`6a631990 00000000`00403244     App6!procF+0x5c
fe 0000ffff`6a631bb0 00000000`00403244     App6!procF+0x5c
ff 0000ffff`6a631dd0 00000000`00403244     App6!procF+0x5c
```

Note: We don't see that start frames, and it looks like a stack overflow.

7. Check if this is a stack overflow indeed. The stack region can be identified from *App6.pmap.19393* from the thread number. Since the problem thread has TID=PID+1 (Id: 4bc1.4bc2), it should be located just below the main stack region:

```
19393:   ./App6
0000000000400000    768K r-x-- App6
00000000004c0000    128K rw--- App6
0000000030aa0000    256K rw---   [ anon ]
0000ffff685c0000     64K -----   [ anon ]
0000ffff685d0000   8192K rw---   [ anon ]
0000ffff68dd0000     64K -----   [ anon ]
0000ffff68de0000   8192K rw---   [ anon ]
0000ffff695e0000     64K -----   [ anon ]
0000ffff695f0000   8192K rw---   [ anon ]
0000ffff69df0000     64K -----   [ anon ]
0000ffff69e00000   8192K rw---   [ anon ]
0000ffff6a600000     64K -----   [ anon ]
0000ffff6a610000   8192K rw---   [ anon ]
0000ffff6ae10000     64K r----   [ anon ]
0000ffff6ae20000     64K r-x--   [ anon ]
0000ffffe3b20000    192K rw---   [ stack ]
 total            42752K
```

8. Check that manually based on the stack pointer value and section boundary addresses:

```
0:000> r sp
sp=0000ffff6a610000

0:000> dp sp - 10
0000ffff`6a60fff0  00000000`00000000 00000000`00000000
0000ffff`6a610000  00000000`00000000 00000000`00000000
0000ffff`6a610010  00000000`00000000 00000000`00000000
0000ffff`6a610020  00000000`00000000 00000000`00000000
0000ffff`6a610030  00000000`00000000 00000000`00000000
0000ffff`6a610040  00000000`00000000 00000000`00000000
0000ffff`6a610050  00000000`00000000 00000000`00000000
0000ffff`6a610060  00000000`00000000 00000000`00000000
```

Note: The stack pointer points to the start of the stack region. The addresses below it should be inaccessible at runtime. However, the committed pages were included in the crash dump, and we see zeroes since WinDbg can read it.

```
0:000> !address

Mapping file section regions...
Mapping module regions...

        BaseAddress      EndAddress+1        RegionSize     Type       State                 Protect             Usage
----------------------------------------------------------------------------------------------------------------------
+        0`00000000        0`00400000        0`00400000                                                      <unknown>
+        0`00400000        0`00410000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ          Image      [App6;
"/home/opc/ALCDA2/App6/App6"]
+        0`00410000        0`004c0000        0`000b0000                                                      Image      [App6;
"/home/opc/ALCDA2/App6/App6"]
+        0`004c0000        0`004e0000        0`00020000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             Image      [App6;
"/home/opc/ALCDA2/App6/App6"]
+        0`004e0000        0`30aa0000        0`305c0000                                                      <unknown>
+        0`30aa0000        0`30ae0000        0`00040000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+        0`30ae0000     ffff`685c0000     ffff`37ae0000                                                      <unknown>
+     ffff`685c0000     ffff`685d0000        0`00010000 MEM_PRIVATE MEM_COMMIT                             <unknown>  [................]
+     ffff`685d0000     ffff`68dd0000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     ffff`68dd0000     ffff`68de0000        0`00010000 MEM_PRIVATE MEM_COMMIT                             <unknown>  [................]
+     ffff`68de0000     ffff`695e0000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     ffff`695e0000     ffff`695f0000        0`00010000 MEM_PRIVATE MEM_COMMIT                             <unknown>  [................]
+     ffff`695f0000     ffff`69df0000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     ffff`69df0000     ffff`69e00000        0`00010000 MEM_PRIVATE MEM_COMMIT                             <unknown>  [................]
+     ffff`69e00000     ffff`6a600000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     ffff`6a600000     ffff`6a610000        0`00010000 MEM_PRIVATE MEM_COMMIT                             <unknown>  [................]
+     ffff`6a610000     ffff`6ae10000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     ffff`6ae10000     ffff`6ae20000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY              <unknown>  [.S......FH..z...]
+     ffff`6ae20000     ffff`6ae30000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ          Image      [linux_vdso_so; "linux-
vdso.so.1"]
+     ffff`6ae30000     ffff`e3b20000        0`78cf0000                                                      <unknown>
+     ffff`e3b20000     ffff`e3b50000        0`00030000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
```

9. Dump the bottom of the raw stack with symbols to see execution residue, such as thread startup:

```
0:000> dps ffff`6ae10000-2000 ffff`6ae10000
0000ffff`6ae0e000  00000000`00000000
0000ffff`6ae0e008  00000000`00000000
0000ffff`6ae0e010  00000000`00000000
0000ffff`6ae0e018  00000000`00000000
0000ffff`6ae0e020  00000000`00000000
0000ffff`6ae0e028  00000000`00000000
0000ffff`6ae0e030  00000000`00000000
0000ffff`6ae0e038  00000000`00000000
0000ffff`6ae0e040  00000000`00000000
0000ffff`6ae0e048  00000000`00000000
0000ffff`6ae0e050  00000000`00000000
0000ffff`6ae0e058  00000000`00000000
0000ffff`6ae0e060  00000000`00000000
0000ffff`6ae0e068  00000000`00000000
0000ffff`6ae0e070  00000000`00000000
0000ffff`6ae0e078  00000000`00000000
0000ffff`6ae0e080  00000000`00000000
0000ffff`6ae0e088  00000000`00000000
0000ffff`6ae0e090  00000000`00000000
0000ffff`6ae0e098  00000000`00000000
0000ffff`6ae0e0a0  00000000`00000000
0000ffff`6ae0e0a8  00000000`00000000
0000ffff`6ae0e0b0  00000000`00000000
0000ffff`6ae0e0b8  00000000`00000000
0000ffff`6ae0e0c0  00000000`00000000
0000ffff`6ae0e0c8  00000000`00000000
0000ffff`6ae0e0d0  00000000`00000000
0000ffff`6ae0e0d8  00000000`00000000
0000ffff`6ae0e0e0  00000000`00000000
0000ffff`6ae0e0e8  00000000`00000000
0000ffff`6ae0e0f0  00000000`00000000
0000ffff`6ae0e0f8  00000000`00000000
0000ffff`6ae0e100  00000000`00000000
0000ffff`6ae0e108  00000000`00000000
0000ffff`6ae0e110  00000000`00000000
0000ffff`6ae0e118  00000000`00000000
```

```
0000ffff`6ae0e120  00000000`00000000
0000ffff`6ae0e128  00000000`00000000
0000ffff`6ae0e130  00000000`00000000
0000ffff`6ae0e138  00000000`00000000
0000ffff`6ae0e140  00000000`00000000
0000ffff`6ae0e148  00000000`00000000
0000ffff`6ae0e150  00000000`00000000
0000ffff`6ae0e158  00000000`00000000
0000ffff`6ae0e160  00000000`00000000
0000ffff`6ae0e168  00000000`00000000
0000ffff`6ae0e170  00000000`00000000
0000ffff`6ae0e178  00000000`00000000
0000ffff`6ae0e180  00000000`00000000
0000ffff`6ae0e188  00000000`00000000
0000ffff`6ae0e190  00000000`00000000
0000ffff`6ae0e198  00000000`00000000
0000ffff`6ae0e1a0  00000000`00000000
0000ffff`6ae0e1a8  00000000`00000000
0000ffff`6ae0e1b0  0000ffff`6ae0e3d0
0000ffff`6ae0e1b8  00000000`00403244 App6!procF+0x5c
0000ffff`6ae0e1c0  00000000`00000000
0000ffff`6ae0e1c8  00000003`00000000
0000ffff`6ae0e1d0  00000000`ffffffff
0000ffff`6ae0e1d8  00000000`00000004
0000ffff`6ae0e1e0  00000000`ffffffff
0000ffff`6ae0e1e8  00000000`00000000
0000ffff`6ae0e1f0  00000000`00000000
0000ffff`6ae0e1f8  00000000`00000000
0000ffff`6ae0e200  00000000`00000000
0000ffff`6ae0e208  00000000`00000000
0000ffff`6ae0e210  00000000`00000000
0000ffff`6ae0e218  00000000`00000000
0000ffff`6ae0e220  00000000`00000000
0000ffff`6ae0e228  00000000`00000000
0000ffff`6ae0e230  00000000`00000000
0000ffff`6ae0e238  00000000`00000000
0000ffff`6ae0e240  00000000`00000000
0000ffff`6ae0e248  00000000`00000000
0000ffff`6ae0e250  00000000`00000000
0000ffff`6ae0e258  00000000`00000000
0000ffff`6ae0e260  00000000`00000000
0000ffff`6ae0e268  00000000`00000000
0000ffff`6ae0e270  00000000`00000000
0000ffff`6ae0e278  00000000`00000000
0000ffff`6ae0e280  00000000`00000000
0000ffff`6ae0e288  00000000`00000000
0000ffff`6ae0e290  00000000`00000000
0000ffff`6ae0e298  00000000`00000000
0000ffff`6ae0e2a0  00000000`00000000
0000ffff`6ae0e2a8  00000000`00000000
0000ffff`6ae0e2b0  00000000`00000000
0000ffff`6ae0e2b8  00000000`00000000
0000ffff`6ae0e2c0  00000000`00000000
0000ffff`6ae0e2c8  00000000`00000000
0000ffff`6ae0e2d0  00000000`00000000
0000ffff`6ae0e2d8  00000000`00000000
0000ffff`6ae0e2e0  00000000`00000000
0000ffff`6ae0e2e8  00000000`00000000
0000ffff`6ae0e2f0  00000000`00000000
0000ffff`6ae0e2f8  00000000`00000000
```

```
0000ffff`6ae0e300  00000000`00000000
0000ffff`6ae0e308  00000000`00000000
0000ffff`6ae0e310  00000000`00000000
0000ffff`6ae0e318  00000000`00000000
0000ffff`6ae0e320  00000000`00000000
0000ffff`6ae0e328  00000000`00000000
0000ffff`6ae0e330  00000000`00000000
0000ffff`6ae0e338  00000000`00000000
0000ffff`6ae0e340  00000000`00000000
0000ffff`6ae0e348  00000000`00000000
0000ffff`6ae0e350  00000000`00000000
0000ffff`6ae0e358  00000000`00000000
0000ffff`6ae0e360  00000000`00000000
0000ffff`6ae0e368  00000000`00000000
0000ffff`6ae0e370  00000000`00000000
0000ffff`6ae0e378  00000000`00000000
0000ffff`6ae0e380  00000000`00000000
0000ffff`6ae0e388  00000000`00000000
0000ffff`6ae0e390  00000000`00000000
0000ffff`6ae0e398  00000000`00000000
0000ffff`6ae0e3a0  00000000`00000000
0000ffff`6ae0e3a8  00000000`00000000
0000ffff`6ae0e3b0  00000000`00000000
0000ffff`6ae0e3b8  00000000`00000000
0000ffff`6ae0e3c0  00000000`00000000
0000ffff`6ae0e3c8  00000000`00000000
0000ffff`6ae0e3d0  0000ffff`6ae0e5f0
0000ffff`6ae0e3d8  00000000`00403244 App6!procF+0x5c
0000ffff`6ae0e3e0  00000000`00000000
0000ffff`6ae0e3e8  00000002`00000000
0000ffff`6ae0e3f0  00000000`ffffffff
0000ffff`6ae0e3f8  00000000`00000003
0000ffff`6ae0e400  00000000`ffffffff
0000ffff`6ae0e408  00000000`00000000
0000ffff`6ae0e410  00000000`00000000
0000ffff`6ae0e418  00000000`00000000
0000ffff`6ae0e420  00000000`00000000
0000ffff`6ae0e428  00000000`00000000
0000ffff`6ae0e430  00000000`00000000
0000ffff`6ae0e438  00000000`00000000
0000ffff`6ae0e440  00000000`00000000
0000ffff`6ae0e448  00000000`00000000
0000ffff`6ae0e450  00000000`00000000
0000ffff`6ae0e458  00000000`00000000
0000ffff`6ae0e460  00000000`00000000
0000ffff`6ae0e468  00000000`00000000
0000ffff`6ae0e470  00000000`00000000
0000ffff`6ae0e478  00000000`00000000
0000ffff`6ae0e480  00000000`00000000
0000ffff`6ae0e488  00000000`00000000
0000ffff`6ae0e490  00000000`00000000
0000ffff`6ae0e498  00000000`00000000
0000ffff`6ae0e4a0  00000000`00000000
0000ffff`6ae0e4a8  00000000`00000000
0000ffff`6ae0e4b0  00000000`00000000
0000ffff`6ae0e4b8  00000000`00000000
0000ffff`6ae0e4c0  00000000`00000000
0000ffff`6ae0e4c8  00000000`00000000
0000ffff`6ae0e4d0  00000000`00000000
0000ffff`6ae0e4d8  00000000`00000000
```

```
0000ffff`6ae0e4e0  00000000`00000000
0000ffff`6ae0e4e8  00000000`00000000
0000ffff`6ae0e4f0  00000000`00000000
0000ffff`6ae0e4f8  00000000`00000000
0000ffff`6ae0e500  00000000`00000000
0000ffff`6ae0e508  00000000`00000000
0000ffff`6ae0e510  00000000`00000000
0000ffff`6ae0e518  00000000`00000000
0000ffff`6ae0e520  00000000`00000000
0000ffff`6ae0e528  00000000`00000000
0000ffff`6ae0e530  00000000`00000000
0000ffff`6ae0e538  00000000`00000000
0000ffff`6ae0e540  00000000`00000000
0000ffff`6ae0e548  00000000`00000000
0000ffff`6ae0e550  00000000`00000000
0000ffff`6ae0e558  00000000`00000000
0000ffff`6ae0e560  00000000`00000000
0000ffff`6ae0e568  00000000`00000000
0000ffff`6ae0e570  00000000`00000000
0000ffff`6ae0e578  00000000`00000000
0000ffff`6ae0e580  00000000`00000000
0000ffff`6ae0e588  00000000`00000000
0000ffff`6ae0e590  00000000`00000000
0000ffff`6ae0e598  00000000`00000000
0000ffff`6ae0e5a0  00000000`00000000
0000ffff`6ae0e5a8  00000000`00000000
0000ffff`6ae0e5b0  00000000`00000000
0000ffff`6ae0e5b8  00000000`00000000
0000ffff`6ae0e5c0  00000000`00000000
0000ffff`6ae0e5c8  00000000`00000000
0000ffff`6ae0e5d0  00000000`00000000
0000ffff`6ae0e5d8  00000000`00000000
0000ffff`6ae0e5e0  00000000`00000000
0000ffff`6ae0e5e8  00000000`00000000
0000ffff`6ae0e5f0  0000ffff`6ae0e810
0000ffff`6ae0e5f8  00000000`00403260 App6!procE+0x10
0000ffff`6ae0e600  0000ffff`6ae0e670
0000ffff`6ae0e608  00000001`00000000
0000ffff`6ae0e610  00000000`ffffffff
0000ffff`6ae0e618  00000000`00000002
0000ffff`6ae0e620  00000000`ffffffff
0000ffff`6ae0e628  00000000`00000000
0000ffff`6ae0e630  00000000`00000000
0000ffff`6ae0e638  00000000`00000000
0000ffff`6ae0e640  00000000`00000000
0000ffff`6ae0e648  00000000`00000000
0000ffff`6ae0e650  00000000`00000000
0000ffff`6ae0e658  00000000`00000000
0000ffff`6ae0e660  00000000`00000000
0000ffff`6ae0e668  00000000`00000000
0000ffff`6ae0e670  00000000`00000000
0000ffff`6ae0e678  00000000`00000000
0000ffff`6ae0e680  00000000`00000000
0000ffff`6ae0e688  00000000`00000000
0000ffff`6ae0e690  00000000`00000000
0000ffff`6ae0e698  00000000`00000000
0000ffff`6ae0e6a0  00000000`00000000
0000ffff`6ae0e6a8  00000000`00000000
0000ffff`6ae0e6b0  00000000`00000000
0000ffff`6ae0e6b8  00000000`00000000
```

```
0000ffff`6ae0e6c0  00000000`00000000
0000ffff`6ae0e6c8  00000000`00000000
0000ffff`6ae0e6d0  00000000`00000000
0000ffff`6ae0e6d8  00000000`00000000
0000ffff`6ae0e6e0  00000000`00000000
0000ffff`6ae0e6e8  00000000`00000000
0000ffff`6ae0e6f0  00000000`00000000
0000ffff`6ae0e6f8  00000000`00000000
0000ffff`6ae0e700  00000000`00000000
0000ffff`6ae0e708  00000000`00000000
0000ffff`6ae0e710  00000000`00000000
0000ffff`6ae0e718  00000000`00000000
0000ffff`6ae0e720  00000000`00000000
0000ffff`6ae0e728  00000000`00000000
0000ffff`6ae0e730  00000000`00000000
0000ffff`6ae0e738  00000000`00000000
0000ffff`6ae0e740  00000000`00000000
0000ffff`6ae0e748  00000000`00000000
0000ffff`6ae0e750  00000000`00000000
0000ffff`6ae0e758  00000000`00000000
0000ffff`6ae0e760  00000000`00000000
0000ffff`6ae0e768  00000000`00000000
0000ffff`6ae0e770  00000000`00000000
0000ffff`6ae0e778  00000000`00000000
0000ffff`6ae0e780  00000000`00000000
0000ffff`6ae0e788  00000000`00000000
0000ffff`6ae0e790  00000000`00000000
0000ffff`6ae0e798  00000000`00000000
0000ffff`6ae0e7a0  00000000`00000000
0000ffff`6ae0e7a8  00000000`00000000
0000ffff`6ae0e7b0  00000000`00000000
0000ffff`6ae0e7b8  00000000`00000000
0000ffff`6ae0e7c0  00000000`00000000
0000ffff`6ae0e7c8  00000000`00000000
0000ffff`6ae0e7d0  00000000`00000000
0000ffff`6ae0e7d8  00000000`00000000
0000ffff`6ae0e7e0  00000000`00000000
0000ffff`6ae0e7e8  00000000`00000000
0000ffff`6ae0e7f0  00000000`00000000
0000ffff`6ae0e7f8  00000000`00000000
0000ffff`6ae0e800  00000000`00000000
0000ffff`6ae0e808  00000000`00000000
0000ffff`6ae0e810  0000ffff`6ae0e820
0000ffff`6ae0e818  00000000`0040327c App6!bar_one+0x14
0000ffff`6ae0e820  0000ffff`6ae0e830
0000ffff`6ae0e828  00000000`00403290 App6!foo_one+0xc
0000ffff`6ae0e830  0000ffff`6ae0e840
0000ffff`6ae0e838  00000000`004032a8 App6!thread_one+0x10
0000ffff`6ae0e840  0000ffff`6ae0e860
0000ffff`6ae0e848  00000000`00404cd4 App6!start_thread+0xb4
0000ffff`6ae0e850  0000ffff`6ae0f080
0000ffff`6ae0e858  00000000`00000000
0000ffff`6ae0e860  00000000`00000000
0000ffff`6ae0e868  00000000`00429c20 App6!thread_start+0x30
0000ffff`6ae0e870  0000ffff`6ae0f080
0000ffff`6ae0e878  00000000`004d7890 App6!_default_pthread_attr
0000ffff`6ae0e880  00000000`004d0000 App6!+0x18
0000ffff`6ae0e888  00000000`00000000
0000ffff`6ae0e890  0000ffff`6ae0f49c
0000ffff`6ae0e898  0000ffff`6ae0f080
```

```
0000ffff`6ae0e8a0  00000000`00000000
0000ffff`6ae0e8a8  00000000`00000000
0000ffff`6ae0e8b0  0000ffff`6ae0f080
0000ffff`6ae0e8b8  00000000`004d7890 App6!_default_pthread_attr
0000ffff`6ae0e8c0  00000000`004d0000 App6!+0x18
0000ffff`6ae0e8c8  00000000`00403298 App6!thread_one
0000ffff`6ae0e8d0  00000000`00000000
0000ffff`6ae0e8d8  0000ffff`6ae0f770
0000ffff`6ae0e8e0  00000000`30aa06f0
0000ffff`6ae0e8e8  00000000`004d7890 App6!_default_pthread_attr
0000ffff`6ae0e8f0  00000000`00010000
0000ffff`6ae0e8f8  00000000`00810000
0000ffff`6ae0e900  0000ffff`6ae0e860
0000ffff`6ae0e908  5afbedf4`15cdf4fb
0000ffff`6ae0e910  00000000`00000000
0000ffff`6ae0e918  5afb120b`7f6d503b
0000ffff`6ae0e920  00000000`00000000
0000ffff`6ae0e928  00000000`00000000
0000ffff`6ae0e930  00000000`00000000
0000ffff`6ae0e938  00000000`00000000
0000ffff`6ae0e940  00000000`00000000
0000ffff`6ae0e948  00000000`00000000
0000ffff`6ae0e950  00000000`00000000
0000ffff`6ae0e958  00000000`00000000
0000ffff`6ae0e960  00000000`00000000
0000ffff`6ae0e968  00000000`00000000
0000ffff`6ae0e970  00000000`00000000
0000ffff`6ae0e978  00000000`00000000
0000ffff`6ae0e980  00000000`00000000
0000ffff`6ae0e988  00000000`00000000
0000ffff`6ae0e990  00000000`00000000
0000ffff`6ae0e998  00000000`00000000
0000ffff`6ae0e9a0  00000000`00000000
0000ffff`6ae0e9a8  00000000`00000000
0000ffff`6ae0e9b0  00000000`00000000
0000ffff`6ae0e9b8  00000000`00000000
0000ffff`6ae0e9c0  00000000`00000000
0000ffff`6ae0e9c8  00000000`00000000
0000ffff`6ae0e9d0  00000000`00000000
0000ffff`6ae0e9d8  00000000`00000000
0000ffff`6ae0e9e0  00000000`00000000
0000ffff`6ae0e9e8  00000000`00000000
0000ffff`6ae0e9f0  00000000`00000000
0000ffff`6ae0e9f8  00000000`00000000
0000ffff`6ae0ea00  00000000`00000000
0000ffff`6ae0ea08  00000000`00000000
0000ffff`6ae0ea10  00000000`00000000
0000ffff`6ae0ea18  00000000`00000000
0000ffff`6ae0ea20  00000000`00000000
0000ffff`6ae0ea28  00000000`00000000
0000ffff`6ae0ea30  00000000`00000000
0000ffff`6ae0ea38  00000000`00000000
0000ffff`6ae0ea40  00000000`00000000
0000ffff`6ae0ea48  00000000`00000000
0000ffff`6ae0ea50  00000000`00000000
0000ffff`6ae0ea58  00000000`00000000
0000ffff`6ae0ea60  00000000`00000000
0000ffff`6ae0ea68  00000000`00000000
0000ffff`6ae0ea70  00000000`00000000
0000ffff`6ae0ea78  00000000`00000000
```

```
0000ffff`6ae0ea80  00000000`00000000
0000ffff`6ae0ea88  00000000`00000000
0000ffff`6ae0ea90  00000000`00000000
0000ffff`6ae0ea98  00000000`00000000
0000ffff`6ae0eaa0  00000000`00000000
0000ffff`6ae0eaa8  00000000`00000000
0000ffff`6ae0eab0  00000000`00000000
0000ffff`6ae0eab8  00000000`00000000
0000ffff`6ae0eac0  00000000`00000000
0000ffff`6ae0eac8  00000000`00000000
0000ffff`6ae0ead0  00000000`00000000
0000ffff`6ae0ead8  00000000`00000000
0000ffff`6ae0eae0  00000000`00000000
0000ffff`6ae0eae8  00000000`00000000
0000ffff`6ae0eaf0  00000000`00000000
0000ffff`6ae0eaf8  00000000`00000000
0000ffff`6ae0eb00  00000000`00000000
0000ffff`6ae0eb08  00000000`00000000
0000ffff`6ae0eb10  00000000`00000000
0000ffff`6ae0eb18  00000000`00000000
0000ffff`6ae0eb20  00000000`00000000
0000ffff`6ae0eb28  00000000`00000000
0000ffff`6ae0eb30  00000000`00000000
0000ffff`6ae0eb38  00000000`00000000
0000ffff`6ae0eb40  00000000`00000000
0000ffff`6ae0eb48  00000000`00000000
0000ffff`6ae0eb50  00000000`00000000
0000ffff`6ae0eb58  00000000`00000000
0000ffff`6ae0eb60  00000000`00000000
0000ffff`6ae0eb68  00000000`00000000
0000ffff`6ae0eb70  00000000`00000000
0000ffff`6ae0eb78  00000000`00000000
0000ffff`6ae0eb80  00000000`00000000
0000ffff`6ae0eb88  00000000`00000000
0000ffff`6ae0eb90  00000000`00000000
0000ffff`6ae0eb98  00000000`00000000
0000ffff`6ae0eba0  00000000`00000000
0000ffff`6ae0eba8  00000000`00000000
0000ffff`6ae0ebb0  00000000`00000000
0000ffff`6ae0ebb8  00000000`00000000
0000ffff`6ae0ebc0  00000000`00000000
0000ffff`6ae0ebc8  00000000`00000000
0000ffff`6ae0ebd0  00000000`00000000
0000ffff`6ae0ebd8  00000000`00000000
0000ffff`6ae0ebe0  00000000`00000000
0000ffff`6ae0ebe8  00000000`00000000
0000ffff`6ae0ebf0  00000000`00000000
0000ffff`6ae0ebf8  00000000`00000000
0000ffff`6ae0ec00  00000000`00000000
0000ffff`6ae0ec08  00000000`00000000
0000ffff`6ae0ec10  00000000`00000000
0000ffff`6ae0ec18  00000000`00000000
0000ffff`6ae0ec20  00000000`00000000
0000ffff`6ae0ec28  00000000`00000000
0000ffff`6ae0ec30  00000000`00000000
0000ffff`6ae0ec38  00000000`00000000
0000ffff`6ae0ec40  00000000`00000000
0000ffff`6ae0ec48  00000000`00000000
0000ffff`6ae0ec50  00000000`00000000
0000ffff`6ae0ec58  00000000`00000000
```

```
0000ffff`6ae0ec60  00000000`00000000
0000ffff`6ae0ec68  00000000`00000000
0000ffff`6ae0ec70  00000000`00000000
0000ffff`6ae0ec78  00000000`00000000
0000ffff`6ae0ec80  00000000`00000000
0000ffff`6ae0ec88  00000000`00000000
0000ffff`6ae0ec90  00000000`00000000
0000ffff`6ae0ec98  00000000`00000000
0000ffff`6ae0eca0  00000000`00000000
0000ffff`6ae0eca8  00000000`00000000
0000ffff`6ae0ecb0  00000000`00000000
0000ffff`6ae0ecb8  00000000`00000000
0000ffff`6ae0ecc0  00000000`00000000
0000ffff`6ae0ecc8  00000000`00000000
0000ffff`6ae0ecd0  00000000`00000000
0000ffff`6ae0ecd8  00000000`00000000
0000ffff`6ae0ece0  00000000`00000000
0000ffff`6ae0ece8  00000000`00000000
0000ffff`6ae0ecf0  00000000`00000000
0000ffff`6ae0ecf8  00000000`00000000
0000ffff`6ae0ed00  00000000`00000000
0000ffff`6ae0ed08  00000000`00000000
0000ffff`6ae0ed10  00000000`00000000
0000ffff`6ae0ed18  00000000`00000000
0000ffff`6ae0ed20  00000000`00000000
0000ffff`6ae0ed28  00000000`00000000
0000ffff`6ae0ed30  00000000`00000000
0000ffff`6ae0ed38  00000000`00000000
0000ffff`6ae0ed40  00000000`00000000
0000ffff`6ae0ed48  00000000`00000000
0000ffff`6ae0ed50  00000000`00000000
0000ffff`6ae0ed58  00000000`00000000
0000ffff`6ae0ed60  00000000`00000000
0000ffff`6ae0ed68  00000000`00000000
0000ffff`6ae0ed70  00000000`00000000
0000ffff`6ae0ed78  00000000`00000000
0000ffff`6ae0ed80  00000000`00000000
0000ffff`6ae0ed88  00000000`00000000
0000ffff`6ae0ed90  00000000`00000000
0000ffff`6ae0ed98  00000000`00000000
0000ffff`6ae0eda0  00000000`00000000
0000ffff`6ae0eda8  00000000`00000000
0000ffff`6ae0edb0  00000000`00000000
0000ffff`6ae0edb8  00000000`00000000
0000ffff`6ae0edc0  00000000`00000000
0000ffff`6ae0edc8  00000000`00000000
0000ffff`6ae0edd0  00000000`00000000
0000ffff`6ae0edd8  00000000`00000000
0000ffff`6ae0ede0  00000000`00000000
0000ffff`6ae0ede8  00000000`00000000
0000ffff`6ae0edf0  00000000`00000000
0000ffff`6ae0edf8  00000000`00000000
0000ffff`6ae0ee00  00000000`00000000
0000ffff`6ae0ee08  00000000`00000000
0000ffff`6ae0ee10  00000000`00000000
0000ffff`6ae0ee18  00000000`00000000
0000ffff`6ae0ee20  00000000`00000000
0000ffff`6ae0ee28  00000000`00000000
0000ffff`6ae0ee30  00000000`00000000
0000ffff`6ae0ee38  00000000`00000000
```

```
0000ffff`6ae0ee40  00000000`00000000
0000ffff`6ae0ee48  00000000`00000000
0000ffff`6ae0ee50  00000000`00000000
0000ffff`6ae0ee58  00000000`00000000
0000ffff`6ae0ee60  00000000`00000000
0000ffff`6ae0ee68  00000000`00000000
0000ffff`6ae0ee70  00000000`00000000
0000ffff`6ae0ee78  00000000`00000000
0000ffff`6ae0ee80  00000000`00000000
0000ffff`6ae0ee88  00000000`00000000
0000ffff`6ae0ee90  00000000`00000000
0000ffff`6ae0ee98  00000000`00000000
0000ffff`6ae0eea0  00000000`00000000
0000ffff`6ae0eea8  00000000`00000000
0000ffff`6ae0eeb0  00000000`00000000
0000ffff`6ae0eeb8  00000000`00000000
0000ffff`6ae0eec0  00000000`00000000
0000ffff`6ae0eec8  00000000`00000000
0000ffff`6ae0eed0  00000000`00000000
0000ffff`6ae0eed8  00000000`00000000
0000ffff`6ae0eee0  00000000`00000000
0000ffff`6ae0eee8  00000000`00000000
0000ffff`6ae0eef0  00000000`00000000
0000ffff`6ae0eef8  00000000`00000000
0000ffff`6ae0ef00  00000000`00000000
0000ffff`6ae0ef08  00000000`00000000
0000ffff`6ae0ef10  00000000`00000000
0000ffff`6ae0ef18  00000000`00000000
0000ffff`6ae0ef20  00000000`00000000
0000ffff`6ae0ef28  00000000`00000000
0000ffff`6ae0ef30  00000000`00000000
0000ffff`6ae0ef38  00000000`00000000
0000ffff`6ae0ef40  00000000`00000000
0000ffff`6ae0ef48  00000000`00000000
0000ffff`6ae0ef50  00000000`00000000
0000ffff`6ae0ef58  00000000`00000000
0000ffff`6ae0ef60  00000000`00000000
0000ffff`6ae0ef68  00000000`00000000
0000ffff`6ae0ef70  00000000`00000000
0000ffff`6ae0ef78  00000000`00000000
0000ffff`6ae0ef80  00000000`00000000
0000ffff`6ae0ef88  00000000`00000000
0000ffff`6ae0ef90  00000000`00000000
0000ffff`6ae0ef98  00000000`00000000
0000ffff`6ae0efa0  00000000`00000000
0000ffff`6ae0efa8  00000000`00000000
0000ffff`6ae0efb0  00000000`00000000
0000ffff`6ae0efb8  00000000`00000000
0000ffff`6ae0efc0  00000000`00000000
0000ffff`6ae0efc8  00000000`00000000
0000ffff`6ae0efd0  00000000`00000000
0000ffff`6ae0efd8  00000000`00000000
0000ffff`6ae0efe0  00000000`00000000
0000ffff`6ae0efe8  00000000`00000000
0000ffff`6ae0eff0  00000000`00000000
0000ffff`6ae0eff8  00000000`00000000
0000ffff`6ae0f000  00000000`00000000
0000ffff`6ae0f008  00000000`00000000
0000ffff`6ae0f010  00000000`00000000
0000ffff`6ae0f018  00000000`00000000
```

```
0000ffff`6ae0f020  00000000`00000000
0000ffff`6ae0f028  00000000`00000000
0000ffff`6ae0f030  00000000`00000000
0000ffff`6ae0f038  00000000`00000000
0000ffff`6ae0f040  00000000`00000000
0000ffff`6ae0f048  00000000`00000000
0000ffff`6ae0f050  00000000`00000000
0000ffff`6ae0f058  00000000`00000000
0000ffff`6ae0f060  00000000`00000000
0000ffff`6ae0f068  00000000`00000000
0000ffff`6ae0f070  00000000`00000000
0000ffff`6ae0f078  00000000`00000000
0000ffff`6ae0f080  00000000`00000001
0000ffff`6ae0f088  00000000`00000000
0000ffff`6ae0f090  00000000`00000000
0000ffff`6ae0f098  00000000`00000000
0000ffff`6ae0f0a0  00000000`00000000
0000ffff`6ae0f0a8  00000000`00000000
0000ffff`6ae0f0b0  00000000`00000000
0000ffff`6ae0f0b8  00000000`00000000
0000ffff`6ae0f0c0  00000000`00000000
0000ffff`6ae0f0c8  00000000`00000000
0000ffff`6ae0f0d0  00000000`00000000
0000ffff`6ae0f0d8  00000000`00000000
0000ffff`6ae0f0e0  00000000`00000000
0000ffff`6ae0f0e8  00000000`00000000
0000ffff`6ae0f0f0  00000000`00000000
0000ffff`6ae0f0f8  00000000`00000000
0000ffff`6ae0f100  00000000`00000000
0000ffff`6ae0f108  00000000`00000000
0000ffff`6ae0f110  00000000`00000000
0000ffff`6ae0f118  00000000`00000000
0000ffff`6ae0f120  00000000`00000000
0000ffff`6ae0f128  00000000`00000000
0000ffff`6ae0f130  00000000`00000000
0000ffff`6ae0f138  00000000`00000000
0000ffff`6ae0f140  00000000`004d0050 App6!stack_used
0000ffff`6ae0f148  0000ffff`6a5ff140
0000ffff`6ae0f150  00004bc1`00004bc2
0000ffff`6ae0f158  0000ffff`6ae0f160
0000ffff`6ae0f160  0000ffff`6ae0f160
0000ffff`6ae0f168  ffffffff`fffffe0
0000ffff`6ae0f170  00000000`00000000
0000ffff`6ae0f178  00000000`00000000
0000ffff`6ae0f180  0000ffff`6ae0e8b0
0000ffff`6ae0f188  00000000`00000000
0000ffff`6ae0f190  00000000`00000000
0000ffff`6ae0f198  00000000`00000000
0000ffff`6ae0f1a0  00000000`00000000
0000ffff`6ae0f1a8  00000000`00000000
0000ffff`6ae0f1b0  00000000`00000000
0000ffff`6ae0f1b8  00000000`00000000
0000ffff`6ae0f1c0  00000000`00000000
0000ffff`6ae0f1c8  00000000`00000000
0000ffff`6ae0f1d0  00000000`00000000
0000ffff`6ae0f1d8  00000000`00000000
0000ffff`6ae0f1e0  00000000`00000000
0000ffff`6ae0f1e8  00000000`00000000
0000ffff`6ae0f1f0  00000000`00000000
0000ffff`6ae0f1f8  00000000`00000000
```

```
0000ffff`6ae0f200  00000000`00000000
0000ffff`6ae0f208  00000000`00000000
0000ffff`6ae0f210  00000000`00000000
0000ffff`6ae0f218  00000000`00000000
0000ffff`6ae0f220  00000000`00000000
0000ffff`6ae0f228  00000000`00000000
0000ffff`6ae0f230  00000000`00000000
0000ffff`6ae0f238  00000000`00000000
0000ffff`6ae0f240  00000000`00000000
0000ffff`6ae0f248  00000000`00000000
0000ffff`6ae0f250  00000000`00000000
0000ffff`6ae0f258  00000000`00000000
0000ffff`6ae0f260  00000000`00000000
0000ffff`6ae0f268  00000000`00000000
0000ffff`6ae0f270  00000000`00000000
0000ffff`6ae0f278  00000000`00000000
0000ffff`6ae0f280  00000000`00000000
0000ffff`6ae0f288  00000000`00000000
0000ffff`6ae0f290  00000000`00000000
0000ffff`6ae0f298  00000000`00000000
0000ffff`6ae0f2a0  00000000`00000000
0000ffff`6ae0f2a8  00000000`00000000
0000ffff`6ae0f2b0  00000000`00000000
0000ffff`6ae0f2b8  00000000`00000000
0000ffff`6ae0f2c0  00000000`00000000
0000ffff`6ae0f2c8  00000000`00000000
0000ffff`6ae0f2d0  00000000`00000000
0000ffff`6ae0f2d8  00000000`00000000
0000ffff`6ae0f2e0  00000000`00000000
0000ffff`6ae0f2e8  00000000`00000000
0000ffff`6ae0f2f0  00000000`00000000
0000ffff`6ae0f2f8  00000000`00000000
0000ffff`6ae0f300  00000000`00000000
0000ffff`6ae0f308  00000000`00000000
0000ffff`6ae0f310  00000000`00000000
0000ffff`6ae0f318  00000000`00000000
0000ffff`6ae0f320  00000000`00000000
0000ffff`6ae0f328  00000000`00000000
0000ffff`6ae0f330  00000000`00000000
0000ffff`6ae0f338  00000000`00000000
0000ffff`6ae0f340  00000000`00000000
0000ffff`6ae0f348  00000000`00000000
0000ffff`6ae0f350  00000000`00000000
0000ffff`6ae0f358  00000000`00000000
0000ffff`6ae0f360  00000000`00000000
0000ffff`6ae0f368  00000000`00000000
0000ffff`6ae0f370  00000000`00000000
0000ffff`6ae0f378  00000000`00000000
0000ffff`6ae0f380  00000000`00000000
0000ffff`6ae0f388  00000000`00000000
0000ffff`6ae0f390  0000ffff`6ae0f190
0000ffff`6ae0f398  00000000`00000000
0000ffff`6ae0f3a0  00000000`00000000
0000ffff`6ae0f3a8  00000000`00000000
0000ffff`6ae0f3b0  00000000`00000000
0000ffff`6ae0f3b8  00000000`00000000
0000ffff`6ae0f3c0  00000000`00000000
0000ffff`6ae0f3c8  00000000`00000000
0000ffff`6ae0f3d0  00000000`00000000
0000ffff`6ae0f3d8  00000000`00000000
```

```
0000ffff`6ae0f3e0  00000000`00000000
0000ffff`6ae0f3e8  00000000`00000000
0000ffff`6ae0f3f0  00000000`00000000
0000ffff`6ae0f3f8  00000000`00000000
0000ffff`6ae0f400  00000000`00000000
0000ffff`6ae0f408  00000000`00000000
0000ffff`6ae0f410  00000000`00000000
0000ffff`6ae0f418  00000000`00000000
0000ffff`6ae0f420  00000000`00000000
0000ffff`6ae0f428  00000000`00000000
0000ffff`6ae0f430  00000000`00000000
0000ffff`6ae0f438  00000000`00000000
0000ffff`6ae0f440  00000000`00000000
0000ffff`6ae0f448  00000000`00000000
0000ffff`6ae0f450  00000000`00000000
0000ffff`6ae0f458  00000000`00000000
0000ffff`6ae0f460  00000000`00000000
0000ffff`6ae0f468  00000000`00000000
0000ffff`6ae0f470  00000000`00000000
0000ffff`6ae0f478  00000000`00000000
0000ffff`6ae0f480  00000000`00000000
0000ffff`6ae0f488  00000000`00000000
0000ffff`6ae0f490  00000000`00000000
0000ffff`6ae0f498  00000000`00000000
0000ffff`6ae0f4a0  00000000`00000000
0000ffff`6ae0f4a8  00000000`00000000
0000ffff`6ae0f4b0  00000000`00000000
0000ffff`6ae0f4b8  00000000`00403298 App6!thread_one
0000ffff`6ae0f4c0  00000000`00000000
0000ffff`6ae0f4c8  00000000`00000000
0000ffff`6ae0f4d0  00000000`00000000
0000ffff`6ae0f4d8  00000000`00000000
0000ffff`6ae0f4e0  00000000`00000000
0000ffff`6ae0f4e8  00000000`00000000
0000ffff`6ae0f4f0  00000000`00000000
0000ffff`6ae0f4f8  00000000`00000000
0000ffff`6ae0f500  00000000`00000000
0000ffff`6ae0f508  00000000`00000000
0000ffff`6ae0f510  0000ffff`6a600000
0000ffff`6ae0f518  00000000`00810000
0000ffff`6ae0f520  00000000`00010000
0000ffff`6ae0f528  00000000`00010000
0000ffff`6ae0f530  00000000`00000000
0000ffff`6ae0f538  00000000`00000000
0000ffff`6ae0f540  00000000`00000000
0000ffff`6ae0f548  00000000`00000000
0000ffff`6ae0f550  00000000`00000000
0000ffff`6ae0f558  00000000`00000000
0000ffff`6ae0f560  00000000`00000000
0000ffff`6ae0f568  00000000`00000000
0000ffff`6ae0f570  00000000`00000000
0000ffff`6ae0f578  00000000`00000000
0000ffff`6ae0f580  00000000`00000000
0000ffff`6ae0f588  00000000`00000000
0000ffff`6ae0f590  00000000`00000000
0000ffff`6ae0f598  00000000`00000000
0000ffff`6ae0f5a0  00000000`00000000
0000ffff`6ae0f5a8  00000000`00000000
0000ffff`6ae0f5b0  00000000`00000000
0000ffff`6ae0f5b8  00000000`00000000
```

```
0000ffff`6ae0f5c0  00000000`00000000
0000ffff`6ae0f5c8  00000000`00000000
0000ffff`6ae0f5d0  00000000`00000000
0000ffff`6ae0f5d8  00000000`00000000
0000ffff`6ae0f5e0  00000000`00000000
0000ffff`6ae0f5e8  00000000`00000000
0000ffff`6ae0f5f0  00000000`00000000
0000ffff`6ae0f5f8  00000000`00000000
0000ffff`6ae0f600  00000000`00000000
0000ffff`6ae0f608  00000000`00000000
0000ffff`6ae0f610  00000000`00000000
0000ffff`6ae0f618  00000000`00000000
0000ffff`6ae0f620  00000000`00000000
0000ffff`6ae0f628  00000000`00000000
0000ffff`6ae0f630  00000000`00000000
0000ffff`6ae0f638  00000000`00000000
0000ffff`6ae0f640  00000000`00000000
0000ffff`6ae0f648  00000000`00000000
0000ffff`6ae0f650  00000000`00000000
0000ffff`6ae0f658  00000000`00000000
0000ffff`6ae0f660  00000000`00000000
0000ffff`6ae0f668  00000000`00000000
0000ffff`6ae0f670  00000000`00000000
0000ffff`6ae0f678  00000000`00000000
0000ffff`6ae0f680  00000000`00000000
0000ffff`6ae0f688  00000000`00000000
0000ffff`6ae0f690  00000000`00000000
0000ffff`6ae0f698  00000000`00000000
0000ffff`6ae0f6a0  00000000`00000000
0000ffff`6ae0f6a8  00000000`00000000
0000ffff`6ae0f6b0  00000000`00000000
0000ffff`6ae0f6b8  00000000`00000000
0000ffff`6ae0f6c0  00000000`00000000
0000ffff`6ae0f6c8  00000000`00000000
0000ffff`6ae0f6d0  00000000`00000000
0000ffff`6ae0f6d8  00000000`00000000
0000ffff`6ae0f6e0  00000000`00000000
0000ffff`6ae0f6e8  00000000`00000000
0000ffff`6ae0f6f0  00000000`00000000
0000ffff`6ae0f6f8  00000000`00000000
0000ffff`6ae0f700  00000000`00000000
0000ffff`6ae0f708  00000000`00000000
0000ffff`6ae0f710  00000000`00000000
0000ffff`6ae0f718  00000000`00000000
0000ffff`6ae0f720  00000000`00000000
0000ffff`6ae0f728  00000000`00000000
0000ffff`6ae0f730  00000000`00000000
0000ffff`6ae0f738  00000000`00000000
0000ffff`6ae0f740  00000000`00000000
0000ffff`6ae0f748  00000000`00000000
0000ffff`6ae0f750  00000000`00000000
0000ffff`6ae0f758  00000000`00000000
0000ffff`6ae0f760  00000000`00000000
0000ffff`6ae0f768  00000000`00000000
0000ffff`6ae0f770  00000000`30aa1d80
0000ffff`6ae0f778  00000000`00000000
0000ffff`6ae0f780  0000ffff`6ae0f538
0000ffff`6ae0f788  00000000`004d13c0 App6!nl_global_locale
0000ffff`6ae0f790  00000000`004d13c0 App6!nl_global_locale
0000ffff`6ae0f798  00000000`004d13e0 App6!nl_global_locale+0x20
```

```
0000ffff`6ae0f7a0  00000000`004d13c8 App6!nl_global_locale+0x8
0000ffff`6ae0f7a8  00000000`00000000
0000ffff`6ae0f7b0  00000000`0048d280 App6!nl_C_LC_CTYPE_class+0x100
0000ffff`6ae0f7b8  00000000`0048c980 App6!nl_C_LC_CTYPE_toupper+0x200
0000ffff`6ae0f7c0  00000000`0048c380 App6!nl_C_LC_CTYPE_tolower+0x200
0000ffff`6ae0f7c8  00000000`00000000
0000ffff`6ae0f7d0  00000000`00000000
0000ffff`6ae0f7d8  00000000`00000000
0000ffff`6ae0f7e0  00000000`00000000
0000ffff`6ae0f7e8  00000000`00000000
0000ffff`6ae0f7f0  00000000`00000000
0000ffff`6ae0f7f8  00000000`00000000
0000ffff`6ae0f800  00000000`00000000
0000ffff`6ae0f808  00000000`00000000
0000ffff`6ae0f810  00000000`00000000
0000ffff`6ae0f818  00000000`00000000
0000ffff`6ae0f820  00000000`00000000
0000ffff`6ae0f828  00000000`00000000
0000ffff`6ae0f830  00000000`00000000
0000ffff`6ae0f838  00000000`00000000
0000ffff`6ae0f840  00000000`00000000
0000ffff`6ae0f848  00000000`00000000
0000ffff`6ae0f850  00000000`00000000
0000ffff`6ae0f858  00000000`00000000
0000ffff`6ae0f860  00000000`00000000
0000ffff`6ae0f868  00000000`00000000
0000ffff`6ae0f870  00000000`00000000
0000ffff`6ae0f878  00000000`00000000
0000ffff`6ae0f880  00000000`00000000
0000ffff`6ae0f888  00000000`00000000
0000ffff`6ae0f890  00000000`00000000
0000ffff`6ae0f898  00000000`00000000
0000ffff`6ae0f8a0  00000000`00000000
0000ffff`6ae0f8a8  00000000`00000000
0000ffff`6ae0f8b0  00000000`00000000
0000ffff`6ae0f8b8  00000000`00000000
0000ffff`6ae0f8c0  00000000`00000000
0000ffff`6ae0f8c8  00000000`00000000
0000ffff`6ae0f8d0  00000000`00000000
0000ffff`6ae0f8d8  00000000`00000000
0000ffff`6ae0f8e0  00000000`00000000
0000ffff`6ae0f8e8  00000000`00000000
0000ffff`6ae0f8f0  00000000`00000000
0000ffff`6ae0f8f8  00000000`00000000
0000ffff`6ae0f900  00000000`00000000
0000ffff`6ae0f908  00000000`00000000
0000ffff`6ae0f910  00000000`00000000
0000ffff`6ae0f918  00000000`00000000
0000ffff`6ae0f920  00000000`00000000
0000ffff`6ae0f928  00000000`00000000
0000ffff`6ae0f930  00000000`00000000
0000ffff`6ae0f938  00000000`00000000
0000ffff`6ae0f940  00000000`00000000
0000ffff`6ae0f948  00000000`00000000
0000ffff`6ae0f950  00000000`00000000
0000ffff`6ae0f958  00000000`00000000
0000ffff`6ae0f960  00000000`00000000
0000ffff`6ae0f968  00000000`00000000
0000ffff`6ae0f970  00000000`00000000
0000ffff`6ae0f978  00000000`00000000
```

```
0000ffff`6ae0f980  00000000`00000000
0000ffff`6ae0f988  00000000`00000000
0000ffff`6ae0f990  00000000`00000000
0000ffff`6ae0f998  00000000`00000000
0000ffff`6ae0f9a0  00000000`00000000
0000ffff`6ae0f9a8  00000000`00000000
0000ffff`6ae0f9b0  00000000`00000000
0000ffff`6ae0f9b8  00000000`00000000
0000ffff`6ae0f9c0  00000000`00000000
0000ffff`6ae0f9c8  00000000`00000000
0000ffff`6ae0f9d0  00000000`00000000
0000ffff`6ae0f9d8  00000000`00000000
0000ffff`6ae0f9e0  00000000`00000000
0000ffff`6ae0f9e8  00000000`00000000
0000ffff`6ae0f9f0  00000000`00000000
0000ffff`6ae0f9f8  00000000`00000000
0000ffff`6ae0fa00  00000000`00000000
0000ffff`6ae0fa08  00000000`00000000
0000ffff`6ae0fa10  00000000`00000000
0000ffff`6ae0fa18  00000000`00000000
0000ffff`6ae0fa20  00000000`00000000
0000ffff`6ae0fa28  00000000`00000000
0000ffff`6ae0fa30  00000000`00000000
0000ffff`6ae0fa38  00000000`00000000
0000ffff`6ae0fa40  00000000`00000000
0000ffff`6ae0fa48  00000000`00000000
0000ffff`6ae0fa50  00000000`00000000
0000ffff`6ae0fa58  00000000`00000000
0000ffff`6ae0fa60  00000000`00000000
0000ffff`6ae0fa68  00000000`00000000
0000ffff`6ae0fa70  00000000`00000000
0000ffff`6ae0fa78  00000000`00000000
0000ffff`6ae0fa80  00000000`00000000
0000ffff`6ae0fa88  00000000`00000000
0000ffff`6ae0fa90  00000000`00000000
0000ffff`6ae0fa98  00000000`00000000
0000ffff`6ae0faa0  00000000`00000000
0000ffff`6ae0faa8  00000000`00000000
0000ffff`6ae0fab0  00000000`00000000
0000ffff`6ae0fab8  00000000`00000000
0000ffff`6ae0fac0  00000000`00000000
0000ffff`6ae0fac8  00000000`00000000
0000ffff`6ae0fad0  00000000`00000000
0000ffff`6ae0fad8  00000000`00000000
0000ffff`6ae0fae0  00000000`00000000
0000ffff`6ae0fae8  00000000`00000000
0000ffff`6ae0faf0  00000000`00000000
0000ffff`6ae0faf8  00000000`00000000
0000ffff`6ae0fb00  00000000`00000000
0000ffff`6ae0fb08  00000000`00000000
0000ffff`6ae0fb10  00000000`00000000
0000ffff`6ae0fb18  00000000`00000000
0000ffff`6ae0fb20  00000000`00000000
0000ffff`6ae0fb28  00000000`00000000
0000ffff`6ae0fb30  00000000`00000000
0000ffff`6ae0fb38  00000000`00000000
0000ffff`6ae0fb40  00000000`00000000
0000ffff`6ae0fb48  00000000`00000000
0000ffff`6ae0fb50  00000000`00000000
0000ffff`6ae0fb58  00000000`00000000
```

```
0000ffff`6ae0fb60  00000000`00000000
0000ffff`6ae0fb68  00000000`00000000
0000ffff`6ae0fb70  00000000`00000000
0000ffff`6ae0fb78  00000000`00000000
0000ffff`6ae0fb80  00000000`00000000
0000ffff`6ae0fb88  00000000`00000000
0000ffff`6ae0fb90  00000000`00000000
0000ffff`6ae0fb98  00000000`00000000
0000ffff`6ae0fba0  00000000`00000000
0000ffff`6ae0fba8  00000000`00000000
0000ffff`6ae0fbb0  00000000`00000000
0000ffff`6ae0fbb8  00000000`00000000
0000ffff`6ae0fbc0  00000000`00000000
0000ffff`6ae0fbc8  00000000`00000000
0000ffff`6ae0fbd0  00000000`00000000
0000ffff`6ae0fbd8  00000000`00000000
0000ffff`6ae0fbe0  00000000`00000000
0000ffff`6ae0fbe8  00000000`00000000
0000ffff`6ae0fbf0  00000000`00000000
0000ffff`6ae0fbf8  00000000`00000000
0000ffff`6ae0fc00  00000000`00000000
0000ffff`6ae0fc08  00000000`00000000
0000ffff`6ae0fc10  00000000`00000000
0000ffff`6ae0fc18  00000000`00000000
0000ffff`6ae0fc20  00000000`00000000
0000ffff`6ae0fc28  00000000`00000000
0000ffff`6ae0fc30  00000000`00000000
0000ffff`6ae0fc38  00000000`00000000
0000ffff`6ae0fc40  00000000`00000000
0000ffff`6ae0fc48  00000000`00000000
0000ffff`6ae0fc50  00000000`00000000
0000ffff`6ae0fc58  00000000`00000000
0000ffff`6ae0fc60  00000000`00000000
0000ffff`6ae0fc68  00000000`00000000
0000ffff`6ae0fc70  00000000`00000000
0000ffff`6ae0fc78  00000000`00000000
0000ffff`6ae0fc80  00000000`00000000
0000ffff`6ae0fc88  00000000`00000000
0000ffff`6ae0fc90  00000000`00000000
0000ffff`6ae0fc98  00000000`00000000
0000ffff`6ae0fca0  00000000`00000000
0000ffff`6ae0fca8  00000000`00000000
0000ffff`6ae0fcb0  00000000`00000000
0000ffff`6ae0fcb8  00000000`00000000
0000ffff`6ae0fcc0  00000000`00000000
0000ffff`6ae0fcc8  00000000`00000000
0000ffff`6ae0fcd0  00000000`00000000
0000ffff`6ae0fcd8  00000000`00000000
0000ffff`6ae0fce0  00000000`00000000
0000ffff`6ae0fce8  00000000`00000000
0000ffff`6ae0fcf0  00000000`00000000
0000ffff`6ae0fcf8  00000000`00000000
0000ffff`6ae0fd00  00000000`00000000
0000ffff`6ae0fd08  00000000`00000000
0000ffff`6ae0fd10  00000000`00000000
0000ffff`6ae0fd18  00000000`00000000
0000ffff`6ae0fd20  00000000`00000000
0000ffff`6ae0fd28  00000000`00000000
0000ffff`6ae0fd30  00000000`00000000
0000ffff`6ae0fd38  00000000`00000000
```

```
0000ffff`6ae0fd40  00000000`00000000
0000ffff`6ae0fd48  00000000`00000000
0000ffff`6ae0fd50  00000000`00000000
0000ffff`6ae0fd58  00000000`00000000
0000ffff`6ae0fd60  00000000`00000000
0000ffff`6ae0fd68  00000000`00000000
0000ffff`6ae0fd70  00000000`00000000
0000ffff`6ae0fd78  00000000`00000000
0000ffff`6ae0fd80  00000000`00000000
0000ffff`6ae0fd88  00000000`00000000
0000ffff`6ae0fd90  00000000`00000000
0000ffff`6ae0fd98  00000000`00000000
0000ffff`6ae0fda0  00000000`00000000
0000ffff`6ae0fda8  00000000`00000000
0000ffff`6ae0fdb0  00000000`00000000
0000ffff`6ae0fdb8  00000000`00000000
0000ffff`6ae0fdc0  00000000`00000000
0000ffff`6ae0fdc8  00000000`00000000
0000ffff`6ae0fdd0  00000000`00000000
0000ffff`6ae0fdd8  00000000`00000000
0000ffff`6ae0fde0  00000000`00000000
0000ffff`6ae0fde8  00000000`00000000
0000ffff`6ae0fdf0  00000000`00000000
0000ffff`6ae0fdf8  00000000`00000000
0000ffff`6ae0fe00  00000000`00000000
0000ffff`6ae0fe08  00000000`00000000
0000ffff`6ae0fe10  00000000`00000000
0000ffff`6ae0fe18  00000000`00000000
0000ffff`6ae0fe20  00000000`00000000
0000ffff`6ae0fe28  00000000`00000000
0000ffff`6ae0fe30  00000000`00000000
0000ffff`6ae0fe38  00000000`00000000
0000ffff`6ae0fe40  00000000`00000000
0000ffff`6ae0fe48  00000000`00000000
0000ffff`6ae0fe50  00000000`00000000
0000ffff`6ae0fe58  00000000`00000000
0000ffff`6ae0fe60  00000000`00000000
0000ffff`6ae0fe68  00000000`00000000
0000ffff`6ae0fe70  00000000`00000000
0000ffff`6ae0fe78  00000000`00000000
0000ffff`6ae0fe80  00000000`00000000
0000ffff`6ae0fe88  00000000`00000000
0000ffff`6ae0fe90  00000000`00000000
0000ffff`6ae0fe98  00000000`00000000
0000ffff`6ae0fea0  00000000`00000000
0000ffff`6ae0fea8  00000000`00000000
0000ffff`6ae0feb0  00000000`00000000
0000ffff`6ae0feb8  00000000`00000000
0000ffff`6ae0fec0  00000000`00000000
0000ffff`6ae0fec8  00000000`00000000
0000ffff`6ae0fed0  00000000`00000000
0000ffff`6ae0fed8  00000000`00000000
0000ffff`6ae0fee0  00000000`00000000
0000ffff`6ae0fee8  00000000`00000000
0000ffff`6ae0fef0  00000000`00000000
0000ffff`6ae0fef8  00000000`00000000
0000ffff`6ae0ff00  00000000`00000000
0000ffff`6ae0ff08  00000000`00000000
0000ffff`6ae0ff10  00000000`00000000
0000ffff`6ae0ff18  00000000`00000000
```

```
0000ffff`6ae0ff20  00000000`00000000
0000ffff`6ae0ff28  00000000`00000000
0000ffff`6ae0ff30  00000000`00000000
0000ffff`6ae0ff38  00000000`00000000
0000ffff`6ae0ff40  00000000`00000000
0000ffff`6ae0ff48  00000000`00000000
0000ffff`6ae0ff50  00000000`00000000
0000ffff`6ae0ff58  00000000`00000000
0000ffff`6ae0ff60  00000000`00000000
0000ffff`6ae0ff68  00000000`00000000
0000ffff`6ae0ff70  00000000`00000000
0000ffff`6ae0ff78  00000000`00000000
0000ffff`6ae0ff80  00000000`00000000
0000ffff`6ae0ff88  00000000`00000000
0000ffff`6ae0ff90  00000000`00000000
0000ffff`6ae0ff98  00000000`00000000
0000ffff`6ae0ffa0  00000000`00000000
0000ffff`6ae0ffa8  00000000`00000000
0000ffff`6ae0ffb0  00000000`00000000
0000ffff`6ae0ffb8  00000000`00000000
0000ffff`6ae0ffc0  00000000`00000000
0000ffff`6ae0ffc8  00000000`00000000
0000ffff`6ae0ffd0  00000000`00000000
0000ffff`6ae0ffd8  00000000`00000000
0000ffff`6ae0ffe0  00000000`00000000
0000ffff`6ae0ffe8  00000000`00000000
0000ffff`6ae0fff0  00000000`00000000
0000ffff`6ae0fff8  00000000`00000000
0000ffff`6ae10000  00000002`0eca5306
```

10. See that the reconstruction of the stack trace is possible because of the standard function prologue and epilogue:

```
[...]
0000ffff`6ae0e1b0  0000ffff`6ae0e3d0
0000ffff`6ae0e1b8  00000000`00403244 App6!procF+0x5c
0000ffff`6ae0e3d0  0000ffff`6ae0e5f0
0000ffff`6ae0e3d8  00000000`00403244 App6!procF+0x5c
0000ffff`6ae0e5f0  0000ffff`6ae0e810
0000ffff`6ae0e5f8  00000000`00403260 App6!procE+0x10
0000ffff`6ae0e810  0000ffff`6ae0e820
0000ffff`6ae0e818  00000000`0040327c App6!bar_one+0x14
0000ffff`6ae0e820  0000ffff`6ae0e830
0000ffff`6ae0e828  00000000`00403290 App6!foo_one+0xc
0000ffff`6ae0e830  0000ffff`6ae0e840
0000ffff`6ae0e838  00000000`004032a8 App6!thread_one+0x10
0000ffff`6ae0e840  0000ffff`6ae0e860
0000ffff`6ae0e848  00000000`00404cd4 App6!start_thread+0xb4
0000ffff`6ae0e850  0000ffff`6ae0f080
0000ffff`6ae0e858  00000000`00000000
0000ffff`6ae0e860  00000000`00000000
0000ffff`6ae0e868  00000000`00429c20 App6!thread_start+0x30

0:000> uf procF
App6!procF:
00000000`004031e8 d10843ff sub         sp,sp,#0x210
00000000`004031ec a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`004031f0 910003fd mov         fp,sp
00000000`004031f4 910073a1 add         x1,fp,#0x1C
00000000`004031f8 b9000020 str         w0,[x1]
```

```
00000000`004031fc 910083a0 add         x0,fp,#0x20
00000000`00403200 d2804002 mov         x2,#0x200
00000000`00403204 52800001 mov         w1,#0
00000000`00403208 97fff422 bl          App6!+0x20 (00000000`00400290)
00000000`0040320c 910083a0 add         x0,fp,#0x20
00000000`00403210 12800001 mov         w1,#-1
00000000`00403214 b9000001 str         w1,[x0]
00000000`00403218 910073a0 add         x0,fp,#0x1C
00000000`0040321c b9400000 ldr         w0,[x0]
00000000`00403220 11000401 add         w1,w0,#1
00000000`00403224 910083a0 add         x0,fp,#0x20
00000000`00403228 b9000801 str         w1,[x0,#8]
00000000`0040322c 910083a0 add         x0,fp,#0x20
00000000`00403230 12800001 mov         w1,#-1
00000000`00403234 b9001001 str         w1,[x0,#0x10]
00000000`00403238 910083a0 add         x0,fp,#0x20
00000000`0040323c b9400800 ldr         w0,[x0,#8]
00000000`00403240 97ffffea bl          App6!procF (00000000`004031e8)
00000000`00403244 a8c17bfd ldp         fp,lr,[sp],#0x10
00000000`00403248 910843ff add         sp,sp,#0x210
00000000`0040324c d65f03c0 ret
```

11. To see the bottom of the stack trace, we can increase the default number of frames:

```
0:000> .kframes 0xffff
Default stack trace depth is 0n65535 frames
```

```
0:000> k
 # Child-SP          RetAddr               Call Site
00 0000ffff`6a610000 00000000`00403244     App6!procF+0x4
01 0000ffff`6a610210 00000000`00403244     App6!procF+0x5c
02 0000ffff`6a610430 00000000`00403244     App6!procF+0x5c
03 0000ffff`6a610650 00000000`00403244     App6!procF+0x5c
04 0000ffff`6a610870 00000000`00403244     App6!procF+0x5c
05 0000ffff`6a610a90 00000000`00403244     App6!procF+0x5c
06 0000ffff`6a610cb0 00000000`00403244     App6!procF+0x5c
07 0000ffff`6a610ed0 00000000`00403244     App6!procF+0x5c
08 0000ffff`6a6110f0 00000000`00403244     App6!procF+0x5c
09 0000ffff`6a611310 00000000`00403244     App6!procF+0x5c
0a 0000ffff`6a611530 00000000`00403244     App6!procF+0x5c
[...]
3c2e 0000ffff`6ae0e1b0 00000000`00403244     App6!procF+0x5c
3c2f 0000ffff`6ae0e3d0 00000000`00403244     App6!procF+0x5c
3c30 0000ffff`6ae0e5f0 00000000`00403260     App6!procF+0x5c
3c31 0000ffff`6ae0e810 00000000`0040327c     App6!procE+0x10
3c32 0000ffff`6ae0e820 00000000`00403290     App6!bar_one+0x14
3c33 0000ffff`6ae0e830 00000000`004032a8     App6!foo_one+0xc
3c34 0000ffff`6ae0e840 00000000`00404cd4     App6!thread_one+0x10
3c35 0000ffff`6ae0e860 00000000`00429c20     App6!start_thread+0xb4
3c36 0000ffff`6ae0e990 ffffffff`ffffffff     App6!thread_start+0x30
3c37 0000ffff`6ae0e990 00000000`00000000     0xffffffff`ffffffff
```

9. We close logging before exiting WinDbg Preview:

```
0:000> .logclose
Closing open log file 'C:\ALCDA2\A64\App6\App6.log'
```

Exercise A7

- **Goal:** Learn how to identify active threads
- **Patterns:** Divide by Zero (User Mode, x64); Invalid Pointer (General); Multiple Exceptions (User Mode); Near Exception
- \ALCDA-Dumps\Exercise-A7-x64-GDB.pdf

Exercise A7 (x64, GDB)

Goal: Learn how to identify active threads.

Patterns: Divide by Zero (User Mode); Invalid Pointer (General); Multiple Exceptions (User Mode); Near Exception.

1. Load *core.App7* dump file and *App7* executable from the x64/App7 directory:

```
~/ALCDA2/x64/App7$ gdb -c core.App7 -se App7
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App7...done.
[New LWP 71]
[New LWP 68]
[New LWP 69]
[New LWP 70]
[New LWP 73]
[New LWP 72]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App7'.
Program terminated with signal SIGFPE, Arithmetic exception.
#0  0x0000000000401c27 in procD ()
[Current thread is 1 (Thread 0x7f64c3075700 (LWP 71))]
```

2. List and identify the possible problem threads:

```
(gdb) info threads
  Id   Target Id                    Frame
* 1    Thread 0x7f64c3075700 (LWP 71) 0x0000000000401c27 in procD ()
  2    Thread 0x1d10880 (LWP 68)      0x0000000000441bf0 in nanosleep ()
  3    Thread 0x7f64c4077700 (LWP 69) 0x0000000000007265 in ?? ()
  4    Thread 0x7f64c3876700 (LWP 70) 0x0000000000441bf0 in nanosleep ()
  5    Thread 0x7f64c2073700 (LWP 73) 0x0000000000401bb8 in procF ()
  6    Thread 0x7f64c2874700 (LWP 72) 0x0000000000441bf0 in nanosleep ()
```

3. We see there is an arithmetic exception in the current thread. Let's list the stack trace for the current problem thread #1 and identify the problem instruction:

```
(gdb) bt
#0  0x0000000000401c27 in procD ()
#1  0x0000000000401c3f in procC ()
#2  0x0000000000401dfd in bar_three ()
#3  0x0000000000401e0e in foo_three ()
#4  0x0000000000401e27 in thread_three ()
#5  0x00000000004032b3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000044444f in clone ()

(gdb) x/i $rip
=> 0x401c27 <procD+14>: idivl  -0x8(%rbp)

(gdb) info r $rax
rax            0x1                 1

(gdb) x/w $rbp-0x8
0x7f64c3074d58: 0x00000000
```

4. We also see something abnormal on thread 3. Switch to it and check the currently executing instruction:

```
(gdb) thread 3
[Switching to thread 3 (Thread 0x7f64c4077700 (LWP 69))]
#0  0x0000000000007265 in ?? ()

(gdb) x/i $rip
=> 0x7265:      Cannot access memory at address 0x7265
```

Note: We see that the current instruction pointer points to an invalid memory address. It can also be considered NULL Pointer (Code) since the address belongs to an inaccessible part of memory to catch NULL pointers. We also see that there can be exceptions on different threads simultaneously.

5. Thread #5 looks active, and we compare the stack pointer with the stack region boundaries since we suspect stack overflow:

```
(gdb) thread 5
[Switching to thread 5 (Thread 0x7f64c2073700 (LWP 73))]
#0  0x0000000000401bb8 in procF () at pthread_create.c:688
688     pthread_create.c: No such file or directory.

(gdb) bt
#0  0x0000000000401bb8 in procF () at pthread_create.c:688
#1  0x0000000000401c05 in procF () at pthread_create.c:688
#2  0x0000000000401c05 in procF () at pthread_create.c:688
#3  0x0000000000401c05 in procF () at pthread_create.c:688
#4  0x0000000000401c05 in procF () at pthread_create.c:688
#5  0x0000000000401c05 in procF () at pthread_create.c:688
#6  0x0000000000401c05 in procF () at pthread_create.c:688
#7  0x0000000000401c05 in procF () at pthread_create.c:688
#8  0x0000000000401c05 in procF () at pthread_create.c:688
#9  0x0000000000401c05 in procF () at pthread_create.c:688
#10 0x0000000000401c05 in procF () at pthread_create.c:688
#11 0x0000000000401c05 in procF () at pthread_create.c:688
#12 0x0000000000401c05 in procF () at pthread_create.c:688
#13 0x0000000000401c05 in procF () at pthread_create.c:688
#14 0x0000000000401c05 in procF () at pthread_create.c:688
```

```
#15 0x0000000000401c05 in procF () at pthread_create.c:688
#16 0x0000000000401c05 in procF () at pthread_create.c:688
#17 0x0000000000401c05 in procF () at pthread_create.c:688
#18 0x0000000000401c05 in procF () at pthread_create.c:688
#19 0x0000000000401c05 in procF () at pthread_create.c:688
#20 0x0000000000401c05 in procF () at pthread_create.c:688
#21 0x0000000000401c05 in procF () at pthread_create.c:688
#22 0x0000000000401c05 in procF () at pthread_create.c:688
#23 0x0000000000401c05 in procF () at pthread_create.c:688
#24 0x0000000000401c05 in procF () at pthread_create.c:688
#25 0x0000000000401c05 in procF () at pthread_create.c:688
#26 0x0000000000401c05 in procF () at pthread_create.c:688
#27 0x0000000000401c05 in procF () at pthread_create.c:688
#28 0x0000000000401c16 in procE () at pthread_create.c:688
#29 0x0000000000401e8f in bar_five () at pthread_create.c:688
#30 0x0000000000401ea0 in foo_five () at pthread_create.c:688
#31 0x0000000000401eb9 in thread_five () at pthread_create.c:688
#32 0x00000000004032b3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#33 0x000000000044444f in clone ()

(gdb) x/i $rip
=> 0x401bb8 <procF+11>: mov    %edi,-0x1004(%rbp)

(gdb) x/a $rbp-0x1004
0x7f64c20569fc: 0x0

(gdb) disassemble $rip
Dump of assembler code for function procF:
   0x0000000000401bad <+0>:     push   %rbp
   0x0000000000401bae <+1>:     mov    %rsp,%rbp
   0x0000000000401bb1 <+4>:     sub    $0x1010,%rsp
=> 0x0000000000401bb8 <+11>:    mov    %edi,-0x1004(%rbp)
   0x0000000000401bbe <+17>:    lea    -0x1000(%rbp),%rdx
   0x0000000000401bc5 <+24>:    mov    $0x0,%eax
   0x0000000000401bca <+29>:    mov    $0x200,%ecx
   0x0000000000401bcf <+34>:    mov    %rdx,%rdi
   0x0000000000401bd2 <+37>:    rep stos %rax,%es:(%rdi)
   0x0000000000401bd5 <+40>:    movl   $0xffffffff,-0x1000(%rbp)
   0x0000000000401bdf <+50>:    mov    -0x1004(%rbp),%eax
   0x0000000000401be5 <+56>:    add    $0x1,%eax
   0x0000000000401be8 <+59>:    mov    %eax,-0xff8(%rbp)
   0x0000000000401bee <+65>:    movl   $0xffffffff,-0xff0(%rbp)
   0x0000000000401bf8 <+75>:    mov    -0xff8(%rbp),%eax
   0x0000000000401bfe <+81>:    mov    %eax,%edi
   0x0000000000401c00 <+83>:    callq  0x401bad <procF>
   0x0000000000401c05 <+88>:    nop
   0x0000000000401c06 <+89>:    leaveq
   0x0000000000401c07 <+90>:    retq
End of assembler dump.

(gdb) info r $rsp
rsp            0x7f64c20569f0      0x7f64c20569f0

(gdb) maintenance info sections
Exec file:
    `/home/coredump/ALCDA2/x64/App7/App7', file type elf64-x86-64.
 [0]     0x00400200->0x00400220 at 0x00000200: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]     0x00400220->0x00400244 at 0x00000220: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]     0x00400248->0x004004d0 at 0x00000248: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]     0x00401000->0x00401017 at 0x00001000: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
 [4]     0x00401018->0x004010f0 at 0x00001018: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
```

```
 [5]     0x004010f0->0x004935b0 at 0x000010f0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]     0x004935b0->0x00494157 at 0x000935b0: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]     0x00494158->0x00494161 at 0x00094158: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [8]     0x00495000->0x004af73c at 0x00095000: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
 [9]     0x004af740->0x004bbc10 at 0x000af740: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]     0x004bbc10->0x004bbcbc at 0x000bbc10: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]     0x004bd0b0->0x004bd0d8 at 0x000bc0b0: .tdata ALLOC LOAD DATA HAS_CONTENTS
 [12]     0x004bd0d8->0x004bd120 at 0x000bc0d8: .tbss ALLOC
 [13]     0x004bd0d8->0x004bd0e0 at 0x000bc0d8: .preinit_array ALLOC LOAD DATA HAS_CONTENTS
 [14]     0x004bd0e0->0x004bd0f0 at 0x000bc0e0: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [15]     0x004bd0f0->0x004bd100 at 0x000bc0f0: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [16]     0x004bd100->0x004bfef4 at 0x000bc100: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [17]     0x004bfef8->0x004c0000 at 0x000beef8: .got ALLOC LOAD DATA HAS_CONTENTS
 [18]     0x004c0000->0x004c00f0 at 0x000bf000: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [19]     0x004c0100->0x004c1c30 at 0x000bf100: .data ALLOC LOAD DATA HAS_CONTENTS
 [20]     0x004c1c30->0x004c1c90 at 0x000c0c30: __libc_subfreeres ALLOC LOAD DATA HAS_CONTENTS
 [21]     0x004c1ca0->0x004c2408 at 0x000c0ca0: __libc_IO_vtables ALLOC LOAD DATA HAS_CONTENTS
 [22]     0x004c2408->0x004c2410 at 0x000c1408: __libc_atexit ALLOC LOAD DATA HAS_CONTENTS
 [23]     0x004c2420->0x004c8528 at 0x000c1410: .bss ALLOC
 [24]     0x004c8528->0x004c8558 at 0x000c1410: __libc_freeres_ptrs ALLOC
 [25]     0x00000000->0x00000038 at 0x000c1410: .comment READONLY HAS_CONTENTS
 [26]     0x00000000->0x00000420 at 0x000c1450: .debug_aranges READONLY HAS_CONTENTS
 [27]     0x00000000->0x000372ad at 0x000c1870: .debug_info READONLY HAS_CONTENTS
 [28]     0x00000000->0x000057e8 at 0x000f8b1d: .debug_abbrev READONLY HAS_CONTENTS
 [29]     0x00000000->0x0000aa2b at 0x000fe305: .debug_line READONLY HAS_CONTENTS
 [30]     0x00000000->0x00004d08 at 0x00108d30: .debug_str READONLY HAS_CONTENTS
 [31]     0x00000000->0x0000d4b8 at 0x0010da38: .debug_loc READONLY HAS_CONTENTS
 [32]     0x00000000->0x000024c0 at 0x0011aef0: .debug_ranges READONLY HAS_CONTENTS
Core file:
    `/home/coredump/ALCDA2/x64/App7/core.App7', file type elf64-x86-64.
 [0]     0x00000000->0x00002c24 at 0x000004a0: note0 READONLY HAS_CONTENTS
 [1]     0x00000000->0x000000d8 at 0x00000524: .reg/71 HAS_CONTENTS
 [2]     0x00000000->0x000000d8 at 0x00000524: .reg HAS_CONTENTS
 [3]     0x00000000->0x00000080 at 0x000006b4: .note.linuxcore.siginfo/71 HAS_CONTENTS
 [4]     0x00000000->0x00000080 at 0x000006b4: .note.linuxcore.siginfo HAS_CONTENTS
 [5]     0x00000000->0x00000140 at 0x00000748: .auxv HAS_CONTENTS
 [6]     0x00000000->0x000000c4 at 0x0000089c: .note.linuxcore.file/71 HAS_CONTENTS
 [7]     0x00000000->0x000000c4 at 0x0000089c: .note.linuxcore.file HAS_CONTENTS
 [8]     0x00000000->0x00000200 at 0x00000974: .reg2/71 HAS_CONTENTS
 [9]     0x00000000->0x00000200 at 0x00000974: .reg2 HAS_CONTENTS
 [10]     0x00000000->0x00000340 at 0x00000b88: .reg-xstate/71 HAS_CONTENTS
 [11]     0x00000000->0x00000340 at 0x00000b88: .reg-xstate HAS_CONTENTS
 [12]     0x00000000->0x000000d8 at 0x00000f4c: .reg/68 HAS_CONTENTS
 [13]     0x00000000->0x00000200 at 0x00001040: .reg2/68 HAS_CONTENTS
 [14]     0x00000000->0x00000340 at 0x00001254: .reg-xstate/68 HAS_CONTENTS
 [15]     0x00000000->0x000000d8 at 0x00001618: .reg/69 HAS_CONTENTS
 [16]     0x00000000->0x00000200 at 0x0000170c: .reg2/69 HAS_CONTENTS
 [17]     0x00000000->0x00000340 at 0x00001920: .reg-xstate/69 HAS_CONTENTS
 [18]     0x00000000->0x000000d8 at 0x00001ce4: .reg/70 HAS_CONTENTS
 [19]     0x00000000->0x00000200 at 0x00001dd8: .reg2/70 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
 [20]     0x00000000->0x00000340 at 0x00001fec: .reg-xstate/70 HAS_CONTENTS
 [21]     0x00000000->0x000000d8 at 0x000023b0: .reg/73 HAS_CONTENTS
 [22]     0x00000000->0x00000200 at 0x000024a4: .reg2/73 HAS_CONTENTS
 [23]     0x00000000->0x00000340 at 0x000026b8: .reg-xstate/73 HAS_CONTENTS
 [24]     0x00000000->0x000000d8 at 0x00002a7c: .reg/72 HAS_CONTENTS
 [25]     0x00000000->0x00000200 at 0x00002b70: .reg2/72 HAS_CONTENTS
 [26]     0x00000000->0x00000340 at 0x00002d84: .reg-xstate/72 HAS_CONTENTS
 [27]     0x00400000->0x00401000 at 0x00004000: load1 ALLOC LOAD READONLY HAS_CONTENTS
 [28]     0x00401000->0x00401000 at 0x00005000: load2 ALLOC READONLY CODE
 [29]     0x00495000->0x00495000 at 0x00005000: load3 ALLOC READONLY
 [30]     0x004bd000->0x004c3000 at 0x00005000: load4 ALLOC LOAD HAS_CONTENTS
 [31]     0x004c3000->0x004c9000 at 0x0000b000: load5 ALLOC LOAD HAS_CONTENTS
 [32]     0x01d10000->0x01d33000 at 0x00011000: load6 ALLOC LOAD HAS_CONTENTS
 [33]     0x7f64c1873000->0x7f64c1873000 at 0x00034000: load7 ALLOC READONLY
 [34]     0x7f64c1874000->0x7f64c2074000 at 0x00034000: load8 ALLOC LOAD HAS_CONTENTS
 [35]     0x7f64c2074000->0x7f64c2074000 at 0x00834000: load9 ALLOC READONLY
```

```
[36]     0x7f64c2075000->0x7f64c2875000 at 0x00834000: load10 ALLOC LOAD HAS_CONTENTS
[37]     0x7f64c2875000->0x7f64c2875000 at 0x01034000: load11 ALLOC READONLY
[38]     0x7f64c2876000->0x7f64c3076000 at 0x01034000: load12 ALLOC LOAD HAS_CONTENTS
[39]     0x7f64c3076000->0x7f64c3076000 at 0x01834000: load13 ALLOC READONLY
[40]     0x7f64c3077000->0x7f64c3877000 at 0x01834000: load14 ALLOC LOAD HAS_CONTENTS
[41]     0x7f64c3877000->0x7f64c3877000 at 0x02034000: load15 ALLOC READONLY
[42]     0x7f64c3878000->0x7f64c4078000 at 0x02034000: load16 ALLOC LOAD HAS_CONTENTS
[43]     0x7ffdfcdd0000->0x7ffdfcdf1000 at 0x02834000: load17 ALLOC LOAD HAS_CONTENTS
[44]     0x7ffdfcdf5000->0x7ffdfcdf9000 at 0x02855000: load18 ALLOC LOAD READONLY HAS_CONTENTS
[45]     0x7ffdfcdf9000->0x7ffdfcdfa000 at 0x02859000: load19 ALLOC LOAD READONLY CODE HAS_CONTENTS
```

Note: We see that the stack pointer value **0x7f64c205**69f0 is inside the stack region address range **0x7f64c187**4000 - **0x7f64c207**4000.

Exercise A8

- **Goal:** Learn how to identify runtime exceptions, past execution residue and stack traces, identify handled exceptions
- **Patterns:** C++ Exception; Execution Residue (User Space); Past Stack Trace; Coincidental Symbolic Information; Handled Exception (User Space)
- \ALCDA-Dumps\Exercise-A8-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A8-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A8-A64-WinDbg.pdf

Exercise A8 (x64, GDB)

Goal: Learn how to identify runtime exceptions, past execution residue and stack traces, identify handled exceptions.

Patterns: C++ Exception; Execution Residue (User Space); Past Stack Trace; Coincidental Symbolic Information; Handled Exception (User Space).

1. Load *core.App8* dump file and *App8* executable from the x64/App8 directory:

```
~/ALCDA2/x64/App8$ gdb -c core.App8 -se App8
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App8...done.
[New LWP 162]
[New LWP 164]
[New LWP 165]
[New LWP 163]
[New LWP 161]
[New LWP 166]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App8'.
Program terminated with signal SIGABRT, Aborted.
#0  0x0000000000424fdb in raise ()
[Current thread is 1 (Thread 0x7f4d60082700 (LWP 162))]
```

1. Set logging to a file in case of lengthy output from some commands:

```
(gdb) set logging on App8.log
Copying output to App8.log.
```

2. List all thread stack traces:

```
(gdb) thread apply all bt

Thread 6 (Thread 0x7f4d5e07e700 (LWP 166)):
#0  0x000000000045aa70 in nanosleep ()
#1  0x000000000045a9fa in sleep ()
#2  0x00000000004023b6 in procNE() ()
#3  0x0000000000402482 in bar_five() ()
#4  0x000000000040248e in foo_five() ()
#5  0x00000000004024a2 in thread_five(void*) ()
```

```
#6  0x000000000041b483 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045d22f in clone ()

Thread 5 (Thread 0x19ad880 (LWP 161)):
#0  0x000000000045aa70 in nanosleep ()
#1  0x000000000045a9fa in sleep ()
#2  0x0000000000402553 in main () at pthread_create.c:688

Thread 4 (Thread 0x7f4d5f881700 (LWP 163)):
#0  0x000000000045aa70 in nanosleep ()
#1  0x000000000045a9fa in sleep ()
#2  0x00000000004023b6 in procNE() () at pthread_create.c:688
#3  0x00000000004023f2 in bar_two() () at pthread_create.c:688
#4  0x00000000004023fe in foo_two() () at pthread_create.c:688
#5  0x0000000000402412 in thread_two(void*) () at pthread_create.c:688
#6  0x000000000041b483 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045d22f in clone ()

Thread 3 (Thread 0x7f4d5e87f700 (LWP 165)):
#0  0x000000000045aa70 in nanosleep ()
#1  0x000000000045a9fa in sleep ()
#2  0x00000000004023b6 in procNE() () at pthread_create.c:688
#3  0x0000000000402452 in bar_four() () at pthread_create.c:688
#4  0x000000000040245e in foo_four() () at pthread_create.c:688
#5  0x0000000000402472 in thread_four(void*) () at pthread_create.c:688
#6  0x000000000041b483 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045d22f in clone ()

Thread 2 (Thread 0x7f4d5f080700 (LWP 164)):
#0  0x000000000045aa70 in nanosleep ()
#1  0x000000000045a9fa in sleep ()
#2  0x000000000040236c in procH() () at pthread_create.c:688
#3  0x0000000000402422 in bar_three() () at pthread_create.c:688
#4  0x000000000040242e in foo_three() () at pthread_create.c:688
#5  0x0000000000402442 in thread_three(void*) () at pthread_create.c:688
#6  0x000000000041b483 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045d22f in clone ()

Thread 1 (Thread 0x7f4d60082700 (LWP 162)):
#0  0x0000000000424fdb in raise ()
#1  0x00000000004017d3 in abort () at pthread_create.c:688
#2  0x0000000000401553 in __gnu_cxx::__verbose_terminate_handler() [clone .cold.1] ()
#3  0x0000000000402b46 in __cxxabiv1::__terminate(void (*)()) () at pthread_create.c:688
#4  0x0000000000402b81 in std::terminate() () at pthread_create.c:688
#5  0x0000000000402ac4 in __cxa_throw () at pthread_create.c:688
#6  0x00000000004022de in procB() () at pthread_create.c:688
#7  0x000000000040233b in procA() () at pthread_create.c:688
#8  0x00000000004023a0 in procNH() () at pthread_create.c:688
#9  0x00000000004023c2 in bar_one() () at pthread_create.c:688
--Type <RET> for more, q to quit, c to continue without paging--
#10 0x00000000004023ce in foo_one() () at pthread_create.c:688
#11 0x00000000004023e2 in thread_one(void*) () at pthread_create.c:688
#12 0x000000000041b483 in start_thread (arg=<optimized out>) at pthread_create.c:486
#13 0x000000000045d22f in clone ()
```

Note: We have C++ exception processing in thread #1.

3. Go to thread #4, identify the execution residue of *work* functions, check their correctness, and reconstruct the past stack trace:

```
(gdb) thread 4
[Switching to thread 4 (Thread 0x7f4d5f881700 (LWP 163))]
#0  0x000000000045aa70 in nanosleep ()

(gdb) bt
#0  0x000000000045aa70 in nanosleep ()
#1  0x000000000045a9fa in sleep ()
#2  0x00000000004023b6 in procNE() () at pthread_create.c:688
#3  0x00000000004023f2 in bar_two() () at pthread_create.c:688
#4  0x00000000004023fe in foo_two() () at pthread_create.c:688
#5  0x0000000000402412 in thread_two(void*) () at pthread_create.c:688
#6  0x000000000041b483 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045d22f in clone ()

(gdb) x/512a $rsp-2000
0x7f4d5f880530: 0x0     0x0
0x7f4d5f880540: 0x0     0x0
0x7f4d5f880550: 0x0     0x0
0x7f4d5f880560: 0x0     0x0
0x7f4d5f880570: 0x0     0x0
0x7f4d5f880580: 0x0     0x0
0x7f4d5f880590: 0x0     0x0
0x7f4d5f8805a0: 0x0     0x0
0x7f4d5f8805b0: 0x0     0x0
0x7f4d5f8805c0: 0x0     0x0
0x7f4d5f8805d0: 0x0     0x0
0x7f4d5f8805e0: 0x0     0x0
0x7f4d5f8805f0: 0x0     0x0
0x7f4d5f880600: 0x0     0x0
0x7f4d5f880610: 0x0     0x0
0x7f4d5f880620: 0x0     0x0
0x7f4d5f880630: 0x0     0x0
0x7f4d5f880640: 0x0     0x0
0x7f4d5f880650: 0x0     0x0
0x7f4d5f880660: 0x0     0x0
0x7f4d5f880670: 0x0     0x0
0x7f4d5f880680: 0x0     0x0
0x7f4d5f880690: 0x0     0x0
0x7f4d5f8806a0: 0x0     0x0
0x7f4d5f8806b0: 0x0     0x0
0x7f4d5f8806c0: 0x0     0x0
0x7f4d5f8806d0: 0x0     0x0
0x7f4d5f8806e0: 0x0     0x0
0x7f4d5f8806f0: 0x0     0x0
0x7f4d5f880700: 0x0     0x0
0x7f4d5f880710: 0x0     0x0
0x7f4d5f880720: 0x0     0x0
0x7f4d5f880730: 0x0     0x0
0x7f4d5f880740: 0x0     0x0
0x7f4d5f880750: 0x0     0x0
0x7f4d5f880760: 0x0     0x0
0x7f4d5f880770: 0x0     0x0
0x7f4d5f880780: 0x0     0x0
0x7f4d5f880790: 0x0     0x0
0x7f4d5f8807a0: 0x0     0x0
0x7f4d5f8807b0: 0x0     0x0
0x7f4d5f8807c0: 0x0     0x0
```

```
0x7f4d5f8807d0: 0x0     0x0
0x7f4d5f8807e0: 0x0     0x0
0x7f4d5f8807f0: 0x0     0x0
0x7f4d5f880800: 0x0     0x0
0x7f4d5f880810: 0x0     0x0
0x7f4d5f880820: 0x0     0x0
0x7f4d5f880830: 0x0     0x0
0x7f4d5f880840: 0x0     0x0
0x7f4d5f880850: 0x0     0x0
0x7f4d5f880860: 0x0     0x0
0x7f4d5f880870: 0x0     0x0
0x7f4d5f880880: 0x0     0x0
0x7f4d5f880890: 0x0     0x0
0x7f4d5f8808a0: 0x0     0x0
0x7f4d5f8808b0: 0x7f4d5f8808c0  0x4021dd <_Z6work_8v+9>
--Type <RET> for more, q to quit, c to continue without paging--
0x7f4d5f8808c0: 0x7f4d5f8808d0  0x4021e9 <_Z6work_7v+9>
0x7f4d5f8808d0: 0x7f4d5f8808e0  0x4021f5 <_Z6work_6v+9>
0x7f4d5f8808e0: 0x7f4d5f8808f0  0x402201 <_Z6work_5v+9>
0x7f4d5f8808f0: 0x7f4d5f880900  0x40220d <_Z6work_4v+9>
0x7f4d5f880900: 0x7f4d5f880910  0x402219 <_Z6work_3v+9>
0x7f4d5f880910: 0x7f4d5f880920  0x402225 <_Z6work_2v+9>
0x7f4d5f880920: 0x7f4d5f880930  0x402231 <_Z6work_1v+9>
0x7f4d5f880930: 0x7f4d5f880d40  0x402244 <_Z4workv+16>
0x7f4d5f880940: 0x0     0x0
0x7f4d5f880950: 0x0     0x0
0x7f4d5f880960: 0x0     0x0
0x7f4d5f880970: 0x0     0x0
0x7f4d5f880980: 0x0     0x0
0x7f4d5f880990: 0x0     0x0
0x7f4d5f8809a0: 0x0     0x0
0x7f4d5f8809b0: 0x0     0x0
0x7f4d5f8809c0: 0x0     0x0
0x7f4d5f8809d0: 0x0     0x0
0x7f4d5f8809e0: 0x0     0x0
0x7f4d5f8809f0: 0x0     0x0
0x7f4d5f880a00: 0x0     0x0
0x7f4d5f880a10: 0x0     0x0
0x7f4d5f880a20: 0x0     0x0
0x7f4d5f880a30: 0x0     0x0
0x7f4d5f880a40: 0x0     0x0
0x7f4d5f880a50: 0x0     0x0
0x7f4d5f880a60: 0x0     0x0
0x7f4d5f880a70: 0x0     0x0
0x7f4d5f880a80: 0x0     0x0
0x7f4d5f880a90: 0x0     0x0
0x7f4d5f880aa0: 0x0     0x0
0x7f4d5f880ab0: 0x0     0x0
0x7f4d5f880ac0: 0x0     0x0
0x7f4d5f880ad0: 0x0     0x0
0x7f4d5f880ae0: 0x0     0x0
0x7f4d5f880af0: 0x0     0x0
0x7f4d5f880b00: 0x0     0x0
0x7f4d5f880b10: 0x0     0x0
0x7f4d5f880b20: 0x0     0x0
0x7f4d5f880b30: 0x0     0x0
0x7f4d5f880b40: 0x0     0x0
0x7f4d5f880b50: 0x0     0x0
0x7f4d5f880b60: 0x0     0x0
0x7f4d5f880b70: 0x0     0x0
```

```
0x7f4d5f880b80: 0x0     0x0
0x7f4d5f880b90: 0x0     0x0
0x7f4d5f880ba0: 0x0     0x0
0x7f4d5f880bb0: 0x0     0x0
0x7f4d5f880bc0: 0x0     0x0
0x7f4d5f880bd0: 0x0     0x0
0x7f4d5f880be0: 0x0     0x0
0x7f4d5f880bf0: 0x0     0x0
0x7f4d5f880c00: 0x0     0x0
0x7f4d5f880c10: 0x0     0x0
0x7f4d5f880c20: 0x0     0x0
0x7f4d5f880c30: 0x0     0x0
0x7f4d5f880c40: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f4d5f880c50: 0x0     0x0
0x7f4d5f880c60: 0x0     0x0
0x7f4d5f880c70: 0x0     0x0
0x7f4d5f880c80: 0x0     0x0
0x7f4d5f880c90: 0x0     0x0
0x7f4d5f880ca0: 0x0     0x0
0x7f4d5f880cb0: 0x0     0x0
0x7f4d5f880cc0: 0x0     0x0
0x7f4d5f880cd0: 0x0     0x0
0x7f4d5f880ce0: 0x0     0x0
0x7f4d5f880cf0: 0x0     0x45aa61 <nanosleep+49>
0x7f4d5f880d00: 0x0     0x0
0x7f4d5f880d10: 0x0     0xfffffffffffffb8
0x7f4d5f880d20: 0x0     0x45a9fa <sleep+58>
0x7f4d5f880d30: 0xffffffff      0x3b87464c
0x7f4d5f880d40: 0x7f4d5f880d50  0x3e6ca15e6c37ea00
0x7f4d5f880d50: 0x7f4d5f880d60  0x0
0x7f4d5f880d60: 0x7f4d5f880d70  0x4023b6 <_Z6procNEv+19>
0x7f4d5f880d70: 0x7f4d5f880d80  0x4023f2 <_Z7bar_twov+9>
0x7f4d5f880d80: 0x7f4d5f880d90  0x4023fe <_Z7foo_twov+9>
0x7f4d5f880d90: 0x7f4d5f880db0  0x402412 <_Z10thread_twoPv+17>
0x7f4d5f880da0: 0x0     0x0
0x7f4d5f880db0: 0x0     0x41b483 <start_thread+243>
0x7f4d5f880dc0: 0x0     0x7f4d5f881700
0x7f4d5f880dd0: 0x7f4d5f881700  0xcaa581bf94a099a1
0x7f4d5f880de0: 0x7ffddda43bfe  0x7ffddda43bff
0x7f4d5f880df0: 0x7f4d5f881700  0x0
0x7f4d5f880e00: 0x343f3eaf8f2099a1      0xcaa5813cfcf299a1
0x7f4d5f880e10: 0x0     0x0
0x7f4d5f880e20: 0x0     0x0
0x7f4d5f880e30: 0x0     0x0
0x7f4d5f880e40: 0x0     0x0
0x7f4d5f880e50: 0x0     0x3e6ca15e6c37ea00
0x7f4d5f880e60: 0x0     0x7f4d5f881700
0x7f4d5f880e70: 0x7f4d5f881700  0x45d22f <clone+63>
0x7f4d5f880e80: 0x0     0x0
0x7f4d5f880e90: 0x0     0x0
0x7f4d5f880ea0: 0x0     0x0
0x7f4d5f880eb0: 0x0     0x0
0x7f4d5f880ec0: 0x0     0x0
0x7f4d5f880ed0: 0x0     0x0
0x7f4d5f880ee0: 0x0     0x0
0x7f4d5f880ef0: 0x0     0x0
0x7f4d5f880f00: 0x0     0x0
0x7f4d5f880f10: 0x0     0x0
0x7f4d5f880f20: 0x0     0x0
```

```
0x7f4d5f880f30: 0x0     0x0
0x7f4d5f880f40: 0x0     0x0
0x7f4d5f880f50: 0x0     0x0
0x7f4d5f880f60: 0x0     0x0
0x7f4d5f880f70: 0x0     0x0
0x7f4d5f880f80: 0x0     0x0
0x7f4d5f880f90: 0x0     0x0
0x7f4d5f880fa0: 0x0     0x0
0x7f4d5f880fb0: 0x0     0x0
0x7f4d5f880fc0: 0x0     0x0
0x7f4d5f880fd0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f4d5f880fe0: 0x0     0x0
0x7f4d5f880ff0: 0x0     0x0
0x7f4d5f881000: 0x0     0x0
0x7f4d5f881010: 0x0     0x0
0x7f4d5f881020: 0x0     0x0
0x7f4d5f881030: 0x0     0x0
0x7f4d5f881040: 0x0     0x0
0x7f4d5f881050: 0x0     0x0
0x7f4d5f881060: 0x0     0x0
0x7f4d5f881070: 0x0     0x0
0x7f4d5f881080: 0x0     0x0
0x7f4d5f881090: 0x0     0x0
0x7f4d5f8810a0: 0x0     0x0
0x7f4d5f8810b0: 0x0     0x0
0x7f4d5f8810c0: 0x0     0x0
0x7f4d5f8810d0: 0x0     0x0
0x7f4d5f8810e0: 0x0     0x0
0x7f4d5f8810f0: 0x0     0x0
0x7f4d5f881100: 0x0     0x0
0x7f4d5f881110: 0x0     0x0
0x7f4d5f881120: 0x0     0x0
0x7f4d5f881130: 0x0     0x0
0x7f4d5f881140: 0x0     0x0
0x7f4d5f881150: 0x0     0x0
0x7f4d5f881160: 0x0     0x0
0x7f4d5f881170: 0x0     0x0
0x7f4d5f881180: 0x0     0x0
0x7f4d5f881190: 0x0     0x0
0x7f4d5f8811a0: 0x0     0x0
0x7f4d5f8811b0: 0x0     0x0
0x7f4d5f8811c0: 0x0     0x0
0x7f4d5f8811d0: 0x0     0x0
0x7f4d5f8811e0: 0x0     0x0
0x7f4d5f8811f0: 0x0     0x0
0x7f4d5f881200: 0x0     0x0
0x7f4d5f881210: 0x0     0x0
0x7f4d5f881220: 0x0     0x0
0x7f4d5f881230: 0x0     0x0
0x7f4d5f881240: 0x0     0x0
0x7f4d5f881250: 0x0     0x0
0x7f4d5f881260: 0x0     0x0
0x7f4d5f881270: 0x0     0x0
0x7f4d5f881280: 0x0     0x0
0x7f4d5f881290: 0x0     0x0
0x7f4d5f8812a0: 0x0     0x0
0x7f4d5f8812b0: 0x0     0x0
0x7f4d5f8812c0: 0x0     0x0
0x7f4d5f8812d0: 0x0     0x0
```

```
0x7f4d5f8812e0: 0x0     0x0
0x7f4d5f8812f0: 0x0     0x0
0x7f4d5f881300: 0x0     0x0
0x7f4d5f881310: 0x0     0x0
0x7f4d5f881320: 0x0     0x0
0x7f4d5f881330: 0x0     0x0
0x7f4d5f881340: 0x0     0x0
0x7f4d5f881350: 0x0     0x0
0x7f4d5f881360: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f4d5f881370: 0x0     0x0
0x7f4d5f881380: 0x0     0x0
0x7f4d5f881390: 0x0     0x0
0x7f4d5f8813a0: 0x0     0x0
0x7f4d5f8813b0: 0x0     0x0
0x7f4d5f8813c0: 0x0     0x0
0x7f4d5f8813d0: 0x0     0x0
0x7f4d5f8813e0: 0x0     0x0
0x7f4d5f8813f0: 0x0     0x0
0x7f4d5f881400: 0x0     0x0
0x7f4d5f881410: 0x0     0x0
0x7f4d5f881420: 0x0     0x0
0x7f4d5f881430: 0x0     0x0
0x7f4d5f881440: 0x0     0x0
0x7f4d5f881450: 0x0     0x0
0x7f4d5f881460: 0x0     0x0
0x7f4d5f881470: 0x0     0x0
0x7f4d5f881480: 0x0     0x0
0x7f4d5f881490: 0x0     0x0
0x7f4d5f8814a0: 0x0     0x0
0x7f4d5f8814b0: 0x0     0x0
0x7f4d5f8814c0: 0x0     0x0
0x7f4d5f8814d0: 0x0     0x0
0x7f4d5f8814e0: 0x0     0x0
0x7f4d5f8814f0: 0x0     0x0
0x7f4d5f881500: 0x0     0x0
0x7f4d5f881510: 0x0     0x0
0x7f4d5f881520: 0x0     0x0
```

```
(gdb) disassemble 0x402219
Dump of assembler code for function _Z6work_3v:
   0x0000000000402210 <+0>:     push   %rbp
   0x0000000000402211 <+1>:     mov    %rsp,%rbp
   0x0000000000402214 <+4>:     callq  0x402204 <_Z6work_4v>
   0x0000000000402219 <+9>:     nop
   0x000000000040221a <+10>:    pop    %rbp
   0x000000000040221b <+11>:    retq
End of assembler dump.
```

Note: Since the saved %rbp register value points to the next line, we can easily reconstruct the fragment of the past stack trace:

```
0x7f4d5f8808b0: 0x7f4d5f8808c0  0x4021dd <_Z6work_8v+9>
0x7f4d5f8808c0: 0x7f4d5f8808d0  0x4021e9 <_Z6work_7v+9>
0x7f4d5f8808d0: 0x7f4d5f8808e0  0x4021f5 <_Z6work_6v+9>
0x7f4d5f8808e0: 0x7f4d5f8808f0  0x402201 <_Z6work_5v+9>
0x7f4d5f8808f0: 0x7f4d5f880900  0x40220d <_Z6work_4v+9>
0x7f4d5f880900: 0x7f4d5f880910  0x402219 <_Z6work_3v+9>
0x7f4d5f880910: 0x7f4d5f880920  0x402225 <_Z6work_2v+9>
0x7f4d5f880920: 0x7f4d5f880930  0x402231 <_Z6work_1v+9>
```

```
0x7f4d5f880930: 0x7f4d5f880d40  0x402244 <_Z4workv+16>
```

4. Go to thread #2, identify the handled exception processing code, and check its validity:

```
(gdb) thread 2
[Switching to thread 2 (Thread 0x7f4d5f080700 (LWP 164))]
#0  0x000000000045aa70 in nanosleep ()

(gdb) bt
#0  0x000000000045aa70 in nanosleep ()
#1  0x000000000045a9fa in sleep ()
#2  0x000000000040236c in procH() () at pthread_create.c:688
#3  0x0000000000402422 in bar_three() () at pthread_create.c:688
#4  0x000000000040242e in foo_three() () at pthread_create.c:688
#5  0x0000000000402442 in thread_three(void*) () at pthread_create.c:688
#6  0x000000000041b483 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045d22f in clone ()

(gdb) x/512a $rsp-2000
0x7f4d5f07f520: 0x418950 <fde_single_encoding_compare>  0x418600 <frame_downheap+112>
0x7f4d5f07f530: 0x7f4d58002520  0x4df460 <object.6779>
0x7f4d5f07f540: 0x7f4d580044c0  0xe3
0x7f4d5f07f550: 0x1130000038c   0x0
0x7f4d5f07f560: 0x7f4d580044b0  0x4df460 <object.6779>
0x7f4d5f07f570: 0x1cb70 0x6cb
0x7f4d5f07f580: 0x7f4d580044a0  0x4186e1 <frame_heapsort+145>
0x7f4d5f07f590: 0x38b8  0x3e6ca15e6c37ea00
0x7f4d5f07f5a0: 0x401b72 <read_encoded_value_with_base.cold.0>  0x418950
<fde_single_encoding_compare>
0x7f4d5f07f5b0: 0x4df460 <object.6779>  0x49
0x7f4d5f07f5c0: 0x7f4d58000be0  0x417d0d <_Unwind_RaiseException+61>
0x7f4d5f07f5d0: 0x4d4dd8        0x4197e6 <search_object+854>
0x7f4d5f07f5e0: 0x0     0x7f4d5f07f628
0x7f4d5f07f5f0: 0x7f4d5f07f630  0x7f4d0000001b
0x7f4d5f07f600: 0x0     0x7f4d0000000b
0x7f4d5f07f610: 0x7f4d5f07fc88  0x7f4d5f07fc90
0x7f4d5f07f620: 0x0     0x40234a <_Z5procHv>
0x7f4d5f07f630: 0x43    0x0
0x7f4d5f07f640: 0x7f4d5f07fd40  0x4df460 <object.6779>
0x7f4d5f07f650: 0x7f4d5f07f760  0x7f4d5f07fab8
0x7f4d5f07f660: 0x41db80 <__pthread_key_create> 0x4030b8 <__gxx_personality_v0+184>
0x7f4d5f07f670: 0x402357 <_Z5procHv+13> 0x7f4d58000b80
0x7f4d5f07f680: 0x60140234a     0x0
0x7f4d5f07f690: 0x4d60c0        0x7f4d5f07f601
0x7f4d5f07f6a0: 0x7f4d5f07fd50  0x40233b <_Z5procAv+9>
0x7f4d5f07f6b0: 0x0     0x4c43d9
0x7f4d5f07f6c0: 0x4c43d9        0x5f07fa10
0x7f4d5f07f6d0: 0x0     0x1b
0x7f4d5f07f6e0: 0x7f4d5f07f760  0x4165d0 <uw_frame_state_for+800>
0x7f4d5f07f6f0: 0x0     0x9b00000000
0x7f4d5f07f700: 0x4c4339        0x7f4d5f07f718
0x7f4d5f07f710: 0x7f4d5f07f760  0x4d60c0
0x7f4d5f07f720: 0x7f4d5f07f760  0x7f4d5f07fa10
0x7f4d5f07f730: 0x4     0x7f4d58000b80
0x7f4d5f07f740: 0x7f4d5f07f760  0x4
0x7f4d5f07f750: 0x7f4d5f07fb00  0x4176fb <_Unwind_RaiseException_Phase2+75>
0x7f4d5f07f760: 0x0     0x0
0x7f4d5f07f770: 0x0     0x0
0x7f4d5f07f780: 0x0     0x0
```

```
0x7f4d5f07f790: 0xfffffffffffffe8      0x1
0x7f4d5f07f7a0: 0x0     0x0
0x7f4d5f07f7b0: 0x0     0x0
0x7f4d5f07f7c0: 0xfffffffffffffff0      0x1
0x7f4d5f07f7d0: 0x0     0x0
0x7f4d5f07f7e0: 0x0     0x0
0x7f4d5f07f7f0: 0x0     0x0
0x7f4d5f07f800: 0x0     0x0
0x7f4d5f07f810: 0x0     0x0
0x7f4d5f07f820: 0x0     0x0
0x7f4d5f07f830: 0x0     0x0
0x7f4d5f07f840: 0x0     0x0
0x7f4d5f07f850: 0x0     0x0
0x7f4d5f07f860: 0xfffffffffffffff8      0x1
0x7f4d5f07f870: 0x0     0x0
0x7f4d5f07f880: 0x0     0x10
0x7f4d5f07f890: 0x6     0x0
0x7f4d5f07f8a0: 0x1     0x40238c <_Z5procHv+66>
--Type <RET> for more, q to quit, c to continue without paging--
0x7f4d5f07f8b0: 0x403000 <__gxx_personality_v0> 0xfffffffffffffff8
0x7f4d5f07f8c0: 0x1     0x10
0x7f4d5f07f8d0: 0x11b1b 0x0
0x7f4d5f07f8e0: 0x7f4d58000b80  0x7f4d5f07fa10
0x7f4d5f07f8f0: 0x7f4d5f07fcc0  0x7f4d5f07fb00
0x7f4d5f07f900: 0x7f4d58000b80  0x7f4d5f07fd60
0x7f4d5f07f910: 0x0     0x417faa <_Unwind_RaiseException+730>
0x7f4d5f07f920: 0x7f4d5f07fc88  0x7f4d5f07fc90
0x7f4d5f07f930: 0x0     0x7f4d5f07fc98
0x7f4d5f07f940: 0x0     0x0
0x7f4d5f07f950: 0x7f4d5f07fcc0  0x0
0x7f4d5f07f960: 0x0     0x0
0x7f4d5f07f970: 0x0     0x0
0x7f4d5f07f980: 0x7f4d5f07fca0  0x7f4d5f07fca8
0x7f4d5f07f990: 0x7f4d5f07fcb0  0x7f4d5f07fcb8
0x7f4d5f07f9a0: 0x7f4d5f07fcc8  0x0
0x7f4d5f07f9b0: 0x7f4d5f07fcd0  0x402ab7 <__cxa_throw+55>
0x7f4d5f07f9c0: 0x0     0x0
0x7f4d5f07f9d0: 0x0     0x417cd0 <_Unwind_RaiseException>
0x7f4d5f07f9e0: 0x4000000000000000      0x0
0x7f4d5f07f9f0: 0x0     0x0
0x7f4d5f07fa00: 0x0     0x0
0x7f4d5f07fa10: 0x7f4d5f07fc88  0x7f4d5f07fc90
0x7f4d5f07fa20: 0x0     0x7f4d5f07fd28
0x7f4d5f07fa30: 0x0     0x0
0x7f4d5f07fa40: 0x7f4d5f07fd50  0x7f4d5f07f908
0x7f4d5f07fa50: 0x0     0x0
0x7f4d5f07fa60: 0x0     0x0
0x7f4d5f07fa70: 0x7f4d5f07fd30  0x7f4d5f07fd38
0x7f4d5f07fa80: 0x7f4d5f07fcb0  0x7f4d5f07fcb8
0x7f4d5f07fa90: 0x7f4d5f07fd58  0x0
0x7f4d5f07faa0: 0x7f4d5f07fd60  0x40235a <_Z5procHv+16>
0x7f4d5f07fab0: 0x4d60c0        0x0
0x7f4d5f07fac0: 0x0     0x40234a <_Z5procHv>
0x7f4d5f07fad0: 0x4000000000000000      0x0
0x7f4d5f07fae0: 0x0     0x0
0x7f4d5f07faf0: 0x0     0x0
0x7f4d5f07fb00: 0x4     0x0
0x7f4d5f07fb10: 0x0     0x0
0x7f4d5f07fb20: 0x0     0x0
0x7f4d5f07fb30: 0xfffffffffffffe8      0x1
```

```
0x7f4d5f07fb40: 0x0     0x0
0x7f4d5f07fb50: 0x0     0x0
0x7f4d5f07fb60: 0xfffffffffffffff0      0x1
0x7f4d5f07fb70: 0x0     0x0
0x7f4d5f07fb80: 0x0     0x0
0x7f4d5f07fb90: 0x0     0x0
0x7f4d5f07fba0: 0x0     0x0
0x7f4d5f07fbb0: 0x0     0x0
0x7f4d5f07fbc0: 0x0     0x0
0x7f4d5f07fbd0: 0x0     0x0
0x7f4d5f07fbe0: 0x0     0x0
0x7f4d5f07fbf0: 0x0     0x0
0x7f4d5f07fc00: 0xfffffffffffffff8      0x1
0x7f4d5f07fc10: 0x0     0x0
0x7f4d5f07fc20: 0x0     0x10
0x7f4d5f07fc30: 0x6     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f4d5f07fc40: 0x1     0x40238c <_Z5procHv+66>
0x7f4d5f07fc50: 0x403000 <__gxx_personality_v0> 0xfffffffffffffff8
0x7f4d5f07fc60: 0x1     0x10
0x7f4d5f07fc70: 0x11b1b 0x0
0x7f4d5f07fc80: 0x88    0x7f4d58000b80
0x7f4d5f07fc90: 0x1     0x0
0x7f4d5f07fca0: 0x7ffddda43bfe  0x7ffddda43bff
0x7f4d5f07fcb0: 0x7f4d5f080700  0x0
0x7f4d5f07fcc0: 0x7f4d5f07fd70  0x402358 <_Z5procHv+14>
0x7f4d5f07fcd0: 0x7f4d58000ba0  0x7f4d5f07fd40
0x7f4d5f07fce0: 0x7f4d58000bb0  0x45aa61 <nanosleep+49>
0x7f4d5f07fcf0: 0x7f4d5f07fd00  0xd
0x7f4d5f07fd00: 0x4420737365636341      0xfffffffffffffb8
0x7f4d5f07fd10: 0x0     0x45a9fa <sleep+58>
0x7f4d5f07fd20: 0xffffffff      0x3b9715f4
0x7f4d5f07fd30: 0x7ffddda43bfe  0x3e6ca15e6c37ea00
0x7f4d5f07fd40: 0x7f4d5f07fd50  0x0
0x7f4d5f07fd50: 0x7f4d5f07fd70  0x40236c <_Z5procHv+34>
0x7f4d5f07fd60: 0x0     0x0
0x7f4d5f07fd70: 0x7f4d5f07fd80  0x402422 <_Z9bar_threev+9>
0x7f4d5f07fd80: 0x7f4d5f07fd90  0x40242e <_Z9foo_threev+9>
0x7f4d5f07fd90: 0x7f4d5f07fdb0  0x402442 <_Z12thread_threePv+17>
0x7f4d5f07fda0: 0x0     0x0
0x7f4d5f07fdb0: 0x0     0x41b483 <start_thread+243>
0x7f4d5f07fdc0: 0x0     0x7f4d5f080700
0x7f4d5f07fdd0: 0x7f4d5f080700  0xcaa581bf94a099a1
0x7f4d5f07fde0: 0x7ffddda43bfe  0x7ffddda43bff
0x7f4d5f07fdf0: 0x7f4d5f080700  0x0
0x7f4d5f07fe00: 0x343f3fb06f2099a1      0xcaa5813cfcf299a1
0x7f4d5f07fe10: 0x0     0x0
0x7f4d5f07fe20: 0x0     0x0
0x7f4d5f07fe30: 0x0     0x0
0x7f4d5f07fe40: 0x0     0x0
0x7f4d5f07fe50: 0x0     0x3e6ca15e6c37ea00
0x7f4d5f07fe60: 0x0     0x7f4d5f080700
0x7f4d5f07fe70: 0x7f4d5f080700  0x45d22f <clone+63>
0x7f4d5f07fe80: 0x0     0x0
0x7f4d5f07fe90: 0x0     0x0
0x7f4d5f07fea0: 0x0     0x0
0x7f4d5f07feb0: 0x0     0x0
0x7f4d5f07fec0: 0x0     0x0
0x7f4d5f07fed0: 0x0     0x0
0x7f4d5f07fee0: 0x0     0x0
```

```
0x7f4d5f07fef0: 0x0     0x0
0x7f4d5f07ff00: 0x0     0x0
0x7f4d5f07ff10: 0x0     0x0
0x7f4d5f07ff20: 0x0     0x0
0x7f4d5f07ff30: 0x0     0x0
0x7f4d5f07ff40: 0x0     0x0
0x7f4d5f07ff50: 0x0     0x0
0x7f4d5f07ff60: 0x0     0x0
0x7f4d5f07ff70: 0x0     0x0
0x7f4d5f07ff80: 0x0     0x0
0x7f4d5f07ff90: 0x0     0x0
0x7f4d5f07ffa0: 0x0     0x0
0x7f4d5f07ffb0: 0x0     0x0
0x7f4d5f07ffc0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f4d5f07ffd0: 0x0     0x0
0x7f4d5f07ffe0: 0x0     0x0
0x7f4d5f07fff0: 0x0     0x0
0x7f4d5f080000: 0x0     0x0
0x7f4d5f080010: 0x0     0x0
0x7f4d5f080020: 0x0     0x0
0x7f4d5f080030: 0x0     0x0
0x7f4d5f080040: 0x0     0x0
0x7f4d5f080050: 0x0     0x0
0x7f4d5f080060: 0x0     0x0
0x7f4d5f080070: 0x0     0x0
0x7f4d5f080080: 0x0     0x0
0x7f4d5f080090: 0x0     0x0
0x7f4d5f0800a0: 0x0     0x0
0x7f4d5f0800b0: 0x0     0x0
0x7f4d5f0800c0: 0x0     0x0
0x7f4d5f0800d0: 0x0     0x0
0x7f4d5f0800e0: 0x0     0x0
0x7f4d5f0800f0: 0x0     0x0
0x7f4d5f080100: 0x0     0x0
0x7f4d5f080110: 0x0     0x0
0x7f4d5f080120: 0x0     0x0
0x7f4d5f080130: 0x0     0x0
0x7f4d5f080140: 0x0     0x0
0x7f4d5f080150: 0x0     0x0
0x7f4d5f080160: 0x0     0x0
0x7f4d5f080170: 0x0     0x0
0x7f4d5f080180: 0x0     0x0
0x7f4d5f080190: 0x0     0x0
0x7f4d5f0801a0: 0x0     0x0
0x7f4d5f0801b0: 0x0     0x0
0x7f4d5f0801c0: 0x0     0x0
0x7f4d5f0801d0: 0x0     0x0
0x7f4d5f0801e0: 0x0     0x0
0x7f4d5f0801f0: 0x0     0x0
0x7f4d5f080200: 0x0     0x0
0x7f4d5f080210: 0x0     0x0
0x7f4d5f080220: 0x0     0x0
0x7f4d5f080230: 0x0     0x0
0x7f4d5f080240: 0x0     0x0
0x7f4d5f080250: 0x0     0x0
0x7f4d5f080260: 0x0     0x0
0x7f4d5f080270: 0x0     0x0
0x7f4d5f080280: 0x0     0x0
0x7f4d5f080290: 0x0     0x0
```

```
0x7f4d5f0802a0: 0x0     0x0
0x7f4d5f0802b0: 0x0     0x0
0x7f4d5f0802c0: 0x0     0x0
0x7f4d5f0802d0: 0x0     0x0
0x7f4d5f0802e0: 0x0     0x0
0x7f4d5f0802f0: 0x0     0x0
0x7f4d5f080300: 0x0     0x0
0x7f4d5f080310: 0x0     0x0
0x7f4d5f080320: 0x0     0x0
0x7f4d5f080330: 0x0     0x0
0x7f4d5f080340: 0x0     0x0
0x7f4d5f080350: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f4d5f080360: 0x0     0x0
0x7f4d5f080370: 0x0     0x0
0x7f4d5f080380: 0x0     0x0
0x7f4d5f080390: 0x0     0x0
0x7f4d5f0803a0: 0x0     0x0
0x7f4d5f0803b0: 0x0     0x0
0x7f4d5f0803c0: 0x0     0x0
0x7f4d5f0803d0: 0x0     0x0
0x7f4d5f0803e0: 0x0     0x0
0x7f4d5f0803f0: 0x0     0x0
0x7f4d5f080400: 0x0     0x0
0x7f4d5f080410: 0x0     0x0
0x7f4d5f080420: 0x0     0x0
0x7f4d5f080430: 0x0     0x0
0x7f4d5f080440: 0x0     0x0
0x7f4d5f080450: 0x0     0x0
0x7f4d5f080460: 0x0     0x0
0x7f4d5f080470: 0x0     0x0
0x7f4d5f080480: 0x0     0x0
0x7f4d5f080490: 0x0     0x0
0x7f4d5f0804a0: 0x0     0x0
0x7f4d5f0804b0: 0x0     0x0
0x7f4d5f0804c0: 0x0     0x0
0x7f4d5f0804d0: 0x0     0x0
0x7f4d5f0804e0: 0x0     0x0
0x7f4d5f0804f0: 0x0     0x0
0x7f4d5f080500: 0x0     0x0
0x7f4d5f080510: 0x0     0x0

(gdb) disassemble 0x417faa
Dump of assembler code for function _Unwind_RaiseException:
   0x0000000000417cd0 <+0>:     push   %rbp
   0x0000000000417cd1 <+1>:     mov    %rsp,%rbp
   0x0000000000417cd4 <+4>:     push   %r15
   0x0000000000417cd6 <+6>:     push   %r14
   0x0000000000417cd8 <+8>:     lea    -0x3a0(%rbp),%r14
   0x0000000000417cdf <+15>:    lea    0x10(%rbp),%rsi
   0x0000000000417ce3 <+19>:    push   %r13
   0x0000000000417ce5 <+21>:    mov    %rdi,%r13
   0x0000000000417ce8 <+24>:    mov    %r14,%rdi
   0x0000000000417ceb <+27>:    push   %r12
   0x0000000000417ced <+29>:    lea    -0x1c0(%rbp),%r12
   0x0000000000417cf4 <+36>:    push   %rbx
   0x0000000000417cf5 <+37>:    lea    -0x2b0(%rbp),%rbx
   0x0000000000417cfc <+44>:    push   %rdx
   0x0000000000417cfd <+45>:    push   %rax
   0x0000000000417cfe <+46>:    sub    $0x368,%rsp
```

```
   0x0000000000417d05 <+53>:    mov    0x8(%rbp),%rdx
   0x0000000000417d09 <+57>:    callq  0x4174a0 <uw_init_context_1>
   0x0000000000417d0e <+62>:    movdqa -0x3a0(%rbp),%xmm0
   0x0000000000417d16 <+70>:    movdqa -0x390(%rbp),%xmm1
   0x0000000000417d1e <+78>:    movdqa -0x380(%rbp),%xmm2
   0x0000000000417d26 <+86>:    movdqa -0x370(%rbp),%xmm3
   0x0000000000417d2e <+94>:    movdqa -0x360(%rbp),%xmm4
   0x0000000000417d36 <+102>:   movdqa -0x350(%rbp),%xmm5
   0x0000000000417d3e <+110>:   movaps %xmm0,-0x2b0(%rbp)
   0x0000000000417d45 <+117>:   movdqa -0x340(%rbp),%xmm6
   0x0000000000417d4d <+125>:   movaps %xmm1,-0x2a0(%rbp)
   0x0000000000417d54 <+132>:   movdqa -0x330(%rbp),%xmm7
   0x0000000000417d5c <+140>:   movaps %xmm2,-0x290(%rbp)
   0x0000000000417d63 <+147>:   movdqa -0x320(%rbp),%xmm0
   0x0000000000417d6b <+155>:   movdqa -0x310(%rbp),%xmm1
   0x0000000000417d73 <+163>:   movaps %xmm3,-0x280(%rbp)
   0x0000000000417d7a <+170>:   movdqa -0x300(%rbp),%xmm2
   0x0000000000417d82 <+178>:   movdqa -0x2f0(%rbp),%xmm3
   0x0000000000417d8a <+186>:   movaps %xmm4,-0x270(%rbp)
   0x0000000000417d91 <+193>:   movdqa -0x2e0(%rbp),%xmm4
   0x0000000000417d99 <+201>:   movaps %xmm5,-0x260(%rbp)
   0x0000000000417da0 <+208>:   movdqa -0x2d0(%rbp),%xmm5
   0x0000000000417da8 <+216>:   movaps %xmm6,-0x250(%rbp)
   0x0000000000417daf <+223>:   movdqa -0x2c0(%rbp),%xmm6
   0x0000000000417db7 <+231>:   movaps %xmm7,-0x240(%rbp)
   0x0000000000417dbe <+238>:   movaps %xmm0,-0x230(%rbp)
   0x0000000000417dc5 <+245>:   movaps %xmm1,-0x220(%rbp)
   0x0000000000417dcc <+252>:   movaps %xmm2,-0x210(%rbp)
   0x0000000000417dd3 <+259>:   movaps %xmm3,-0x200(%rbp)
   0x0000000000417dda <+266>:   movaps %xmm4,-0x1f0(%rbp)
   0x0000000000417de1 <+273>:   movaps %xmm5,-0x1e0(%rbp)
   0x0000000000417de8 <+280>:   movaps %xmm6,-0x1d0(%rbp)
   0x0000000000417def <+287>:   jmp    0x417e30 <_Unwind_RaiseException+352>
   0x0000000000417df1 <+289>:   nopl   0x0(%rax)
   0x0000000000417df8 <+296>:   test   %eax,%eax
   0x0000000000417dfa <+298>:   jne    0x417e60 <_Unwind_RaiseException+400>
   0x0000000000417dfc <+300>:   mov    -0x70(%rbp),%rax
   0x0000000000417e00 <+304>:   test   %rax,%rax
   0x0000000000417e03 <+307>:   je     0x417e25 <_Unwind_RaiseException+341>
   0x0000000000417e05 <+309>:   mov    %rbx,%r8
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000417e08 <+312>:   mov    %r13,%rcx
   0x0000000000417e0b <+315>:   mov    0x0(%r13),%rdx
   0x0000000000417e0f <+319>:   mov    $0x1,%esi
   0x0000000000417e14 <+324>:   mov    $0x1,%edi
   0x0000000000417e19 <+329>:   callq  *%rax
   0x0000000000417e1b <+331>:   cmp    $0x6,%eax
   0x0000000000417e1e <+334>:   je     0x417e70 <_Unwind_RaiseException+416>
   0x0000000000417e20 <+336>:   cmp    $0x8,%eax
   0x0000000000417e23 <+339>:   jne    0x417e60 <_Unwind_RaiseException+400>
   0x0000000000417e25 <+341>:   mov    %r12,%rsi
   0x0000000000417e28 <+344>:   mov    %rbx,%rdi
   0x0000000000417e2b <+347>:   callq  0x417620 <uw_update_context>
   0x0000000000417e30 <+352>:   mov    %r12,%rsi
   0x0000000000417e33 <+355>:   mov    %rbx,%rdi
   0x0000000000417e36 <+358>:   callq  0x4162b0 <uw_frame_state_for>
   0x0000000000417e3b <+363>:   cmp    $0x5,%eax
   0x0000000000417e3e <+366>:   jne    0x417df8 <_Unwind_RaiseException+296>
   0x0000000000417e40 <+368>:   mov    -0x28(%rbp),%rbx
   0x0000000000417e44 <+372>:   mov    -0x20(%rbp),%r12
```

```
   0x0000000000417e48 <+376>:   mov    -0x18(%rbp),%r13
   0x0000000000417e4c <+380>:   mov    -0x10(%rbp),%r14
   0x0000000000417e50 <+384>:   mov    -0x8(%rbp),%r15
   0x0000000000417e54 <+388>:   leaveq
   0x0000000000417e55 <+389>:   retq
   0x0000000000417e56 <+390>:   nopw   %cs:0x0(%rax,%rax,1)
   0x0000000000417e60 <+400>:   mov    $0x3,%eax
   0x0000000000417e65 <+405>:   jmp    0x417e40 <_Unwind_RaiseException+368>
   0x0000000000417e67 <+407>:   nopw   0x0(%rax,%rax,1)
   0x0000000000417e70 <+416>:   movdqa -0x3a0(%rbp),%xmm7
   0x0000000000417e78 <+424>:   mov    -0x1f0(%rbp),%rax
   0x0000000000417e7f <+431>:   movq   $0x0,0x10(%r13)
   0x0000000000417e87 <+439>:   mov    %r12,%rdx
   0x0000000000417e8a <+442>:   movdqa -0x350(%rbp),%xmm0
   0x0000000000417e92 <+450>:   mov    -0x220(%rbp),%rcx
   0x0000000000417e99 <+457>:   mov    %rbx,%rsi
   0x0000000000417e9c <+460>:   mov    %r13,%rdi
   0x0000000000417e9f <+463>:   movaps %xmm7,-0x2b0(%rbp)
   0x0000000000417ea6 <+470>:   movdqa -0x390(%rbp),%xmm7
   0x0000000000417eae <+478>:   shr    $0x3f,%rax
   0x0000000000417eb2 <+482>:   movdqa -0x340(%rbp),%xmm1
   0x0000000000417eba <+490>:   movaps %xmm0,-0x260(%rbp)
   0x0000000000417ec1 <+497>:   movdqa -0x330(%rbp),%xmm2
   0x0000000000417ec9 <+505>:   movdqa -0x320(%rbp),%xmm3
   0x0000000000417ed1 <+513>:   sub    %rax,%rcx
   0x0000000000417ed4 <+516>:   movaps %xmm7,-0x2a0(%rbp)
   0x0000000000417edb <+523>:   movdqa -0x380(%rbp),%xmm7
   0x0000000000417ee3 <+531>:   movdqa -0x310(%rbp),%xmm4
   0x0000000000417eeb <+539>:   movdqa -0x300(%rbp),%xmm5
   0x0000000000417ef3 <+547>:   movdqa -0x2f0(%rbp),%xmm6
   0x0000000000417efb <+555>:   mov    %rcx,0x18(%r13)
   0x0000000000417eff <+559>:   movaps %xmm7,-0x290(%rbp)
   0x0000000000417f06 <+566>:   movdqa -0x370(%rbp),%xmm7
   0x0000000000417f0e <+574>:   movdqa -0x2d0(%rbp),%xmm0
   0x0000000000417f16 <+582>:   movaps %xmm1,-0x250(%rbp)
   0x0000000000417f1d <+589>:   movaps %xmm7,-0x280(%rbp)
   0x0000000000417f24 <+596>:   movdqa -0x360(%rbp),%xmm7
   0x0000000000417f2c <+604>:   movaps %xmm2,-0x240(%rbp)
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000417f33 <+611>:   movaps %xmm7,-0x270(%rbp)
   0x0000000000417f3a <+618>:   movdqa -0x2e0(%rbp),%xmm7
   0x0000000000417f42 <+626>:   movaps %xmm3,-0x230(%rbp)
   0x0000000000417f49 <+633>:   movaps %xmm4,-0x220(%rbp)
   0x0000000000417f50 <+640>:   movaps %xmm5,-0x210(%rbp)
   0x0000000000417f57 <+647>:   movaps %xmm6,-0x200(%rbp)
   0x0000000000417f5e <+654>:   movaps %xmm7,-0x1f0(%rbp)
   0x0000000000417f65 <+661>:   movaps %xmm0,-0x1e0(%rbp)
   0x0000000000417f6c <+668>:   movdqa -0x2c0(%rbp),%xmm1
   0x0000000000417f74 <+676>:   movaps %xmm1,-0x1d0(%rbp)
   0x0000000000417f7b <+683>:   callq  0x4176b0 <_Unwind_RaiseException_Phase2>
   0x0000000000417f80 <+688>:   cmp    $0x7,%eax
   0x0000000000417f83 <+691>:   jne    0x417e40 <_Unwind_RaiseException+368>
   0x0000000000417f89 <+697>:   mov    %rbx,%rsi
   0x0000000000417f8c <+700>:   mov    %r14,%rdi
   0x0000000000417f8f <+703>:   callq  0x417890 <uw_install_context_1>
   0x0000000000417f94 <+708>:   mov    -0x218(%rbp),%r8
   0x0000000000417f9b <+715>:   mov    -0x220(%rbp),%rdi
   0x0000000000417fa2 <+722>:   mov    %r8,%rsi
   0x0000000000417fa5 <+725>:   callq  0x417cc0 <_Unwind_DebugHook>
   0x0000000000417faa <+730>:   mov    %rax,%rcx
```

```
   0x0000000000417fad <+733>:   mov    %r8,0x8(%rbp,%rax,1)
   0x0000000000417fb2 <+738>:   mov    -0x38(%rbp),%rax
   0x0000000000417fb6 <+742>:   lea    0x8(%rbp,%rcx,1),%rcx
   0x0000000000417fbb <+747>:   mov    -0x30(%rbp),%rdx
   0x0000000000417fbf <+751>:   mov    -0x28(%rbp),%rbx
   0x0000000000417fc3 <+755>:   mov    -0x20(%rbp),%r12
   0x0000000000417fc7 <+759>:   mov    -0x18(%rbp),%r13
   0x0000000000417fcb <+763>:   mov    -0x10(%rbp),%r14
   0x0000000000417fcf <+767>:   mov    -0x8(%rbp),%r15
   0x0000000000417fd3 <+771>:   mov    0x0(%rbp),%rbp
   0x0000000000417fd7 <+775>:   mov    %rcx,%rsp
   0x0000000000417fda <+778>:   retq
End of assembler dump.
```

Exercise A8 (A64, GDB)

Goal: Learn how to identify runtime exceptions, past execution residue and stack traces, identify handled exceptions.

Patterns: C++ Exception; Execution Residue (User Space); Past Stack Trace; Coincidental Symbolic Information; Handled Exception (User Space).

1. Load *core.25889* dump file and *App8* executable from the A64/App8 directory:

```
~/ALCDA2/A64/App8$ gdb -c core.25889 -se App8
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App8...
(No debugging symbols found in App8)

warning: Can't open file /home/opc/ALCDA2/App8/App8 during file-backed mapping note processing
[New LWP 25890]
[New LWP 25892]
[New LWP 25889]
[New LWP 25891]
[New LWP 25894]
[New LWP 25893]
Core was generated by `./App8'.
Program terminated with signal SIGABRT, Aborted.
#0  0x0000000000420cfc in raise ()
[Current thread is 1 (LWP 25890)]
```

2. Set logging to a file in case of lengthy output from some commands:

```
(gdb) set logging file App8.log

(gdb) set logging enabled on
Copying output to App8.log.
Copying debug output to App8.log.

(gdb) set style enabled off
```

3. List all thread stack traces:

```
(gdb) thread apply all bt

Thread 6 (LWP 25893):
#0  0x0000000000420174 in nanosleep ()
#1  0x0000000000438e34 in sleep ()
#2  0x0000000000403520 in procNE() ()
#3  0x0000000000403600 in bar_four() ()
#4  0x0000000000403614 in foo_four() ()
#5  0x000000000040362c in thread_four(void*) ()
#6  0x00000000004183f4 in start_thread ()
#7  0x000000000043dd20 in thread_start ()

Thread 5 (LWP 25894):
#0  0x0000000000420174 in nanosleep ()
#1  0x0000000000438e34 in sleep ()
#2  0x0000000000403520 in procNE() ()
#3  0x0000000000403644 in bar_five() ()
#4  0x0000000000403658 in foo_five() ()
#5  0x0000000000403670 in thread_five(void*) ()
#6  0x00000000004183f4 in start_thread ()
#7  0x000000000043dd20 in thread_start ()

Thread 4 (LWP 25891):
#0  0x0000000000420174 in nanosleep ()
#1  0x0000000000438e34 in sleep ()
#2  0x0000000000403520 in procNE() ()
#3  0x0000000000403578 in bar_two() ()
#4  0x000000000040358c in foo_two() ()
#5  0x00000000004035a4 in thread_two(void*) ()
#6  0x00000000004183f4 in start_thread ()
#7  0x000000000043dd20 in thread_start ()

Thread 3 (LWP 25889):
#0  0x0000000000420174 in nanosleep ()
#1  0x0000000000438e34 in sleep ()
#2  0x000000000040370c in main ()

Thread 2 (LWP 25892):
#0  0x0000000000420174 in nanosleep ()
#1  0x0000000000438e34 in sleep ()
#2  0x00000000004034cc in procH() ()
#3  0x00000000004035bc in bar_three() ()
#4  0x00000000004035d0 in foo_three() ()
#5  0x00000000004035e8 in thread_three(void*) ()
#6  0x00000000004183f4 in start_thread ()
#7  0x000000000043dd20 in thread_start ()

Thread 1 (LWP 25890):
#0  0x0000000000420cfc in raise ()
#1  0x0000000000422d38 in abort ()
--Type <RET> for more, q to quit, c to continue without paging--
#2  0x00000000004086f0 in __gnu_cxx::__verbose_terminate_handler() ()
#3  0x0000000000404c0c in __cxxabiv1::__terminate(void (*)()) ()
#4  0x0000000000404c30 in std::terminate() ()
#5  0x0000000000404d88 in __cxa_throw ()
#6  0x0000000000403424 in procB() ()
#7  0x0000000000403490 in procA() ()
#8  0x0000000000403504 in procNH() ()
```

```
#9  0x0000000000403534 in bar_one() ()
#10 0x0000000000403548 in foo_one() ()
#11 0x0000000000403560 in thread_one(void*) ()
#12 0x00000000004183f4 in start_thread ()
#13 0x000000000043dd20 in thread_start ()
```

Note: We have C++ exception processing in thread #1.

4. Go to thread #4, identify the execution residue of *work* functions, check their correctness, and reconstruct the past stack trace:

```
(gdb) thread 4
[Switching to thread 4 (LWP 25891)]
#0  0x0000000000420174 in nanosleep ()

(gdb) bt
#0  0x0000000000420174 in nanosleep ()
#1  0x0000000000438e34 in sleep ()
#2  0x0000000000403520 in procNE() ()
#3  0x0000000000403578 in bar_two() ()
#4  0x000000000040358c in foo_two() ()
#5  0x00000000004035a4 in thread_two(void*) ()
#6  0x00000000004183f4 in start_thread ()
#7  0x000000000043dd20 in thread_start ()

(gdb) x/512a $sp-2000
0xfffe79bbde00: 0x0     0x0
0xfffe79bbde10: 0x0     0x0
0xfffe79bbde20: 0x0     0x0
0xfffe79bbde30: 0x0     0x0
0xfffe79bbde40: 0x0     0x0
0xfffe79bbde50: 0x0     0x0
0xfffe79bbde60: 0x0     0x0
0xfffe79bbde70: 0x0     0x0
0xfffe79bbde80: 0x0     0x0
0xfffe79bbde90: 0x0     0x0
0xfffe79bbdea0: 0x0     0x0
0xfffe79bbdeb0: 0x0     0x0
0xfffe79bbdec0: 0x0     0x0
0xfffe79bbded0: 0x0     0x0
0xfffe79bbdee0: 0x0     0x0
0xfffe79bbdef0: 0x0     0x0
0xfffe79bbdf00: 0x0     0x0
0xfffe79bbdf10: 0x0     0x0
0xfffe79bbdf20: 0x0     0x0
0xfffe79bbdf30: 0x0     0x0
0xfffe79bbdf40: 0x0     0x0
0xfffe79bbdf50: 0x0     0x0
0xfffe79bbdf60: 0x0     0x0
0xfffe79bbdf70: 0x0     0x0
0xfffe79bbdf80: 0x0     0x0
0xfffe79bbdf90: 0x0     0x0
0xfffe79bbdfa0: 0x0     0x0
0xfffe79bbdfb0: 0x0     0x0
0xfffe79bbdfc0: 0x0     0x0
0xfffe79bbdfd0: 0x0     0x0
0xfffe79bbdfe0: 0x0     0x0
0xfffe79bbdff0: 0x0     0x0
0xfffe79bbe000: 0x0     0x0
```

```
0xfffe79bbe010: 0x0     0x0
0xfffe79bbe020: 0x0     0x0
0xfffe79bbe030: 0x0     0x0
0xfffe79bbe040: 0x0     0x0
0xfffe79bbe050: 0x0     0x0
0xfffe79bbe060: 0x0     0x0
0xfffe79bbe070: 0x0     0x0
0xfffe79bbe080: 0x0     0x0
0xfffe79bbe090: 0x0     0x0
0xfffe79bbe0a0: 0x0     0x0
0xfffe79bbe0b0: 0x0     0x0
0xfffe79bbe0c0: 0x0     0x0
0xfffe79bbe0d0: 0x0     0x0
0xfffe79bbe0e0: 0x0     0x0
0xfffe79bbe0f0: 0x0     0x0
0xfffe79bbe100: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe79bbe110: 0x0     0x0
0xfffe79bbe120: 0x0     0x0
0xfffe79bbe130: 0x0     0x0
0xfffe79bbe140: 0x0     0x0
0xfffe79bbe150: 0x0     0x0
0xfffe79bbe160: 0x0     0x0
0xfffe79bbe170: 0x0     0x0
0xfffe79bbe180: 0x0     0x0
0xfffe79bbe190: 0x0     0x0
0xfffe79bbe1a0: 0x0     0x0
0xfffe79bbe1b0: 0x0     0x0
0xfffe79bbe1c0: 0x0     0x0
0xfffe79bbe1d0: 0x0     0x0
0xfffe79bbe1e0: 0x0     0x0
0xfffe79bbe1f0: 0x0     0x0
0xfffe79bbe200: 0x0     0x0
0xfffe79bbe210: 0x0     0x0
0xfffe79bbe220: 0x0     0x0
0xfffe79bbe230: 0x0     0x0
0xfffe79bbe240: 0x0     0x0
0xfffe79bbe250: 0x0     0x0
0xfffe79bbe260: 0x0     0x0
0xfffe79bbe270: 0x0     0x0
0xfffe79bbe280: 0x0     0x0
0xfffe79bbe290: 0x0     0x0
0xfffe79bbe2a0: 0x0     0x0
0xfffe79bbe2b0: 0x0     0x0
0xfffe79bbe2c0: 0x0     0x0
0xfffe79bbe2d0: 0x0     0x0
0xfffe79bbe2e0: 0x0     0x0
0xfffe79bbe2f0: 0x0     0x0
0xfffe79bbe300: 0x0     0x0
0xfffe79bbe310: 0x0     0x0
0xfffe79bbe320: 0x0     0x0
0xfffe79bbe330: 0x0     0x0
0xfffe79bbe340: 0x0     0x0
0xfffe79bbe350: 0xfffe79bbe360  0x403304 <_Z6work_7v+12>
0xfffe79bbe360: 0xfffe79bbe370  0x403318 <_Z6work_6v+12>
0xfffe79bbe370: 0xfffe79bbe380  0x40332c <_Z6work_5v+12>
0xfffe79bbe380: 0xfffe79bbe390  0x403340 <_Z6work_4v+12>
0xfffe79bbe390: 0xfffe79bbe3a0  0x403354 <_Z6work_3v+12>
0xfffe79bbe3a0: 0xfffe79bbe3b0  0x403368 <_Z6work_2v+12>
0xfffe79bbe3b0: 0xfffe79bbe3c0  0x40337c <_Z6work_1v+12>
```

```
0xfffe79bbe3c0: 0xfffe79bbe3d0  0x403394 <_Z4workv+16>
0xfffe79bbe3d0: 0xfffe79bbe7e0  0x40347c <_Z6procNBv+12>
0xfffe79bbe3e0: 0x0     0x0
0xfffe79bbe3f0: 0x0     0x0
0xfffe79bbe400: 0x0     0x0
0xfffe79bbe410: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe79bbe420: 0x0     0x0
0xfffe79bbe430: 0x0     0x0
0xfffe79bbe440: 0x0     0x0
0xfffe79bbe450: 0x0     0x0
0xfffe79bbe460: 0x0     0x0
0xfffe79bbe470: 0x0     0x0
0xfffe79bbe480: 0x0     0x0
0xfffe79bbe490: 0x0     0x0
0xfffe79bbe4a0: 0xfffe79bbe610  0x438e08 <sleep+228>
0xfffe79bbe4b0: 0xffffffff      0x10000
0xfffe79bbe4c0: 0x0     0x0
0xfffe79bbe4d0: 0x0     0x0
0xfffe79bbe4e0: 0x0     0x0
0xfffe79bbe4f0: 0x0     0x0
0xfffe79bbe500: 0x0     0x0
0xfffe79bbe510: 0x0     0x0
0xfffe79bbe520: 0x0     0x0
0xfffe79bbe530: 0x0     0x0
0xfffe79bbe540: 0x0     0x0
0xfffe79bbe550: 0x0     0x0
0xfffe79bbe560: 0x0     0x0
0xfffe79bbe570: 0xfffe79bbe610  0x438e28 <sleep+260>
0xfffe79bbe580: 0xffffffff      0x10000
0xfffe79bbe590: 0x0     0x0
0xfffe79bbe5a0: 0x0     0x0
0xfffe79bbe5b0: 0xfffe79bbe610  0x420168 <nanosleep+24>
0xfffe79bbe5c0: 0x0     0x0
0xfffe79bbe5d0: 0x438e34 <sleep+272>    0xfffe79bbe650
0xfffe79bbe5e0: 0xfffe79bbe650  0x0
0xfffe79bbe5f0: 0x0     0x0
0xfffe79bbe600: 0x0     0x0
0xfffe79bbe610: 0xfffe79bbe800  0x403520 <_Z6procNEv+20>
0xfffe79bbe620: 0xfffe79bbf070  0x0
0xfffe79bbe630: 0x4e0000        0x403594 <_Z10thread_twoPv>
0xfffe79bbe640: 0x0     0x0
0xfffe79bbe650: 0xfffffff4      0x3b985e11
0xfffe79bbe660: 0x0     0x0
0xfffe79bbe670: 0x0     0x0
0xfffe79bbe680: 0x0     0x0
0xfffe79bbe690: 0x0     0x0
0xfffe79bbe6a0: 0x0     0x0
0xfffe79bbe6b0: 0x0     0x0
0xfffe79bbe6c0: 0x0     0x0
0xfffe79bbe6d0: 0x0     0x0
0xfffe79bbe6e0: 0x10000 0x0
0xfffe79bbe6f0: 0x0     0x0
0xfffe79bbe700: 0x0     0x0
0xfffe79bbe710: 0x0     0x0
0xfffe79bbe720: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe79bbe730: 0x0     0x0
0xfffe79bbe740: 0x0     0x0
0xfffe79bbe750: 0x0     0x0
```

```
0xfffe79bbe760: 0x0     0x0
0xfffe79bbe770: 0x0     0x0
0xfffe79bbe780: 0x0     0x0
0xfffe79bbe790: 0x0     0x0
0xfffe79bbe7a0: 0x0     0x0
0xfffe79bbe7b0: 0x0     0x0
0xfffe79bbe7c0: 0x0     0x0
0xfffe79bbe7d0: 0x0     0x0
0xfffe79bbe7e0: 0x0     0x0
0xfffe79bbe7f0: 0x0     0x403518 <_Z6procNEv+12>
0xfffe79bbe800: 0xfffe79bbe810  0x403578 <_Z7bar_twov+12>
0xfffe79bbe810: 0xfffe79bbe820  0x40358c <_Z7foo_twov+12>
0xfffe79bbe820: 0xfffe79bbe830  0x4035a4 <_Z10thread_twoPv+16>
0xfffe79bbe830: 0xfffe79bbe850  0x4183f4 <start_thread+180>
0xfffe79bbe840: 0xfffe79bbf070  0x0
0xfffe79bbe850: 0x0     0x43dd20 <thread_start+48>
0xfffe79bbe860: 0xfffe79bbf070  0x4f9540 <__default_pthread_attr>
0xfffe79bbe870: 0x4e0000        0x0
0xfffe79bbe880: 0xfffe79bbf48c  0xfffe79bbf070
0xfffe79bbe890: 0x0     0x0
0xfffe79bbe8a0: 0xfffe79bbf070  0x4f9540 <__default_pthread_attr>
0xfffe79bbe8b0: 0x4e0000        0x403594 <_Z10thread_twoPv>
0xfffe79bbe8c0: 0x0     0xfffe79bbf760
0xfffe79bbe8d0: 0x32b706f0      0x4f9540 <__default_pthread_attr>
0xfffe79bbe8e0: 0x10000 0x810000
0xfffe79bbe8f0: 0xfffe79bbe850  0x1be0e4ebeeaf72fa
0xfffe79bbe900: 0x0     0x1be01b159755196a
0xfffe79bbe910: 0x0     0x0
0xfffe79bbe920: 0x0     0x0
0xfffe79bbe930: 0x0     0x0
0xfffe79bbe940: 0x0     0x0
0xfffe79bbe950: 0x0     0x0
0xfffe79bbe960: 0x0     0x0
0xfffe79bbe970: 0x0     0x0
0xfffe79bbe980: 0x0     0x0
0xfffe79bbe990: 0x0     0x0
0xfffe79bbe9a0: 0x0     0x0
0xfffe79bbe9b0: 0x0     0x0
0xfffe79bbe9c0: 0x0     0x0
0xfffe79bbe9d0: 0x0     0x0
0xfffe79bbe9e0: 0x0     0x0
0xfffe79bbe9f0: 0x0     0x0
0xfffe79bbea00: 0x0     0x0
0xfffe79bbea10: 0x0     0x0
0xfffe79bbea20: 0x0     0x0
0xfffe79bbea30: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe79bbea40: 0x0     0x0
0xfffe79bbea50: 0x0     0x0
0xfffe79bbea60: 0x0     0x0
0xfffe79bbea70: 0x0     0x0
0xfffe79bbea80: 0x0     0x0
0xfffe79bbea90: 0x0     0x0
0xfffe79bbeaa0: 0x0     0x0
0xfffe79bbeab0: 0x0     0x0
0xfffe79bbeac0: 0x0     0x0
0xfffe79bbead0: 0x0     0x0
0xfffe79bbeae0: 0x0     0x0
0xfffe79bbeaf0: 0x0     0x0
0xfffe79bbeb00: 0x0     0x0
```

```
0xfffe79bbeb10: 0x0     0x0
0xfffe79bbeb20: 0x0     0x0
0xfffe79bbeb30: 0x0     0x0
0xfffe79bbeb40: 0x0     0x0
0xfffe79bbeb50: 0x0     0x0
0xfffe79bbeb60: 0x0     0x0
0xfffe79bbeb70: 0x0     0x0
0xfffe79bbeb80: 0x0     0x0
0xfffe79bbeb90: 0x0     0x0
0xfffe79bbeba0: 0x0     0x0
0xfffe79bbebb0: 0x0     0x0
0xfffe79bbebc0: 0x0     0x0
0xfffe79bbebd0: 0x0     0x0
0xfffe79bbebe0: 0x0     0x0
0xfffe79bbebf0: 0x0     0x0
0xfffe79bbec00: 0x0     0x0
0xfffe79bbec10: 0x0     0x0
0xfffe79bbec20: 0x0     0x0
0xfffe79bbec30: 0x0     0x0
0xfffe79bbec40: 0x0     0x0
0xfffe79bbec50: 0x0     0x0
0xfffe79bbec60: 0x0     0x0
0xfffe79bbec70: 0x0     0x0
0xfffe79bbec80: 0x0     0x0
0xfffe79bbec90: 0x0     0x0
0xfffe79bbeca0: 0x0     0x0
0xfffe79bbecb0: 0x0     0x0
0xfffe79bbecc0: 0x0     0x0
0xfffe79bbecd0: 0x0     0x0
0xfffe79bbece0: 0x0     0x0
0xfffe79bbecf0: 0x0     0x0
0xfffe79bbed00: 0x0     0x0
0xfffe79bbed10: 0x0     0x0
0xfffe79bbed20: 0x0     0x0
0xfffe79bbed30: 0x0     0x0
0xfffe79bbed40: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe79bbed50: 0x0     0x0
0xfffe79bbed60: 0x0     0x0
0xfffe79bbed70: 0x0     0x0
0xfffe79bbed80: 0x0     0x0
0xfffe79bbed90: 0x0     0x0
0xfffe79bbeda0: 0x0     0x0
0xfffe79bbedb0: 0x0     0x0
0xfffe79bbedc0: 0x0     0x0
0xfffe79bbedd0: 0x0     0x0
0xfffe79bbede0: 0x0     0x0
0xfffe79bbedf0: 0x0     0x0

(gdb) disassemble 0x403354
Dump of assembler code for function _Z6work_3v:
   0x0000000000403348 <+0>:     stp     x29, x30, [sp, #-16]!
   0x000000000040334c <+4>:     mov     x29, sp
   0x0000000000403350 <+8>:     bl      0x403334 <_Z6work_4v>
   0x0000000000403354 <+12>:    ldp     x29, x30, [sp], #16
   0x0000000000403358 <+16>:    ret
End of assembler dump.
```

Note: Since the saved X29 register value points to the next line, we can easily reconstruct the fragment of the past stack trace:

```
0xfffe79bbe350: 0xfffe79bbe360  0x403304 <_Z6work_7v+12>
0xfffe79bbe360: 0xfffe79bbe370  0x403318 <_Z6work_6v+12>
0xfffe79bbe370: 0xfffe79bbe380  0x40332c <_Z6work_5v+12>
0xfffe79bbe380: 0xfffe79bbe390  0x403340 <_Z6work_4v+12>
0xfffe79bbe390: 0xfffe79bbe3a0  0x403354 <_Z6work_3v+12>
0xfffe79bbe3a0: 0xfffe79bbe3b0  0x403368 <_Z6work_2v+12>
0xfffe79bbe3b0: 0xfffe79bbe3c0  0x40337c <_Z6work_1v+12>
0xfffe79bbe3c0: 0xfffe79bbe3d0  0x403394 <_Z4workv+16>
0xfffe79bbe3d0: 0xfffe79bbe7e0  0x40347c <_Z6procNBv+12>
```

5. Go to thread #2, identify the handled exception processing code, and check its validity:

```
(gdb) thread 2
[Switching to thread 2 (LWP 25892)]
#0  0x0000000000420174 in nanosleep ()

(gdb) bt
#0  0x0000000000420174 in nanosleep ()
#1  0x0000000000438e34 in sleep ()
#2  0x00000000004034cc in procH() ()
#3  0x00000000004035bc in bar_three() ()
#4  0x00000000004035d0 in foo_three() ()
#5  0x00000000004035e8 in thread_three(void*) ()
#6  0x00000000004183f4 in start_thread ()
#7  0x000000000043dd20 in thread_start ()

(gdb) x/1024a $sp-8000
0xfffe793ac680: 0x0     0x0
0xfffe793ac690: 0x0     0x0
0xfffe793ac6a0: 0x0     0x0
0xfffe793ac6b0: 0x0     0x0
0xfffe793ac6c0: 0x0     0x0
0xfffe793ac6d0: 0x0     0x0
0xfffe793ac6e0: 0x0     0x0
0xfffe793ac6f0: 0x0     0x0
0xfffe793ac700: 0x0     0x0
0xfffe793ac710: 0x0     0x0
0xfffe793ac720: 0x0     0x0
0xfffe793ac730: 0x0     0x0
0xfffe793ac740: 0x0     0x0
0xfffe793ac750: 0x0     0x0
0xfffe793ac760: 0x0     0x0
0xfffe793ac770: 0x0     0x0
0xfffe793ac780: 0x0     0x0
0xfffe793ac790: 0x0     0x0
0xfffe793ac7a0: 0x0     0x0
0xfffe793ac7b0: 0x0     0x0
0xfffe793ac7c0: 0x0     0x0
0xfffe793ac7d0: 0x0     0x0
0xfffe793ac7e0: 0x0     0x0
0xfffe793ac7f0: 0x0     0x0
0xfffe793ac800: 0x0     0x0
0xfffe793ac810: 0x0     0x0
0xfffe793ac820: 0x0     0x0
0xfffe793ac830: 0x0     0x0
0xfffe793ac840: 0x0     0x0
0xfffe793ac850: 0x0     0x0
0xfffe793ac860: 0x0     0x0
```

```
0xfffe793ac870: 0x0     0x0
0xfffe793ac880: 0x0     0x0
0xfffe793ac890: 0x0     0x0
0xfffe793ac8a0: 0x0     0x0
0xfffe793ac8b0: 0x0     0x0
0xfffe793ac8c0: 0x0     0x0
0xfffe793ac8d0: 0x0     0x0
0xfffe793ac8e0: 0x0     0x0
0xfffe793ac8f0: 0x0     0x0
0xfffe793ac900: 0x0     0x0
0xfffe793ac910: 0x0     0x0
0xfffe793ac920: 0x0     0x0
0xfffe793ac930: 0x0     0x0
0xfffe793ac940: 0x0     0x0
0xfffe793ac950: 0x0     0x0
0xfffe793ac960: 0x0     0x0
0xfffe793ac970: 0x0     0x0
0xfffe793ac980: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe793ac990: 0x0     0x0
0xfffe793ac9a0: 0x0     0x0
0xfffe793ac9b0: 0x0     0x0
0xfffe793ac9c0: 0x0     0x0
0xfffe793ac9d0: 0x0     0x0
0xfffe793ac9e0: 0x0     0x0
0xfffe793ac9f0: 0x0     0x0
0xfffe793aca00: 0x0     0x0
0xfffe793aca10: 0x0     0x0
0xfffe793aca20: 0x0     0x0
0xfffe793aca30: 0x0     0x0
0xfffe793aca40: 0x0     0x0
0xfffe793aca50: 0x0     0x0
0xfffe793aca60: 0x0     0x0
0xfffe793aca70: 0x0     0x0
0xfffe793aca80: 0x0     0x0
0xfffe793aca90: 0x0     0x0
0xfffe793acaa0: 0x0     0x0
0xfffe793acab0: 0x0     0x0
0xfffe793acac0: 0x0     0x0
0xfffe793acad0: 0x0     0x0
0xfffe793acae0: 0x0     0x0
0xfffe793acaf0: 0x0     0x0
0xfffe793acb00: 0x0     0x0
0xfffe793acb10: 0x0     0x0
0xfffe793acb20: 0x0     0x0
0xfffe793acb30: 0x0     0x0
0xfffe793acb40: 0x0     0x0
0xfffe793acb50: 0x0     0x0
0xfffe793acb60: 0x0     0x0
0xfffe793acb70: 0x0     0x0
0xfffe793acb80: 0x0     0x0
0xfffe793acb90: 0x0     0x0
0xfffe793acba0: 0x0     0x0
0xfffe793acbb0: 0x0     0x0
0xfffe793acbc0: 0x0     0x0
0xfffe793acbd0: 0x0     0x0
0xfffe793acbe0: 0x0     0x0
0xfffe793acbf0: 0x0     0x0
0xfffe793acc00: 0x0     0x0
0xfffe793acc10: 0x0     0x0
```

```
0xfffe793acc20: 0x0     0x0
0xfffe793acc30: 0x0     0x0
0xfffe793acc40: 0x0     0x0
0xfffe793acc50: 0x0     0x0
0xfffe793acc60: 0x0     0x0
0xfffe793acc70: 0x0     0x0
0xfffe793acc80: 0x0     0x0
0xfffe793acc90: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe793acca0: 0x0     0x0
0xfffe793accb0: 0x0     0x0
0xfffe793accc0: 0x0     0x0
0xfffe793accd0: 0x0     0x0
0xfffe793acce0: 0x0     0x0
0xfffe793accf0: 0x0     0x0
0xfffe793acd00: 0x0     0x0
0xfffe793acd10: 0x0     0x0
0xfffe793acd20: 0x0     0x0
0xfffe793acd30: 0x0     0x0
0xfffe793acd40: 0x0     0x0
0xfffe793acd50: 0x0     0x0
0xfffe793acd60: 0x0     0x0
0xfffe793acd70: 0x0     0x0
0xfffe793acd80: 0xfffe793ad1c0  0x4144a8 <uw_update_context+24>
0xfffe793acd90: 0xfffe793add30  0xfffe793ad220
0xfffe793acda0: 0x0     0xfffe74000b80
0xfffe793acdb0: 0x4     0xfffe793af760
0xfffe793acdc0: 0x32b706f0      0x4f9540 <__default_pthread_attr>
0xfffe793acdd0: 0x10000 0x810000
0xfffe793acde0: 0x4f3000 <_ZL16emergency_buffer+65008>  0xfffe793ad220
0xfffe793acdf0: 0x4f3000 <_ZL16emergency_buffer+65008>  0xfffe793ae7e0
0xfffe793ace00: 0xfffe793ad8b0  0xfffe793ad8b8
0xfffe793ace10: 0xfffe793ad8c0  0xfffe793ad8c8
0xfffe793ace20: 0x0     0x0
0xfffe793ace30: 0x0     0x0
0xfffe793ace40: 0x0     0x0
0xfffe793ace50: 0x0     0x0
0xfffe793ace60: 0x0     0x0
0xfffe793ace70: 0x0     0x0
0xfffe793ace80: 0x0     0x0
0xfffe793ace90: 0x0     0xfffe793ae7b0
0xfffe793acea0: 0xfffe793ae7b8  0xfffe793ae7c0
0xfffe793aceb0: 0xfffe793ad8e8  0xfffe793ad8f0
0xfffe793acec0: 0xfffe793ad8f8  0xfffe793ad900
0xfffe793aced0: 0xfffe793ad908  0xfffe793ad910
0xfffe793acee0: 0xfffe793ad918  0xfffe793ae7a0
0xfffe793acef0: 0xfffe793ae7a8  0xfffe793acdf8
0xfffe793acf00: 0x0     0x0
0xfffe793acf10: 0x0     0x0
0xfffe793acf20: 0x0     0x0
0xfffe793acf30: 0x0     0x0
0xfffe793acf40: 0x0     0x0
0xfffe793acf50: 0x0     0x0
0xfffe793acf60: 0x0     0x0
0xfffe793acf70: 0x0     0x0
0xfffe793acf80: 0x0     0x0
0xfffe793acf90: 0x0     0x0
0xfffe793acfa0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe793acfb0: 0x0     0x0
```

```
0xfffe793acfc0: 0x0     0x0
0xfffe793acfd0: 0x0     0x0
0xfffe793acfe0: 0x0     0x0
0xfffe793acff0: 0xfffe793ad030  0x416218 <search_object+516>
0xfffe793ad000: 0x4b2fb8        0x2d
0xfffe793ad010: 0x0     0x0
0xfffe793ad020: 0xfffe793ad030  0x4161a0 <search_object+396>
0xfffe793ad030: 0xfffe793ad0d0  0x416b28 <_Unwind_Find_FDE+372>
0xfffe793ad040: 0x4e15a8 <object.6205>  0xfffe793ad220
0xfffe793ad050: 0x4034bb <_Z5procHv+15> 0xfffe74000b80
0xfffe793ad060: 0x4f3000 <_ZL16emergency_buffer+65008>  0xfffe793ae058
0xfffe793ad070: 0x4df000 <_ZTIh+8>      0x4f9540 <__default_pthread_attr>
0xfffe793ad080: 0x10000 0x810000
0xfffe793ad090: 0xfffe793ad0d0  0x416ae4 <_Unwind_Find_FDE+304>
0xfffe793ad0a0: 0x4e15a8 <object.6205>  0x1b
0xfffe793ad0b0: 0xfffe793add30  0x4034ac <_Z5procHv>
0xfffe793ad0c0: 0xfffe793ad0d0  0x416ad0 <_Unwind_Find_FDE+284>
0xfffe793ad0d0: 0xfffe793ad150  0x4136cc <uw_frame_state_for+1484>
0xfffe793ad0e0: 0xfffe793ad0f0  0x404754 <__gxx_personality_v0+240>
0xfffe793ad0f0: 0xfffe793ad1e0  0x4145ac <_Unwind_RaiseException_Phase2+112>
0xfffe793ad100: 0xfffe793add30  0xfffe793ad220
0xfffe793ad110: 0x4     0xfffe74000b80
0xfffe793ad120: 0x4     0xfffe793af760
0xfffe793ad130: 0x32b706f0      0x4f9540 <__default_pthread_attr>
0xfffe793ad140: 0x10000 0x810000
0xfffe793ad150: 0xfffe793ad1e0  0x414570 <_Unwind_RaiseException_Phase2+52>
0xfffe793ad160: 0xfffe793add30  0x4c57d8
0xfffe793ad170: 0x0     0xfffe74000b80
0xfffe793ad180: 0x4     0x0
0xfffe793ad190: 0x32b706f0      0x4f9540 <__default_pthread_attr>
0xfffe793ad1a0: 0x10000 0x810000
0xfffe793ad1b0: 0x0     0x0
0xfffe793ad1c0: 0xfffe793ad1e0  0x4145cc <_Unwind_RaiseException_Phase2+144>
0xfffe793ad1d0: 0xffffffffffffffff8      0x76a28b436af36f00
0xfffe793ad1e0: 0xfffe793ad8a0  0x414bf4 <_Unwind_RaiseException+324>
0xfffe793ad1f0: 0xfffe793add30  0xfffe793ae0f0
0xfffe793ad200: 0xfffe74000b80  0xfffe793ad970
0xfffe793ad210: 0x0     0xfffe793ae770
0xfffe793ad220: 0x0     0x0
0xfffe793ad230: 0x0     0x0
0xfffe793ad240: 0x0     0x0
0xfffe793ad250: 0x0     0x0
0xfffe793ad260: 0x0     0x0
0xfffe793ad270: 0x0     0x0
0xfffe793ad280: 0x0     0x0
0xfffe793ad290: 0x0     0x0
0xfffe793ad2a0: 0x0     0x0
0xfffe793ad2b0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe793ad2c0: 0x0     0x0
0xfffe793ad2d0: 0x0     0x0
0xfffe793ad2e0: 0x0     0x0
0xfffe793ad2f0: 0x0     0x0
0xfffe793ad300: 0x0     0x0
0xfffe793ad310: 0x0     0x0
0xfffe793ad320: 0x0     0x0
0xfffe793ad330: 0x0     0x0
0xfffe793ad340: 0x0     0x0
0xfffe793ad350: 0xfffffffffffffff0      0x1
0xfffe793ad360: 0x0     0x0
```

```
0xfffe793ad370: 0x0     0x0
0xfffe793ad380: 0x0     0x0
0xfffe793ad390: 0x0     0x0
0xfffe793ad3a0: 0x0     0x0
0xfffe793ad3b0: 0x0     0x0
0xfffe793ad3c0: 0x0     0x0
0xfffe793ad3d0: 0x0     0x0
0xfffe793ad3e0: 0x0     0x0
0xfffe793ad3f0: 0xfffffffffffffe0       0x1
0xfffe793ad400: 0xfffffffffffffe8       0x1
0xfffe793ad410: 0x0     0x0
0xfffe793ad420: 0x0     0x0
0xfffe793ad430: 0x0     0x0
0xfffe793ad440: 0x0     0x0
0xfffe793ad450: 0x0     0x0
0xfffe793ad460: 0x0     0x0
0xfffe793ad470: 0x0     0x0
0xfffe793ad480: 0x0     0x0
0xfffe793ad490: 0x0     0x0
0xfffe793ad4a0: 0x0     0x0
0xfffe793ad4b0: 0x0     0x0
0xfffe793ad4c0: 0x0     0x0
0xfffe793ad4d0: 0x0     0x0
0xfffe793ad4e0: 0x0     0x0
0xfffe793ad4f0: 0x0     0x0
0xfffe793ad500: 0x0     0x0
0xfffe793ad510: 0x0     0x0
0xfffe793ad520: 0x0     0x0
0xfffe793ad530: 0x0     0x0
0xfffe793ad540: 0x0     0x0
0xfffe793ad550: 0x0     0x0
0xfffe793ad560: 0x0     0x0
0xfffe793ad570: 0x0     0x0
0xfffe793ad580: 0x0     0x0
0xfffe793ad590: 0x0     0x0
0xfffe793ad5a0: 0x0     0x0
0xfffe793ad5b0: 0x0     0x0
0xfffe793ad5c0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe793ad5d0: 0x0     0x0
0xfffe793ad5e0: 0x0     0x0
0xfffe793ad5f0: 0x0     0x0
0xfffe793ad600: 0x0     0x0
0xfffe793ad610: 0x0     0x0
0xfffe793ad620: 0x0     0x0
0xfffe793ad630: 0x0     0x0
0xfffe793ad640: 0x0     0x0
0xfffe793ad650: 0x0     0x0
0xfffe793ad660: 0x0     0x0
0xfffe793ad670: 0x0     0x0
0xfffe793ad680: 0x0     0x0
0xfffe793ad690: 0x0     0x0
0xfffe793ad6a0: 0x0     0x0
0xfffe793ad6b0: 0x0     0x0
0xfffe793ad6c0: 0x0     0x0
0xfffe793ad6d0: 0x0     0x0
0xfffe793ad6e0: 0x0     0x0
0xfffe793ad6f0: 0x0     0x0
0xfffe793ad700: 0x0     0x0
0xfffe793ad710: 0x0     0x0
```

```
0xfffe793ad720: 0x0     0x0
0xfffe793ad730: 0x0     0x0
0xfffe793ad740: 0x0     0x0
0xfffe793ad750: 0x0     0x0
0xfffe793ad760: 0x0     0x0
0xfffe793ad770: 0x0     0x0
0xfffe793ad780: 0x0     0x0
0xfffe793ad790: 0x0     0x0
0xfffe793ad7a0: 0x0     0x0
0xfffe793ad7b0: 0x0     0x0
0xfffe793ad7c0: 0x0     0x0
0xfffe793ad7d0: 0x0     0x0
0xfffe793ad7e0: 0x0     0x0
0xfffe793ad7f0: 0x0     0x0
0xfffe793ad800: 0x0     0x0
0xfffe793ad810: 0x0     0x0
0xfffe793ad820: 0x0     0x0
0xfffe793ad830: 0x0     0x0
0xfffe793ad840: 0x0     0x20
0xfffe793ad850: 0xfffe793ad8a0  0x414c08 <_Unwind_RaiseException+344>
0xfffe793ad860: 0xfffe793add30  0xfffe793ae0f0
0xfffe793ad870: 0xfffe74000b80  0xfffe793ad970
0xfffe793ad880: 0x0     0x1e
0xfffe793ad890: 0x11b1b 0xfffe793ae7f0
0xfffe793ad8a0: 0xfffe793ae7f0  0x4034c0 <_Z5procHv+20>
0xfffe793ad8b0: 0xfffe74000b80  0x1
0xfffe793ad8c0: 0x0     0x1
0xfffe793ad8d0: 0xfffe793af070  0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe793ad8e0: 0x4e0000        0x4035d8 <_Z12thread_threePv>
0xfffe793ad8f0: 0x0     0xfffe793af760
0xfffe793ad900: 0x32b706f0      0x4f9540 <__default_pthread_attr>
0xfffe793ad910: 0x10000 0x810000
0xfffe793ad920: 0x0     0x0
0xfffe793ad930: 0x0     0x0
0xfffe793ad940: 0x0     0x0
0xfffe793ad950: 0x0     0x0
0xfffe793ad960: 0x0     0x80
0xfffe793ad970: 0xfffe793ad8b0  0xfffe793ad8b8
0xfffe793ad980: 0xfffe793ad8c0  0xfffe793ad8c8
0xfffe793ad990: 0x0     0x0
0xfffe793ad9a0: 0x0     0x0
0xfffe793ad9b0: 0x0     0x0
0xfffe793ad9c0: 0x0     0x0
0xfffe793ad9d0: 0x0     0x0
0xfffe793ad9e0: 0x0     0x0
0xfffe793ad9f0: 0x0     0x0
0xfffe793ada00: 0x0     0xfffe793ad8d0
0xfffe793ada10: 0xfffe793ad8d8  0xfffe793ad8e0
0xfffe793ada20: 0xfffe793ad8e8  0xfffe793ad8f0
0xfffe793ada30: 0xfffe793ad8f8  0xfffe793ad900
0xfffe793ada40: 0xfffe793ad908  0xfffe793ad910
0xfffe793ada50: 0xfffe793ad918  0xfffe793ad8a0
0xfffe793ada60: 0xfffe793ad8a8  0x0
0xfffe793ada70: 0x0     0x0
0xfffe793ada80: 0x0     0x0
0xfffe793ada90: 0x0     0x0
0xfffe793adaa0: 0x0     0x0
0xfffe793adab0: 0x0     0x0
0xfffe793adac0: 0x0     0x0
```

```
0xfffe793adad0: 0x0     0x0
0xfffe793adae0: 0x0     0x0
0xfffe793adaf0: 0x0     0x0
0xfffe793adb00: 0x0     0x0
0xfffe793adb10: 0x0     0x0
0xfffe793adb20: 0x0     0x0
0xfffe793adb30: 0x0     0x0
0xfffe793adb40: 0x0     0x0
0xfffe793adb50: 0x0     0x0
0xfffe793adb60: 0x0     0x0
0xfffe793adb70: 0x0     0x0
0xfffe793adb80: 0x0     0x0
0xfffe793adb90: 0x0     0x0
0xfffe793adba0: 0x0     0x0
0xfffe793adbb0: 0xfffe793ad920  0xfffe793ad928
0xfffe793adbc0: 0xfffe793ad930  0xfffe793ad938
0xfffe793adbd0: 0xfffe793ad940  0xfffe793ad948
0xfffe793adbe0: 0xfffe793ad950  0xfffe793ad958
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe793adbf0: 0x0     0x0
0xfffe793adc00: 0x0     0x0
0xfffe793adc10: 0x0     0x0
0xfffe793adc20: 0x0     0x0
0xfffe793adc30: 0x0     0x0
0xfffe793adc40: 0x0     0x0
0xfffe793adc50: 0x0     0x0
0xfffe793adc60: 0x0     0x0
0xfffe793adc70: 0x0     0x0
0xfffe793adc80: 0xfffe793ae770  0x404d7c <__cxa_throw+144>
0xfffe793adc90: 0x0     0x0
0xfffe793adca0: 0x0     0x414ab0 <_Unwind_RaiseException>
0xfffe793adcb0: 0x4000000000000000      0x0
0xfffe793adcc0: 0x0     0x0
0xfffe793adcd0: 0x0     0x0
0xfffe793adce0: 0x0     0x0
0xfffe793adcf0: 0x0     0x0
0xfffe793add00: 0x0     0x0
0xfffe793add10: 0x0     0x0
0xfffe793add20: 0x0     0x0
0xfffe793add30: 0xfffe793ad8b0  0xfffe793ad8b8
0xfffe793add40: 0xfffe793ad8c0  0xfffe793ad8c8
0xfffe793add50: 0x0     0x0
0xfffe793add60: 0x0     0x0
0xfffe793add70: 0x0     0x0
0xfffe793add80: 0x0     0x0
0xfffe793add90: 0x0     0x0
0xfffe793adda0: 0x0     0x0
0xfffe793addb0: 0x0     0x0
0xfffe793addc0: 0x0     0xfffe793ae7b0
0xfffe793addd0: 0xfffe793ae7b8  0xfffe793ae7c0
0xfffe793adde0: 0xfffe793ad8e8  0xfffe793ad8f0
0xfffe793addf0: 0xfffe793ad8f8  0xfffe793ad900
0xfffe793ade00: 0xfffe793ad908  0xfffe793ad910
0xfffe793ade10: 0xfffe793ad918  0xfffe793ae7e0
0xfffe793ade20: 0xfffe793ae7e8  0xfffe793ad898
0xfffe793ade30: 0x0     0x0
0xfffe793ade40: 0x0     0x0
0xfffe793ade50: 0x0     0x0
0xfffe793ade60: 0x0     0x0
0xfffe793ade70: 0x0     0x0
```

```
0xfffe793ade80: 0x0     0x0
0xfffe793ade90: 0x0     0x0
0xfffe793adea0: 0x0     0x0
0xfffe793adeb0: 0x0     0x0
0xfffe793adec0: 0x0     0x0
0xfffe793aded0: 0x0     0x0
0xfffe793adee0: 0x0     0x0
0xfffe793adef0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe793adf00: 0x0     0x0
0xfffe793adf10: 0x0     0x0
0xfffe793adf20: 0x0     0x0
0xfffe793adf30: 0x0     0x0
0xfffe793adf40: 0x0     0x0
0xfffe793adf50: 0x0     0x0
0xfffe793adf60: 0x0     0x0
0xfffe793adf70: 0xfffe793ad920  0xfffe793ad928
0xfffe793adf80: 0xfffe793ad930  0xfffe793ad938
0xfffe793adf90: 0xfffe793ad940  0xfffe793ad948
0xfffe793adfa0: 0xfffe793ad950  0xfffe793ad958
0xfffe793adfb0: 0x0     0x0
0xfffe793adfc0: 0x0     0x0
0xfffe793adfd0: 0x0     0x0
0xfffe793adfe0: 0x0     0x0
0xfffe793adff0: 0x0     0x0
0xfffe793ae000: 0x0     0x0
0xfffe793ae010: 0x0     0x0
0xfffe793ae020: 0x0     0x0
0xfffe793ae030: 0x0     0x0
0xfffe793ae040: 0xfffe793ae7f0  0x4034c0 <_Z5procHv+20>
0xfffe793ae050: 0x4c57d8        0x0
0xfffe793ae060: 0x0     0x4034ac <_Z5procHv>
0xfffe793ae070: 0x4000000000000000      0x0
0xfffe793ae080: 0x0     0x0
0xfffe793ae090: 0x0     0x0
0xfffe793ae0a0: 0x0     0x0
0xfffe793ae0b0: 0x0     0x0
0xfffe793ae0c0: 0x0     0x0
0xfffe793ae0d0: 0x0     0x0
0xfffe793ae0e0: 0x0     0x0
0xfffe793ae0f0: 0x0     0x0
0xfffe793ae100: 0x0     0x0
0xfffe793ae110: 0x0     0x0
0xfffe793ae120: 0x0     0x0
0xfffe793ae130: 0x0     0x0
0xfffe793ae140: 0x0     0x0
0xfffe793ae150: 0x0     0x0
0xfffe793ae160: 0x0     0x0
0xfffe793ae170: 0x0     0x0
0xfffe793ae180: 0x0     0x0
0xfffe793ae190: 0x0     0x0
0xfffe793ae1a0: 0x0     0x0
0xfffe793ae1b0: 0x0     0x0
0xfffe793ae1c0: 0x0     0x0
0xfffe793ae1d0: 0x0     0x0
0xfffe793ae1e0: 0x0     0x0
0xfffe793ae1f0: 0x0     0x0
0xfffe793ae200: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe793ae210: 0x0     0x0
```

```
0xfffe793ae220: 0xfffffffffffffff0      0x1
0xfffe793ae230: 0x0     0x0
0xfffe793ae240: 0x0     0x0
0xfffe793ae250: 0x0     0x0
0xfffe793ae260: 0x0     0x0
0xfffe793ae270: 0x0     0x0
0xfffe793ae280: 0x0     0x0
0xfffe793ae290: 0x0     0x0
0xfffe793ae2a0: 0x0     0x0
0xfffe793ae2b0: 0x0     0x0
0xfffe793ae2c0: 0xfffffffffffffe0      0x1
0xfffe793ae2d0: 0xfffffffffffffe8      0x1
0xfffe793ae2e0: 0x0     0x0
0xfffe793ae2f0: 0x0     0x0
0xfffe793ae300: 0x0     0x0
0xfffe793ae310: 0x0     0x0
0xfffe793ae320: 0x0     0x0
0xfffe793ae330: 0x0     0x0
0xfffe793ae340: 0x0     0x0
0xfffe793ae350: 0x0     0x0
0xfffe793ae360: 0x0     0x0
0xfffe793ae370: 0x0     0x0
0xfffe793ae380: 0x0     0x0
0xfffe793ae390: 0x0     0x0
0xfffe793ae3a0: 0x0     0x0
0xfffe793ae3b0: 0x0     0x0
0xfffe793ae3c0: 0x0     0x0
0xfffe793ae3d0: 0x0     0x0
0xfffe793ae3e0: 0x0     0x0
0xfffe793ae3f0: 0x0     0x0
0xfffe793ae400: 0x0     0x0
0xfffe793ae410: 0x0     0x0
0xfffe793ae420: 0x0     0x0
0xfffe793ae430: 0x0     0x0
0xfffe793ae440: 0x0     0x0
0xfffe793ae450: 0x0     0x0
0xfffe793ae460: 0x0     0x0
0xfffe793ae470: 0x0     0x0
0xfffe793ae480: 0x0     0x0
0xfffe793ae490: 0xfffe793ae600  0x438e08 <sleep+228>
0xfffe793ae4a0: 0xffffffff      0x10000
0xfffe793ae4b0: 0x0     0x0
0xfffe793ae4c0: 0x0     0x0
0xfffe793ae4d0: 0x0     0x0
0xfffe793ae4e0: 0x0     0x0
0xfffe793ae4f0: 0x0     0x0
0xfffe793ae500: 0x0     0x0
0xfffe793ae510: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe793ae520: 0x0     0x0
0xfffe793ae530: 0x0     0x0
0xfffe793ae540: 0x0     0x0
0xfffe793ae550: 0x0     0x0
0xfffe793ae560: 0xfffe793ae600  0x438e28 <sleep+260>
0xfffe793ae570: 0xffffffff      0x10000
0xfffe793ae580: 0x0     0x0
0xfffe793ae590: 0x0     0x0
0xfffe793ae5a0: 0xfffe793ae600  0x420168 <nanosleep+24>
0xfffe793ae5b0: 0x0     0x0
0xfffe793ae5c0: 0x438e34 <sleep+272>    0xfffe793ae640
```

```
0xfffe793ae5d0: 0xfffe793ae640  0x0
0xfffe793ae5e0: 0x0     0x0
0xfffe793ae5f0: 0x0     0x0
0xfffe793ae600: 0xfffe793ae7f0  0x4034cc <_Z5procHv+32>
0xfffe793ae610: 0xfffe793af070  0x0
0xfffe793ae620: 0x4e0000        0x4035d8 <_Z12thread_threePv>
0xfffe793ae630: 0x0     0x0
0xfffe793ae640: 0xffffffff      0xb854a
0xfffe793ae650: 0x0     0x0
0xfffe793ae660: 0x0     0x0
0xfffe793ae670: 0x0     0x0
```

```
(gdb) disassemble 0x414bf4
Dump of assembler code for function _Unwind_RaiseException:
   0x0000000000414ab0 <+0>:     sub     sp, sp, #0xe10
   0x0000000000414ab4 <+4>:     stp     x29, x30, [sp, #-192]!
   0x0000000000414ab8 <+8>:     mov     x29, sp
   0x0000000000414abc <+12>:    stp     x21, x22, [sp, #64]
   0x0000000000414ac0 <+16>:    add     x22, x29, #0xd0
   0x0000000000414ac4 <+20>:    stp     x0, x1, [sp, #16]
   0x0000000000414ac8 <+24>:    stp     x2, x3, [sp, #32]
   0x0000000000414acc <+28>:    add     x1, x29, #0xed0
   0x0000000000414ad0 <+32>:    mov     x2, x30
   0x0000000000414ad4 <+36>:    mov     x21, x0
   0x0000000000414ad8 <+40>:    mov     x0, x22
   0x0000000000414adc <+44>:    stp     x19, x20, [sp, #48]
   0x0000000000414ae0 <+48>:    stp     d8, d9, [sp, #128]
   0x0000000000414ae4 <+52>:    stp     d10, d11, [sp, #144]
   0x0000000000414ae8 <+56>:    stp     d12, d13, [sp, #160]
   0x0000000000414aec <+60>:    stp     d14, d15, [sp, #176]
   0x0000000000414af0 <+64>:    stp     x23, x24, [sp, #80]
   0x0000000000414af4 <+68>:    stp     x25, x26, [sp, #96]
   0x0000000000414af8 <+72>:    stp     x27, x28, [sp, #112]
   0x0000000000414afc <+76>:    add     x19, x29, #0x490
   0x0000000000414b00 <+80>:    bl      0x414268 <uw_init_context_1>
   0x0000000000414b04 <+84>:    mov     x0, x19
   0x0000000000414b08 <+88>:    mov     x1, x22
   0x0000000000414b0c <+92>:    mov     x2, #0x3c0                      // #960
   0x0000000000414b10 <+96>:    bl      0x400280
   0x0000000000414b14 <+100>:   add     x20, x29, #0x850
   0x0000000000414b18 <+104>:   b       0x414b4c <_Unwind_RaiseException+156>
   0x0000000000414b1c <+108>:   cbnz    w2, 0x414bbc <_Unwind_RaiseException+268>
   0x0000000000414b20 <+112>:   ldr     x5, [x20, #1616]
   0x0000000000414b24 <+116>:   cbz     x5, 0x414b40 <_Unwind_RaiseException+144>
   0x0000000000414b28 <+120>:   ldr     x2, [x21]
   0x0000000000414b2c <+124>:   blr     x5
   0x0000000000414b30 <+128>:   cmp     w0, #0x6
   0x0000000000414b34 <+132>:   b.eq    0x414bc4 <_Unwind_RaiseException+276>  // b.none
   0x0000000000414b38 <+136>:   cmp     w0, #0x8
   0x0000000000414b3c <+140>:   b.ne    0x414bbc <_Unwind_RaiseException+268>  // b.any
   0x0000000000414b40 <+144>:   mov     x0, x19
   0x0000000000414b44 <+148>:   mov     x1, x20
   0x0000000000414b48 <+152>:   bl      0x414490 <uw_update_context>
   0x0000000000414b4c <+156>:   mov     x1, x20
   0x0000000000414b50 <+160>:   mov     x0, x19
   0x0000000000414b54 <+164>:   bl      0x413100 <uw_frame_state_for>
   0x0000000000414b58 <+168>:   mov     w2, w0
   0x0000000000414b5c <+172>:   cmp     w2, #0x5
   0x0000000000414b60 <+176>:   mov     w0, #0x1                        // #1
   0x0000000000414b64 <+180>:   mov     x3, x21
```

```
   0x0000000000414b68 <+184>:   mov     x4, x19
   0x0000000000414b6c <+188>:   mov     w1, w0
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000414b70 <+192>:   b.ne    0x414b1c <_Unwind_RaiseException+108>  // b.any
   0x0000000000414b74 <+196>:   mov     w0, w2
   0x0000000000414b78 <+200>:   mov     x4, #0x0                        // #0
   0x0000000000414b7c <+204>:   ldp     x2, x3, [sp, #32]
   0x0000000000414b80 <+208>:   ldp     x19, x20, [sp, #48]
   0x0000000000414b84 <+212>:   ldp     x21, x22, [sp, #64]
   0x0000000000414b88 <+216>:   ldp     x23, x24, [sp, #80]
   0x0000000000414b8c <+220>:   ldp     x25, x26, [sp, #96]
   0x0000000000414b90 <+224>:   ldp     x27, x28, [sp, #112]
   0x0000000000414b94 <+228>:   ldp     d8, d9, [sp, #128]
   0x0000000000414b98 <+232>:   ldp     d10, d11, [sp, #144]
   0x0000000000414b9c <+236>:   ldp     d12, d13, [sp, #160]
   0x0000000000414ba0 <+240>:   ldp     d14, d15, [sp, #176]
   0x0000000000414ba4 <+244>:   ldp     x0, x1, [sp, #16]
   0x0000000000414ba8 <+248>:   ldp     x29, x30, [sp], #192
   0x0000000000414bac <+252>:   mov     x16, sp
   0x0000000000414bb0 <+256>:   add     sp, sp, x4
   0x0000000000414bb4 <+260>:   add     sp, sp, #0xe10
   0x0000000000414bb8 <+264>:   ret
   0x0000000000414bbc <+268>:   mov     w0, #0x3                        // #3
   0x0000000000414bc0 <+272>:   b       0x414b78 <_Unwind_RaiseException+200>
   0x0000000000414bc4 <+276>:   ldr     x1, [x19, #784]
   0x0000000000414bc8 <+280>:   ldr     x0, [x19, #832]
   0x0000000000414bcc <+284>:   mov     x2, #0x3c0                      // #960
   0x0000000000414bd0 <+288>:   sub     x0, x1, x0, lsr #63
   0x0000000000414bd4 <+292>:   str     x0, [x21, #24]
   0x0000000000414bd8 <+296>:   mov     x1, x22
   0x0000000000414bdc <+300>:   str     xzr, [x21, #16]
   0x0000000000414be0 <+304>:   mov     x0, x19
   0x0000000000414be4 <+308>:   bl      0x400280
   0x0000000000414be8 <+312>:   mov     x0, x21
   0x0000000000414bec <+316>:   mov     x1, x19
   0x0000000000414bf0 <+320>:   bl      0x41453c <_Unwind_RaiseException_Phase2>
   0x0000000000414bf4 <+324>:   cmp     w0, #0x7
   0x0000000000414bf8 <+328>:   b.ne    0x414b78 <_Unwind_RaiseException+200>  // b.any
   0x0000000000414bfc <+332>:   mov     x1, x19
   0x0000000000414c00 <+336>:   mov     x0, x22
   0x0000000000414c04 <+340>:   bl      0x4146fc <uw_install_context_1>
   0x0000000000414c08 <+344>:   ldr     x1, [x22, #832]
   0x0000000000414c0c <+348>:   mov     x4, x0
   0x0000000000414c10 <+352>:   ldr     x20, [x19, #792]
   0x0000000000414c14 <+356>:   ldr     x0, [x22, #784]
   0x0000000000414c18 <+360>:   tbz     x1, #61, 0x414c2c <_Unwind_RaiseException+380>
   0x0000000000414c1c <+364>:   mov     x17, x20
   0x0000000000414c20 <+368>:   mov     x16, x0
   0x0000000000414c24 <+372>:   pacia1716
   0x0000000000414c28 <+376>:   mov     x20, x17
   0x0000000000414c2c <+380>:   ldr     x0, [x19, #784]
   0x0000000000414c30 <+384>:   mov     x1, x20
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000414c34 <+388>:   str     x4, [x29, #200]
   0x0000000000414c38 <+392>:   bl      0x414aa8 <_Unwind_DebugHook>
   0x0000000000414c3c <+396>:   ldr     x4, [x29, #200]
   0x0000000000414c40 <+400>:   str     x20, [sp, #8]
   0x0000000000414c44 <+404>:   b       0x414b7c <_Unwind_RaiseException+204>
End of assembler dump.
```

Exercise A8 (A64, WinDbg Preview)

Goal: Learn how to identify runtime exceptions, past execution residue and stack traces, identify handled exceptions.

Patterns: C++ Exception; Execution Residue (User Space); Coincidental Symbolic Information; Handled Exception (User Space).

1. Launch WinDbg Preview.

2. Load *core.25889* dump file from the A64\App8 folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App8\core.25889]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
(6521.6522): Signal SIGABRT code SI_TKILL (Sent by tkill system call) originating from PID
6521*** WARNING: Unable to verify timestamp for App8
App8+0x20cfc:
00000000`00420cfc ?? ???
```

3. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App8\App8.log
Opened log file 'C:\ALCDA2\A64\App8\App8.log'
```

4. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App8\
Symbol search path is: srv*;C:\ALCDA2\A64\App8\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app8\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App8\
*** WARNING: Unable to verify timestamp for App8
```

```
0:000> .reload
..
*** WARNING: Unable to verify timestamp for App8

************* Symbol Loading Error Summary **************
Module name            Error
App8                   The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

5. List all thread stack traces:

```
0:000> ~*k

Unable to get thread data for thread 0
.  0  Id: 6521.6522 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`7a3ce5c0 00000000`00422d38     App8!raise+0x2c
01 0000fffe`7a3ce5c0 00000000`004086f0     App8!abort+0x128
02 0000fffe`7a3ce710 00000000`00404c0c     App8!__gnu_cxx::__verbose_terminate_handler+0x16c
03 0000fffe`7a3ce760 00000000`00404c30 (T) App8!__cxxabiv1::__terminate+0xc
04 0000fffe`7a3ce770 00000000`00404d88 (T) App8!std::terminate+0x14
05 0000fffe`7a3ce780 00000000`00403424     App8!_cxa_throw+0x98
06 0000fffe`7a3ce7b0 00000000`00403490     App8!procB+0x7c
07 0000fffe`7a3ce7f0 00000000`00403504     App8!procA+0xc
08 0000fffe`7a3ce800 00000000`00403534     App8!procNH+0x14
09 0000fffe`7a3ce810 00000000`00403548     App8!bar_one+0xc
0a 0000fffe`7a3ce820 00000000`00403560     App8!foo_one+0xc
0b 0000fffe`7a3ce830 00000000`004183f4     App8!thread_one+0x10
0c 0000fffe`7a3ce850 00000000`0043dd20     App8!start_thread+0xb4
0d 0000fffe`7a3ce980 ffffffff`ffffffff     App8!thread_start+0x30
0e 0000fffe`7a3ce980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 1
   1  Id: 6521.6524 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`793ae5c0 00000000`00438e34     App8!_libc_nanosleep+0x24
01 0000fffe`793ae600 00000000`004034cc     App8!sleep+0x110
02 0000fffe`793ae7f0 00000000`004035bc     App8!procH+0x20
03 0000fffe`793ae810 00000000`004035d0     App8!bar_three+0xc
04 0000fffe`793ae820 00000000`004035e8     App8!foo_three+0xc
05 0000fffe`793ae830 00000000`004183f4     App8!thread_three+0x10
06 0000fffe`793ae850 00000000`0043dd20     App8!start_thread+0xb4
07 0000fffe`793ae980 ffffffff`ffffffff     App8!thread_start+0x30
08 0000fffe`793ae980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 2
   2  Id: 6521.6521 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`d79fcc20 00000000`00438e34     App8!_libc_nanosleep+0x24
01 0000ffff`d79fcc60 00000000`0040370c     App8!sleep+0x110
02 0000ffff`d79fce50 00000000`0042240c     App8!main+0x90
03 0000ffff`d79fcea0 00000000`00403188     App8!_libc_start_main+0x304
04 0000ffff`d79fd000 00000000`00000000     App8!start+0x4c
```

```
Unable to get thread data for thread 3
   3  Id: 6521.6523 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`79bbe5d0 00000000`00438e34     App8!_libc_nanosleep+0x24
01 0000fffe`79bbe610 00000000`00403520     App8!sleep+0x110
02 0000fffe`79bbe800 00000000`00403578     App8!procNE+0x14
03 0000fffe`79bbe810 00000000`0040358c     App8!bar_two+0xc
04 0000fffe`79bbe820 00000000`004035a4     App8!foo_two+0xc
05 0000fffe`79bbe830 00000000`004183f4     App8!thread_two+0x10
06 0000fffe`79bbe850 00000000`0043dd20     App8!start_thread+0xb4
07 0000fffe`79bbe980 ffffffff`ffffffff     App8!thread_start+0x30
08 0000fffe`79bbe980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 4
   4  Id: 6521.6526 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`73ffe5d0 00000000`00438e34     App8!_libc_nanosleep+0x24
01 0000fffe`73ffe610 00000000`00403520     App8!sleep+0x110
02 0000fffe`73ffe800 00000000`00403644     App8!procNE+0x14
03 0000fffe`73ffe810 00000000`00403658     App8!bar_five+0xc
04 0000fffe`73ffe820 00000000`00403670     App8!foo_five+0xc
05 0000fffe`73ffe830 00000000`004183f4     App8!thread_five+0x10
06 0000fffe`73ffe850 00000000`0043dd20     App8!start_thread+0xb4
07 0000fffe`73ffe980 ffffffff`ffffffff     App8!thread_start+0x30
08 0000fffe`73ffe980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 5
   5  Id: 6521.6525 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`78b9e5d0 00000000`00438e34     App8!_libc_nanosleep+0x24
01 0000fffe`78b9e610 00000000`00403520     App8!sleep+0x110
02 0000fffe`78b9e800 00000000`00403600     App8!procNE+0x14
03 0000fffe`78b9e810 00000000`00403614     App8!bar_four+0xc
04 0000fffe`78b9e820 00000000`0040362c     App8!foo_four+0xc
05 0000fffe`78b9e830 00000000`004183f4     App8!thread_four+0x10
06 0000fffe`78b9e850 00000000`0043dd20     App8!start_thread+0xb4
07 0000fffe`78b9e980 ffffffff`ffffffff     App8!thread_start+0x30
08 0000fffe`78b9e980 00000000`00000000     0xffffffff`ffffffff
```

Note: We have C++ exception processing in the current thread #0.

6. Go to thread #3, identify the execution residue of *work* functions, check their correctness, and reconstruct the past stack trace:

```
0:000> ~3s
App8!_libc_nanosleep+0x24:
00000000`00420174 d4000001 svc         #0

0:003> k
 # Child-SP          RetAddr               Call Site
00 0000fffe`79bbe5d0 00000000`00438e34     App8!_libc_nanosleep+0x24
01 0000fffe`79bbe610 00000000`00403520     App8!sleep+0x110
02 0000fffe`79bbe800 00000000`00403578     App8!procNE+0x14
03 0000fffe`79bbe810 00000000`0040358c     App8!bar_two+0xc
04 0000fffe`79bbe820 00000000`004035a4     App8!foo_two+0xc
05 0000fffe`79bbe830 00000000`004183f4     App8!thread_two+0x10
06 0000fffe`79bbe850 00000000`0043dd20     App8!start_thread+0xb4
07 0000fffe`79bbe980 ffffffff`ffffffff     App8!thread_start+0x30
08 0000fffe`79bbe980 00000000`00000000     0xffffffff`ffffffff
```

```
0:003> dps sp-300 sp
0000fffe`79bbe2d0  00000000`00000000
0000fffe`79bbe2d8  00000000`00000000
0000fffe`79bbe2e0  00000000`00000000
0000fffe`79bbe2e8  00000000`00000000
0000fffe`79bbe2f0  00000000`00000000
0000fffe`79bbe2f8  00000000`00000000
0000fffe`79bbe300  00000000`00000000
0000fffe`79bbe308  00000000`00000000
0000fffe`79bbe310  00000000`00000000
0000fffe`79bbe318  00000000`00000000
0000fffe`79bbe320  00000000`00000000
0000fffe`79bbe328  00000000`00000000
0000fffe`79bbe330  00000000`00000000
0000fffe`79bbe338  00000000`00000000
0000fffe`79bbe340  00000000`00000000
0000fffe`79bbe348  00000000`00000000
0000fffe`79bbe350  0000fffe`79bbe360
0000fffe`79bbe358  00000000`00403304 App8!work_7+0xc
0000fffe`79bbe360  0000fffe`79bbe370
0000fffe`79bbe368  00000000`00403318 App8!work_6+0xc
0000fffe`79bbe370  0000fffe`79bbe380
0000fffe`79bbe378  00000000`0040332c App8!work_5+0xc
0000fffe`79bbe380  0000fffe`79bbe390
0000fffe`79bbe388  00000000`00403340 App8!work_4+0xc
0000fffe`79bbe390  0000fffe`79bbe3a0
0000fffe`79bbe398  00000000`00403354 App8!work_3+0xc
0000fffe`79bbe3a0  0000fffe`79bbe3b0
0000fffe`79bbe3a8  00000000`00403368 App8!work_2+0xc
0000fffe`79bbe3b0  0000fffe`79bbe3c0
0000fffe`79bbe3b8  00000000`0040337c App8!work_1+0xc
0000fffe`79bbe3c0  0000fffe`79bbe3d0
0000fffe`79bbe3c8  00000000`00403394 App8!work+0x10
0000fffe`79bbe3d0  0000fffe`79bbe7e0
0000fffe`79bbe3d8  00000000`0040347c App8!procNB+0xc
0000fffe`79bbe3e0  00000000`00000000
0000fffe`79bbe3e8  00000000`00000000
0000fffe`79bbe3f0  00000000`00000000
0000fffe`79bbe3f8  00000000`00000000
0000fffe`79bbe400  00000000`00000000
0000fffe`79bbe408  00000000`00000000
0000fffe`79bbe410  00000000`00000000
0000fffe`79bbe418  00000000`00000000
0000fffe`79bbe420  00000000`00000000
0000fffe`79bbe428  00000000`00000000
0000fffe`79bbe430  00000000`00000000
0000fffe`79bbe438  00000000`00000000
0000fffe`79bbe440  00000000`00000000
0000fffe`79bbe448  00000000`00000000
0000fffe`79bbe450  00000000`00000000
0000fffe`79bbe458  00000000`00000000
0000fffe`79bbe460  00000000`00000000
0000fffe`79bbe468  00000000`00000000
0000fffe`79bbe470  00000000`00000000
0000fffe`79bbe478  00000000`00000000
0000fffe`79bbe480  00000000`00000000
0000fffe`79bbe488  00000000`00000000
0000fffe`79bbe490  00000000`00000000
0000fffe`79bbe498  00000000`00000000
```

```
0000fffe`79bbe4a0  0000fffe`79bbe610
0000fffe`79bbe4a8  00000000`00438e08 App8!sleep+0xe4
0000fffe`79bbe4b0  00000000`ffffffff
0000fffe`79bbe4b8  00000000`00010000
0000fffe`79bbe4c0  00000000`00000000
0000fffe`79bbe4c8  00000000`00000000
0000fffe`79bbe4d0  00000000`00000000
0000fffe`79bbe4d8  00000000`00000000
0000fffe`79bbe4e0  00000000`00000000
0000fffe`79bbe4e8  00000000`00000000
0000fffe`79bbe4f0  00000000`00000000
0000fffe`79bbe4f8  00000000`00000000
0000fffe`79bbe500  00000000`00000000
0000fffe`79bbe508  00000000`00000000
0000fffe`79bbe510  00000000`00000000
0000fffe`79bbe518  00000000`00000000
0000fffe`79bbe520  00000000`00000000
0000fffe`79bbe528  00000000`00000000
0000fffe`79bbe530  00000000`00000000
0000fffe`79bbe538  00000000`00000000
0000fffe`79bbe540  00000000`00000000
0000fffe`79bbe548  00000000`00000000
0000fffe`79bbe550  00000000`00000000
0000fffe`79bbe558  00000000`00000000
0000fffe`79bbe560  00000000`00000000
0000fffe`79bbe568  00000000`00000000
0000fffe`79bbe570  0000fffe`79bbe610
0000fffe`79bbe578  00000000`00438e28 App8!sleep+0x104
0000fffe`79bbe580  00000000`ffffffff
0000fffe`79bbe588  00000000`00010000
0000fffe`79bbe590  00000000`00000000
0000fffe`79bbe598  00000000`00000000
0000fffe`79bbe5a0  00000000`00000000
0000fffe`79bbe5a8  00000000`00000000
0000fffe`79bbe5b0  0000fffe`79bbe610
0000fffe`79bbe5b8  00000000`00420168 App8!_libc_nanosleep+0x18
0000fffe`79bbe5c0  00000000`00000000
0000fffe`79bbe5c8  00000000`00000000
0000fffe`79bbe5d0  00000000`00438e34 App8!sleep+0x110

0:003> ub 00000000`00403354
App8!work_4:
00000000`00403334 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`00403338 910003fd mov         fp,sp
00000000`0040333c 97fffff9 bl          App8!work_5 (00000000`00403320)
00000000`00403340 a8c17bfd ldp         fp,lr,[sp],#0x10
00000000`00403344 d65f03c0 ret
App8!work_3:
00000000`00403348 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`0040334c 910003fd mov         fp,sp
00000000`00403350 97fffff9 bl          App8!work_4 (00000000`00403334)
```

Note: Since the saved fp value points to the next line we can easily reconstract the fragment of the past stack trace:

```
0000fffe`79bbe350  0000fffe`79bbe360
0000fffe`79bbe358  00000000`00403304 App8!work_7+0xc
0000fffe`79bbe360  0000fffe`79bbe370
0000fffe`79bbe368  00000000`00403318 App8!work_6+0xc
0000fffe`79bbe370  0000fffe`79bbe380
0000fffe`79bbe378  00000000`0040332c App8!work_5+0xc
0000fffe`79bbe380  0000fffe`79bbe390
0000fffe`79bbe388  00000000`00403340 App8!work_4+0xc
0000fffe`79bbe390  0000fffe`79bbe3a0
0000fffe`79bbe398  00000000`00403354 App8!work_3+0xc
0000fffe`79bbe3a0  0000fffe`79bbe3b0
0000fffe`79bbe3a8  00000000`00403368 App8!work_2+0xc
0000fffe`79bbe3b0  0000fffe`79bbe3c0
0000fffe`79bbe3b8  00000000`0040337c App8!work_1+0xc
0000fffe`79bbe3c0  0000fffe`79bbe3d0
0000fffe`79bbe3c8  00000000`00403394 App8!work+0x10
0000fffe`79bbe3d0  0000fffe`79bbe7e0
0000fffe`79bbe3d8  00000000`0040347c App8!procNB+0xc
```

7. Go to thread #1, identify the handled exception processing code, and check its validity:

```
0:003> ~1s
App8!_libc_nanosleep+0x24:
00000000`00420174 d4000001 svc         #0

0:001> k
 # Child-SP          RetAddr               Call Site
00 0000fffe`793ae5c0 00000000`00438e34     App8!_libc_nanosleep+0x24
01 0000fffe`793ae600 00000000`004034cc     App8!sleep+0x110
02 0000fffe`793ae7f0 00000000`004035bc     App8!procH+0x20
03 0000fffe`793ae810 00000000`004035d0     App8!bar_three+0xc
04 0000fffe`793ae820 00000000`004035e8     App8!foo_three+0xc
05 0000fffe`793ae830 00000000`004183f4     App8!thread_three+0x10
06 0000fffe`793ae850 00000000`0043dd20     App8!start_thread+0xb4
07 0000fffe`793ae980 ffffffff`ffffffff     App8!thread_start+0x30
08 0000fffe`793ae980 00000000`00000000     0xffffffff`ffffffff

0:001> dps sp-2000 sp
0000fffe`793ac5c0  00000000`00000000
0000fffe`793ac5c8  00000000`00000000
0000fffe`793ac5d0  00000000`00000000
0000fffe`793ac5d8  00000000`00000000
0000fffe`793ac5e0  00000000`00000000
0000fffe`793ac5e8  00000000`00000000
0000fffe`793ac5f0  00000000`00000000
0000fffe`793ac5f8  00000000`00000000
0000fffe`793ac600  00000000`00000000
0000fffe`793ac608  00000000`00000000
0000fffe`793ac610  00000000`00000000
0000fffe`793ac618  00000000`00000000
0000fffe`793ac620  00000000`00000000
0000fffe`793ac628  00000000`00000000
0000fffe`793ac630  00000000`00000000
0000fffe`793ac638  00000000`00000000
0000fffe`793ac640  00000000`00000000
0000fffe`793ac648  00000000`00000000
0000fffe`793ac650  00000000`00000000
0000fffe`793ac658  00000000`00000000
```

```
0000fffe`793ac660  00000000`00000000
0000fffe`793ac668  00000000`00000000
0000fffe`793ac670  00000000`00000000
0000fffe`793ac678  00000000`00000000
0000fffe`793ac680  00000000`00000000
0000fffe`793ac688  00000000`00000000
0000fffe`793ac690  00000000`00000000
0000fffe`793ac698  00000000`00000000
0000fffe`793ac6a0  00000000`00000000
0000fffe`793ac6a8  00000000`00000000
0000fffe`793ac6b0  00000000`00000000
0000fffe`793ac6b8  00000000`00000000
0000fffe`793ac6c0  00000000`00000000
0000fffe`793ac6c8  00000000`00000000
0000fffe`793ac6d0  00000000`00000000
0000fffe`793ac6d8  00000000`00000000
0000fffe`793ac6e0  00000000`00000000
0000fffe`793ac6e8  00000000`00000000
0000fffe`793ac6f0  00000000`00000000
0000fffe`793ac6f8  00000000`00000000
0000fffe`793ac700  00000000`00000000
0000fffe`793ac708  00000000`00000000
0000fffe`793ac710  00000000`00000000
0000fffe`793ac718  00000000`00000000
0000fffe`793ac720  00000000`00000000
0000fffe`793ac728  00000000`00000000
0000fffe`793ac730  00000000`00000000
0000fffe`793ac738  00000000`00000000
0000fffe`793ac740  00000000`00000000
0000fffe`793ac748  00000000`00000000
0000fffe`793ac750  00000000`00000000
0000fffe`793ac758  00000000`00000000
0000fffe`793ac760  00000000`00000000
0000fffe`793ac768  00000000`00000000
0000fffe`793ac770  00000000`00000000
0000fffe`793ac778  00000000`00000000
0000fffe`793ac780  00000000`00000000
0000fffe`793ac788  00000000`00000000
0000fffe`793ac790  00000000`00000000
0000fffe`793ac798  00000000`00000000
0000fffe`793ac7a0  00000000`00000000
0000fffe`793ac7a8  00000000`00000000
0000fffe`793ac7b0  00000000`00000000
0000fffe`793ac7b8  00000000`00000000
0000fffe`793ac7c0  00000000`00000000
0000fffe`793ac7c8  00000000`00000000
0000fffe`793ac7d0  00000000`00000000
0000fffe`793ac7d8  00000000`00000000
0000fffe`793ac7e0  00000000`00000000
0000fffe`793ac7e8  00000000`00000000
0000fffe`793ac7f0  00000000`00000000
0000fffe`793ac7f8  00000000`00000000
0000fffe`793ac800  00000000`00000000
0000fffe`793ac808  00000000`00000000
0000fffe`793ac810  00000000`00000000
0000fffe`793ac818  00000000`00000000
0000fffe`793ac820  00000000`00000000
0000fffe`793ac828  00000000`00000000
0000fffe`793ac830  00000000`00000000
0000fffe`793ac838  00000000`00000000
```

```
0000fffe`793ac840  00000000`00000000
0000fffe`793ac848  00000000`00000000
0000fffe`793ac850  00000000`00000000
0000fffe`793ac858  00000000`00000000
0000fffe`793ac860  00000000`00000000
0000fffe`793ac868  00000000`00000000
0000fffe`793ac870  00000000`00000000
0000fffe`793ac878  00000000`00000000
0000fffe`793ac880  00000000`00000000
0000fffe`793ac888  00000000`00000000
0000fffe`793ac890  00000000`00000000
0000fffe`793ac898  00000000`00000000
0000fffe`793ac8a0  00000000`00000000
0000fffe`793ac8a8  00000000`00000000
0000fffe`793ac8b0  00000000`00000000
0000fffe`793ac8b8  00000000`00000000
0000fffe`793ac8c0  00000000`00000000
0000fffe`793ac8c8  00000000`00000000
0000fffe`793ac8d0  00000000`00000000
0000fffe`793ac8d8  00000000`00000000
0000fffe`793ac8e0  00000000`00000000
0000fffe`793ac8e8  00000000`00000000
0000fffe`793ac8f0  00000000`00000000
0000fffe`793ac8f8  00000000`00000000
0000fffe`793ac900  00000000`00000000
0000fffe`793ac908  00000000`00000000
0000fffe`793ac910  00000000`00000000
0000fffe`793ac918  00000000`00000000
0000fffe`793ac920  00000000`00000000
0000fffe`793ac928  00000000`00000000
0000fffe`793ac930  00000000`00000000
0000fffe`793ac938  00000000`00000000
0000fffe`793ac940  00000000`00000000
0000fffe`793ac948  00000000`00000000
0000fffe`793ac950  00000000`00000000
0000fffe`793ac958  00000000`00000000
0000fffe`793ac960  00000000`00000000
0000fffe`793ac968  00000000`00000000
0000fffe`793ac970  00000000`00000000
0000fffe`793ac978  00000000`00000000
0000fffe`793ac980  00000000`00000000
0000fffe`793ac988  00000000`00000000
0000fffe`793ac990  00000000`00000000
0000fffe`793ac998  00000000`00000000
0000fffe`793ac9a0  00000000`00000000
0000fffe`793ac9a8  00000000`00000000
0000fffe`793ac9b0  00000000`00000000
0000fffe`793ac9b8  00000000`00000000
0000fffe`793ac9c0  00000000`00000000
0000fffe`793ac9c8  00000000`00000000
0000fffe`793ac9d0  00000000`00000000
0000fffe`793ac9d8  00000000`00000000
0000fffe`793ac9e0  00000000`00000000
0000fffe`793ac9e8  00000000`00000000
0000fffe`793ac9f0  00000000`00000000
0000fffe`793ac9f8  00000000`00000000
0000fffe`793aca00  00000000`00000000
0000fffe`793aca08  00000000`00000000
0000fffe`793aca10  00000000`00000000
0000fffe`793aca18  00000000`00000000
```

```
0000fffe`793aca20  00000000`00000000
0000fffe`793aca28  00000000`00000000
0000fffe`793aca30  00000000`00000000
0000fffe`793aca38  00000000`00000000
0000fffe`793aca40  00000000`00000000
0000fffe`793aca48  00000000`00000000
0000fffe`793aca50  00000000`00000000
0000fffe`793aca58  00000000`00000000
0000fffe`793aca60  00000000`00000000
0000fffe`793aca68  00000000`00000000
0000fffe`793aca70  00000000`00000000
0000fffe`793aca78  00000000`00000000
0000fffe`793aca80  00000000`00000000
0000fffe`793aca88  00000000`00000000
0000fffe`793aca90  00000000`00000000
0000fffe`793aca98  00000000`00000000
0000fffe`793acaa0  00000000`00000000
0000fffe`793acaa8  00000000`00000000
0000fffe`793acab0  00000000`00000000
0000fffe`793acab8  00000000`00000000
0000fffe`793acac0  00000000`00000000
0000fffe`793acac8  00000000`00000000
0000fffe`793acad0  00000000`00000000
0000fffe`793acad8  00000000`00000000
0000fffe`793acae0  00000000`00000000
0000fffe`793acae8  00000000`00000000
0000fffe`793acaf0  00000000`00000000
0000fffe`793acaf8  00000000`00000000
0000fffe`793acb00  00000000`00000000
0000fffe`793acb08  00000000`00000000
0000fffe`793acb10  00000000`00000000
0000fffe`793acb18  00000000`00000000
0000fffe`793acb20  00000000`00000000
0000fffe`793acb28  00000000`00000000
0000fffe`793acb30  00000000`00000000
0000fffe`793acb38  00000000`00000000
0000fffe`793acb40  00000000`00000000
0000fffe`793acb48  00000000`00000000
0000fffe`793acb50  00000000`00000000
0000fffe`793acb58  00000000`00000000
0000fffe`793acb60  00000000`00000000
0000fffe`793acb68  00000000`00000000
0000fffe`793acb70  00000000`00000000
0000fffe`793acb78  00000000`00000000
0000fffe`793acb80  00000000`00000000
0000fffe`793acb88  00000000`00000000
0000fffe`793acb90  00000000`00000000
0000fffe`793acb98  00000000`00000000
0000fffe`793acba0  00000000`00000000
0000fffe`793acba8  00000000`00000000
0000fffe`793acbb0  00000000`00000000
0000fffe`793acbb8  00000000`00000000
0000fffe`793acbc0  00000000`00000000
0000fffe`793acbc8  00000000`00000000
0000fffe`793acbd0  00000000`00000000
0000fffe`793acbd8  00000000`00000000
0000fffe`793acbe0  00000000`00000000
0000fffe`793acbe8  00000000`00000000
0000fffe`793acbf0  00000000`00000000
0000fffe`793acbf8  00000000`00000000
```

```
0000fffe`793acc00  00000000`00000000
0000fffe`793acc08  00000000`00000000
0000fffe`793acc10  00000000`00000000
0000fffe`793acc18  00000000`00000000
0000fffe`793acc20  00000000`00000000
0000fffe`793acc28  00000000`00000000
0000fffe`793acc30  00000000`00000000
0000fffe`793acc38  00000000`00000000
0000fffe`793acc40  00000000`00000000
0000fffe`793acc48  00000000`00000000
0000fffe`793acc50  00000000`00000000
0000fffe`793acc58  00000000`00000000
0000fffe`793acc60  00000000`00000000
0000fffe`793acc68  00000000`00000000
0000fffe`793acc70  00000000`00000000
0000fffe`793acc78  00000000`00000000
0000fffe`793acc80  00000000`00000000
0000fffe`793acc88  00000000`00000000
0000fffe`793acc90  00000000`00000000
0000fffe`793acc98  00000000`00000000
0000fffe`793acca0  00000000`00000000
0000fffe`793acca8  00000000`00000000
0000fffe`793accb0  00000000`00000000
0000fffe`793accb8  00000000`00000000
0000fffe`793accc0  00000000`00000000
0000fffe`793accc8  00000000`00000000
0000fffe`793accd0  00000000`00000000
0000fffe`793accd8  00000000`00000000
0000fffe`793acce0  00000000`00000000
0000fffe`793acce8  00000000`00000000
0000fffe`793accf0  00000000`00000000
0000fffe`793accf8  00000000`00000000
0000fffe`793acd00  00000000`00000000
0000fffe`793acd08  00000000`00000000
0000fffe`793acd10  00000000`00000000
0000fffe`793acd18  00000000`00000000
0000fffe`793acd20  00000000`00000000
0000fffe`793acd28  00000000`00000000
0000fffe`793acd30  00000000`00000000
0000fffe`793acd38  00000000`00000000
0000fffe`793acd40  00000000`00000000
0000fffe`793acd48  00000000`00000000
0000fffe`793acd50  00000000`00000000
0000fffe`793acd58  00000000`00000000
0000fffe`793acd60  00000000`00000000
0000fffe`793acd68  00000000`00000000
0000fffe`793acd70  00000000`00000000
0000fffe`793acd78  00000000`00000000
0000fffe`793acd80  0000fffe`793ad1c0
0000fffe`793acd88  00000000`004144a8 App8!uw_update_context+0x18
0000fffe`793acd90  0000fffe`793add30
0000fffe`793acd98  0000fffe`793ad220
0000fffe`793acda0  00000000`00000000
0000fffe`793acda8  0000fffe`74000b80
0000fffe`793acdb0  00000000`00000004
0000fffe`793acdb8  0000fffe`793af760
0000fffe`793acdc0  00000000`32b706f0
0000fffe`793acdc8  00000000`004f9540 App8!_default_pthread_attr
0000fffe`793acdd0  00000000`00010000
0000fffe`793acdd8  00000000`00810000
```

```
0000fffe`793acde0  00000000`004f3000 App8!ZL16emergency_buffer+0xfdf0
0000fffe`793acde8  0000fffe`793ad220
0000fffe`793acdf0  00000000`004f3000 App8!ZL16emergency_buffer+0xfdf0
0000fffe`793acdf8  0000fffe`793ae7e0
0000fffe`793ace00  0000fffe`793ad8b0
0000fffe`793ace08  0000fffe`793ad8b8
0000fffe`793ace10  0000fffe`793ad8c0
0000fffe`793ace18  0000fffe`793ad8c8
0000fffe`793ace20  00000000`00000000
0000fffe`793ace28  00000000`00000000
0000fffe`793ace30  00000000`00000000
0000fffe`793ace38  00000000`00000000
0000fffe`793ace40  00000000`00000000
0000fffe`793ace48  00000000`00000000
0000fffe`793ace50  00000000`00000000
0000fffe`793ace58  00000000`00000000
0000fffe`793ace60  00000000`00000000
0000fffe`793ace68  00000000`00000000
0000fffe`793ace70  00000000`00000000
0000fffe`793ace78  00000000`00000000
0000fffe`793ace80  00000000`00000000
0000fffe`793ace88  00000000`00000000
0000fffe`793ace90  00000000`00000000
0000fffe`793ace98  0000fffe`793ae7b0
0000fffe`793acea0  0000fffe`793ae7b8
0000fffe`793acea8  0000fffe`793ae7c0
0000fffe`793aceb0  0000fffe`793ad8e8
0000fffe`793aceb8  0000fffe`793ad8f0
0000fffe`793acec0  0000fffe`793ad8f8
0000fffe`793acec8  0000fffe`793ad900
0000fffe`793aced0  0000fffe`793ad908
0000fffe`793aced8  0000fffe`793ad910
0000fffe`793acee0  0000fffe`793ad918
0000fffe`793acee8  0000fffe`793ae7a0
0000fffe`793acef0  0000fffe`793ae7a8
0000fffe`793acef8  0000fffe`793acdf8
0000fffe`793acf00  00000000`00000000
0000fffe`793acf08  00000000`00000000
0000fffe`793acf10  00000000`00000000
0000fffe`793acf18  00000000`00000000
0000fffe`793acf20  00000000`00000000
0000fffe`793acf28  00000000`00000000
0000fffe`793acf30  00000000`00000000
0000fffe`793acf38  00000000`00000000
0000fffe`793acf40  00000000`00000000
0000fffe`793acf48  00000000`00000000
0000fffe`793acf50  00000000`00000000
0000fffe`793acf58  00000000`00000000
0000fffe`793acf60  00000000`00000000
0000fffe`793acf68  00000000`00000000
0000fffe`793acf70  00000000`00000000
0000fffe`793acf78  00000000`00000000
0000fffe`793acf80  00000000`00000000
0000fffe`793acf88  00000000`00000000
0000fffe`793acf90  00000000`00000000
0000fffe`793acf98  00000000`00000000
0000fffe`793acfa0  00000000`00000000
0000fffe`793acfa8  00000000`00000000
0000fffe`793acfb0  00000000`00000000
0000fffe`793acfb8  00000000`00000000
```

```
0000fffe`793acfc0  00000000`00000000
0000fffe`793acfc8  00000000`00000000
0000fffe`793acfd0  00000000`00000000
0000fffe`793acfd8  00000000`00000000
0000fffe`793acfe0  00000000`00000000
0000fffe`793acfe8  00000000`00000000
0000fffe`793acff0  0000fffe`793ad030
0000fffe`793acff8  00000000`00416218 App8!search_object+0x204
0000fffe`793ad000  00000000`004b2fb8 App8!$d+0x25c
0000fffe`793ad008  00000000`0000002d
0000fffe`793ad010  00000000`00000000
0000fffe`793ad018  00000000`00000000
0000fffe`793ad020  0000fffe`793ad030
0000fffe`793ad028  00000000`004161a0 App8!search_object+0x18c
0000fffe`793ad030  0000fffe`793ad0d0
0000fffe`793ad038  00000000`00416b28 App8!Unwind_Find_FDE+0x174
0000fffe`793ad040  00000000`004e15a8 App8!object.6205
0000fffe`793ad048  0000fffe`793ad220
0000fffe`793ad050  00000000`004034bb App8!Z5procHv+0xf
0000fffe`793ad058  0000fffe`74000b80
0000fffe`793ad060  00000000`004f3000 App8!ZL16emergency_buffer+0xfdf0
0000fffe`793ad068  0000fffe`793ae058
0000fffe`793ad070  00000000`004df000 App8!ZTIh+0x8
0000fffe`793ad078  00000000`004f9540 App8!_default_pthread_attr
0000fffe`793ad080  00000000`00010000
0000fffe`793ad088  00000000`00810000
0000fffe`793ad090  0000fffe`793ad0d0
0000fffe`793ad098  00000000`00416ae4 App8!Unwind_Find_FDE+0x130
0000fffe`793ad0a0  00000000`004e15a8 App8!object.6205
0000fffe`793ad0a8  00000000`0000001b
0000fffe`793ad0b0  0000fffe`793add30
0000fffe`793ad0b8  00000000`004034ac App8!Z5procHv
0000fffe`793ad0c0  0000fffe`793ad0d0
0000fffe`793ad0c8  00000000`00416ad0 App8!Unwind_Find_FDE+0x11c
0000fffe`793ad0d0  0000fffe`793ad150
0000fffe`793ad0d8  00000000`004136cc App8!uw_frame_state_for+0x5cc
0000fffe`793ad0e0  0000fffe`793ad0f0
0000fffe`793ad0e8  00000000`00404754 App8!_gxx_personality_v0+0xf0
0000fffe`793ad0f0  0000fffe`793ad1e0
0000fffe`793ad0f8  00000000`004145ac App8!Unwind_RaiseException_Phase2+0x70
0000fffe`793ad100  0000fffe`793add30
0000fffe`793ad108  0000fffe`793ad220
0000fffe`793ad110  00000000`00000004
0000fffe`793ad118  0000fffe`74000b80
0000fffe`793ad120  00000000`00000004
0000fffe`793ad128  0000fffe`793af760
0000fffe`793ad130  00000000`32b706f0
0000fffe`793ad138  00000000`004f9540 App8!_default_pthread_attr
0000fffe`793ad140  00000000`00010000
0000fffe`793ad148  00000000`00810000
0000fffe`793ad150  0000fffe`793ad1e0
0000fffe`793ad158  00000000`00414570 App8!Unwind_RaiseException_Phase2+0x34
0000fffe`793ad160  0000fffe`793add30
0000fffe`793ad168  00000000`004c57d8 App8!$d+0x1
0000fffe`793ad170  00000000`00000000
0000fffe`793ad178  0000fffe`74000b80
0000fffe`793ad180  00000000`00000004
0000fffe`793ad188  00000000`00000000
0000fffe`793ad190  00000000`32b706f0
0000fffe`793ad198  00000000`004f9540 App8!_default_pthread_attr
```

```
0000fffe`793ad1a0  00000000`00010000
0000fffe`793ad1a8  00000000`00810000
0000fffe`793ad1b0  00000000`00000000
0000fffe`793ad1b8  00000000`00000000
0000fffe`793ad1c0  0000fffe`793ad1e0
0000fffe`793ad1c8  00000000`004145cc App8!Unwind_RaiseException_Phase2+0x90
0000fffe`793ad1d0  ffffffff`fffffff8
0000fffe`793ad1d8  76a28b43`6af36f00
0000fffe`793ad1e0  0000fffe`793ad8a0
0000fffe`793ad1e8  00000000`00414bf4 App8!Unwind_RaiseException+0x144
0000fffe`793ad1f0  0000fffe`793add30
0000fffe`793ad1f8  0000fffe`793ae0f0
0000fffe`793ad200  0000fffe`74000b80
0000fffe`793ad208  0000fffe`793ad970
0000fffe`793ad210  00000000`00000000
0000fffe`793ad218  0000fffe`793ae770
0000fffe`793ad220  00000000`00000000
0000fffe`793ad228  00000000`00000000
0000fffe`793ad230  00000000`00000000
0000fffe`793ad238  00000000`00000000
0000fffe`793ad240  00000000`00000000
0000fffe`793ad248  00000000`00000000
0000fffe`793ad250  00000000`00000000
0000fffe`793ad258  00000000`00000000
0000fffe`793ad260  00000000`00000000
0000fffe`793ad268  00000000`00000000
0000fffe`793ad270  00000000`00000000
0000fffe`793ad278  00000000`00000000
0000fffe`793ad280  00000000`00000000
0000fffe`793ad288  00000000`00000000
0000fffe`793ad290  00000000`00000000
0000fffe`793ad298  00000000`00000000
0000fffe`793ad2a0  00000000`00000000
0000fffe`793ad2a8  00000000`00000000
0000fffe`793ad2b0  00000000`00000000
0000fffe`793ad2b8  00000000`00000000
0000fffe`793ad2c0  00000000`00000000
0000fffe`793ad2c8  00000000`00000000
0000fffe`793ad2d0  00000000`00000000
0000fffe`793ad2d8  00000000`00000000
0000fffe`793ad2e0  00000000`00000000
0000fffe`793ad2e8  00000000`00000000
0000fffe`793ad2f0  00000000`00000000
0000fffe`793ad2f8  00000000`00000000
0000fffe`793ad300  00000000`00000000
0000fffe`793ad308  00000000`00000000
0000fffe`793ad310  00000000`00000000
0000fffe`793ad318  00000000`00000000
0000fffe`793ad320  00000000`00000000
0000fffe`793ad328  00000000`00000000
0000fffe`793ad330  00000000`00000000
0000fffe`793ad338  00000000`00000000
0000fffe`793ad340  00000000`00000000
0000fffe`793ad348  00000000`00000000
0000fffe`793ad350  ffffffff`fffffff0
0000fffe`793ad358  00000000`00000001
0000fffe`793ad360  00000000`00000000
0000fffe`793ad368  00000000`00000000
0000fffe`793ad370  00000000`00000000
0000fffe`793ad378  00000000`00000000
```

```
0000fffe`793ad380  00000000`00000000
0000fffe`793ad388  00000000`00000000
0000fffe`793ad390  00000000`00000000
0000fffe`793ad398  00000000`00000000
0000fffe`793ad3a0  00000000`00000000
0000fffe`793ad3a8  00000000`00000000
0000fffe`793ad3b0  00000000`00000000
0000fffe`793ad3b8  00000000`00000000
0000fffe`793ad3c0  00000000`00000000
0000fffe`793ad3c8  00000000`00000000
0000fffe`793ad3d0  00000000`00000000
0000fffe`793ad3d8  00000000`00000000
0000fffe`793ad3e0  00000000`00000000
0000fffe`793ad3e8  00000000`00000000
0000fffe`793ad3f0  ffffffff`fffffe0
0000fffe`793ad3f8  00000000`00000001
0000fffe`793ad400  ffffffff`fffffe8
0000fffe`793ad408  00000000`00000001
0000fffe`793ad410  00000000`00000000
0000fffe`793ad418  00000000`00000000
0000fffe`793ad420  00000000`00000000
0000fffe`793ad428  00000000`00000000
0000fffe`793ad430  00000000`00000000
0000fffe`793ad438  00000000`00000000
0000fffe`793ad440  00000000`00000000
0000fffe`793ad448  00000000`00000000
0000fffe`793ad450  00000000`00000000
0000fffe`793ad458  00000000`00000000
0000fffe`793ad460  00000000`00000000
0000fffe`793ad468  00000000`00000000
0000fffe`793ad470  00000000`00000000
0000fffe`793ad478  00000000`00000000
0000fffe`793ad480  00000000`00000000
0000fffe`793ad488  00000000`00000000
0000fffe`793ad490  00000000`00000000
0000fffe`793ad498  00000000`00000000
0000fffe`793ad4a0  00000000`00000000
0000fffe`793ad4a8  00000000`00000000
0000fffe`793ad4b0  00000000`00000000
0000fffe`793ad4b8  00000000`00000000
0000fffe`793ad4c0  00000000`00000000
0000fffe`793ad4c8  00000000`00000000
0000fffe`793ad4d0  00000000`00000000
0000fffe`793ad4d8  00000000`00000000
0000fffe`793ad4e0  00000000`00000000
0000fffe`793ad4e8  00000000`00000000
0000fffe`793ad4f0  00000000`00000000
0000fffe`793ad4f8  00000000`00000000
0000fffe`793ad500  00000000`00000000
0000fffe`793ad508  00000000`00000000
0000fffe`793ad510  00000000`00000000
0000fffe`793ad518  00000000`00000000
0000fffe`793ad520  00000000`00000000
0000fffe`793ad528  00000000`00000000
0000fffe`793ad530  00000000`00000000
0000fffe`793ad538  00000000`00000000
0000fffe`793ad540  00000000`00000000
0000fffe`793ad548  00000000`00000000
0000fffe`793ad550  00000000`00000000
0000fffe`793ad558  00000000`00000000
```

```
0000fffe`793ad560  00000000`00000000
0000fffe`793ad568  00000000`00000000
0000fffe`793ad570  00000000`00000000
0000fffe`793ad578  00000000`00000000
0000fffe`793ad580  00000000`00000000
0000fffe`793ad588  00000000`00000000
0000fffe`793ad590  00000000`00000000
0000fffe`793ad598  00000000`00000000
0000fffe`793ad5a0  00000000`00000000
0000fffe`793ad5a8  00000000`00000000
0000fffe`793ad5b0  00000000`00000000
0000fffe`793ad5b8  00000000`00000000
0000fffe`793ad5c0  00000000`00000000
0000fffe`793ad5c8  00000000`00000000
0000fffe`793ad5d0  00000000`00000000
0000fffe`793ad5d8  00000000`00000000
0000fffe`793ad5e0  00000000`00000000
0000fffe`793ad5e8  00000000`00000000
0000fffe`793ad5f0  00000000`00000000
0000fffe`793ad5f8  00000000`00000000
0000fffe`793ad600  00000000`00000000
0000fffe`793ad608  00000000`00000000
0000fffe`793ad610  00000000`00000000
0000fffe`793ad618  00000000`00000000
0000fffe`793ad620  00000000`00000000
0000fffe`793ad628  00000000`00000000
0000fffe`793ad630  00000000`00000000
0000fffe`793ad638  00000000`00000000
0000fffe`793ad640  00000000`00000000
0000fffe`793ad648  00000000`00000000
0000fffe`793ad650  00000000`00000000
0000fffe`793ad658  00000000`00000000
0000fffe`793ad660  00000000`00000000
0000fffe`793ad668  00000000`00000000
0000fffe`793ad670  00000000`00000000
0000fffe`793ad678  00000000`00000000
0000fffe`793ad680  00000000`00000000
0000fffe`793ad688  00000000`00000000
0000fffe`793ad690  00000000`00000000
0000fffe`793ad698  00000000`00000000
0000fffe`793ad6a0  00000000`00000000
0000fffe`793ad6a8  00000000`00000000
0000fffe`793ad6b0  00000000`00000000
0000fffe`793ad6b8  00000000`00000000
0000fffe`793ad6c0  00000000`00000000
0000fffe`793ad6c8  00000000`00000000
0000fffe`793ad6d0  00000000`00000000
0000fffe`793ad6d8  00000000`00000000
0000fffe`793ad6e0  00000000`00000000
0000fffe`793ad6e8  00000000`00000000
0000fffe`793ad6f0  00000000`00000000
0000fffe`793ad6f8  00000000`00000000
0000fffe`793ad700  00000000`00000000
0000fffe`793ad708  00000000`00000000
0000fffe`793ad710  00000000`00000000
0000fffe`793ad718  00000000`00000000
0000fffe`793ad720  00000000`00000000
0000fffe`793ad728  00000000`00000000
0000fffe`793ad730  00000000`00000000
0000fffe`793ad738  00000000`00000000
```

```
0000fffe`793ad740  00000000`00000000
0000fffe`793ad748  00000000`00000000
0000fffe`793ad750  00000000`00000000
0000fffe`793ad758  00000000`00000000
0000fffe`793ad760  00000000`00000000
0000fffe`793ad768  00000000`00000000
0000fffe`793ad770  00000000`00000000
0000fffe`793ad778  00000000`00000000
0000fffe`793ad780  00000000`00000000
0000fffe`793ad788  00000000`00000000
0000fffe`793ad790  00000000`00000000
0000fffe`793ad798  00000000`00000000
0000fffe`793ad7a0  00000000`00000000
0000fffe`793ad7a8  00000000`00000000
0000fffe`793ad7b0  00000000`00000000
0000fffe`793ad7b8  00000000`00000000
0000fffe`793ad7c0  00000000`00000000
0000fffe`793ad7c8  00000000`00000000
0000fffe`793ad7d0  00000000`00000000
0000fffe`793ad7d8  00000000`00000000
0000fffe`793ad7e0  00000000`00000000
0000fffe`793ad7e8  00000000`00000000
0000fffe`793ad7f0  00000000`00000000
0000fffe`793ad7f8  00000000`00000000
0000fffe`793ad800  00000000`00000000
0000fffe`793ad808  00000000`00000000
0000fffe`793ad810  00000000`00000000
0000fffe`793ad818  00000000`00000000
0000fffe`793ad820  00000000`00000000
0000fffe`793ad828  00000000`00000000
0000fffe`793ad830  00000000`00000000
0000fffe`793ad838  00000000`00000000
0000fffe`793ad840  00000000`00000000
0000fffe`793ad848  00000000`00000020
0000fffe`793ad850  0000fffe`793ad8a0
0000fffe`793ad858  00000000`00414c08 App8!Unwind_RaiseException+0x158
0000fffe`793ad860  0000fffe`793add30
0000fffe`793ad868  0000fffe`793ae0f0
0000fffe`793ad870  0000fffe`74000b80
0000fffe`793ad878  0000fffe`793ad970
0000fffe`793ad880  00000000`00000000
0000fffe`793ad888  00000000`0000001e
0000fffe`793ad890  00000000`00011b1b
0000fffe`793ad898  0000fffe`793ae7f0
0000fffe`793ad8a0  0000fffe`793ae7f0
0000fffe`793ad8a8  00000000`004034c0 App8!Z5procHv+0x14
0000fffe`793ad8b0  0000fffe`74000b80
0000fffe`793ad8b8  00000000`00000001
0000fffe`793ad8c0  00000000`00000000
0000fffe`793ad8c8  00000000`00000001
0000fffe`793ad8d0  0000fffe`793af070
0000fffe`793ad8d8  00000000`00000000
0000fffe`793ad8e0  00000000`004e0000 App8!+0x18
0000fffe`793ad8e8  00000000`004035d8 App8!Z12thread_threePv
0000fffe`793ad8f0  00000000`00000000
0000fffe`793ad8f8  0000fffe`793af760
0000fffe`793ad900  00000000`32b706f0
0000fffe`793ad908  00000000`004f9540 App8!_default_pthread_attr
0000fffe`793ad910  00000000`00010000
0000fffe`793ad918  00000000`00810000
```

```
0000fffe`793ad920  00000000`00000000
0000fffe`793ad928  00000000`00000000
0000fffe`793ad930  00000000`00000000
0000fffe`793ad938  00000000`00000000
0000fffe`793ad940  00000000`00000000
0000fffe`793ad948  00000000`00000000
0000fffe`793ad950  00000000`00000000
0000fffe`793ad958  00000000`00000000
0000fffe`793ad960  00000000`00000000
0000fffe`793ad968  00000000`00000080
0000fffe`793ad970  0000fffe`793ad8b0
0000fffe`793ad978  0000fffe`793ad8b8
0000fffe`793ad980  0000fffe`793ad8c0
0000fffe`793ad988  0000fffe`793ad8c8
0000fffe`793ad990  00000000`00000000
0000fffe`793ad998  00000000`00000000
0000fffe`793ad9a0  00000000`00000000
0000fffe`793ad9a8  00000000`00000000
0000fffe`793ad9b0  00000000`00000000
0000fffe`793ad9b8  00000000`00000000
0000fffe`793ad9c0  00000000`00000000
0000fffe`793ad9c8  00000000`00000000
0000fffe`793ad9d0  00000000`00000000
0000fffe`793ad9d8  00000000`00000000
0000fffe`793ad9e0  00000000`00000000
0000fffe`793ad9e8  00000000`00000000
0000fffe`793ad9f0  00000000`00000000
0000fffe`793ad9f8  00000000`00000000
0000fffe`793ada00  00000000`00000000
0000fffe`793ada08  0000fffe`793ad8d0
0000fffe`793ada10  0000fffe`793ad8d8
0000fffe`793ada18  0000fffe`793ad8e0
0000fffe`793ada20  0000fffe`793ad8e8
0000fffe`793ada28  0000fffe`793ad8f0
0000fffe`793ada30  0000fffe`793ad8f8
0000fffe`793ada38  0000fffe`793ad900
0000fffe`793ada40  0000fffe`793ad908
0000fffe`793ada48  0000fffe`793ad910
0000fffe`793ada50  0000fffe`793ad918
0000fffe`793ada58  0000fffe`793ad8a0
0000fffe`793ada60  0000fffe`793ad8a8
0000fffe`793ada68  00000000`00000000
0000fffe`793ada70  00000000`00000000
0000fffe`793ada78  00000000`00000000
0000fffe`793ada80  00000000`00000000
0000fffe`793ada88  00000000`00000000
0000fffe`793ada90  00000000`00000000
0000fffe`793ada98  00000000`00000000
0000fffe`793adaa0  00000000`00000000
0000fffe`793adaa8  00000000`00000000
0000fffe`793adab0  00000000`00000000
0000fffe`793adab8  00000000`00000000
0000fffe`793adac0  00000000`00000000
0000fffe`793adac8  00000000`00000000
0000fffe`793adad0  00000000`00000000
0000fffe`793adad8  00000000`00000000
0000fffe`793adae0  00000000`00000000
0000fffe`793adae8  00000000`00000000
0000fffe`793adaf0  00000000`00000000
0000fffe`793adaf8  00000000`00000000
```

```
0000fffe`793adb00  00000000`00000000
0000fffe`793adb08  00000000`00000000
0000fffe`793adb10  00000000`00000000
0000fffe`793adb18  00000000`00000000
0000fffe`793adb20  00000000`00000000
0000fffe`793adb28  00000000`00000000
0000fffe`793adb30  00000000`00000000
0000fffe`793adb38  00000000`00000000
0000fffe`793adb40  00000000`00000000
0000fffe`793adb48  00000000`00000000
0000fffe`793adb50  00000000`00000000
0000fffe`793adb58  00000000`00000000
0000fffe`793adb60  00000000`00000000
0000fffe`793adb68  00000000`00000000
0000fffe`793adb70  00000000`00000000
0000fffe`793adb78  00000000`00000000
0000fffe`793adb80  00000000`00000000
0000fffe`793adb88  00000000`00000000
0000fffe`793adb90  00000000`00000000
0000fffe`793adb98  00000000`00000000
0000fffe`793adba0  00000000`00000000
0000fffe`793adba8  00000000`00000000
0000fffe`793adbb0  0000fffe`793ad920
0000fffe`793adbb8  0000fffe`793ad928
0000fffe`793adbc0  0000fffe`793ad930
0000fffe`793adbc8  0000fffe`793ad938
0000fffe`793adbd0  0000fffe`793ad940
0000fffe`793adbd8  0000fffe`793ad948
0000fffe`793adbe0  0000fffe`793ad950
0000fffe`793adbe8  0000fffe`793ad958
0000fffe`793adbf0  00000000`00000000
0000fffe`793adbf8  00000000`00000000
0000fffe`793adc00  00000000`00000000
0000fffe`793adc08  00000000`00000000
0000fffe`793adc10  00000000`00000000
0000fffe`793adc18  00000000`00000000
0000fffe`793adc20  00000000`00000000
0000fffe`793adc28  00000000`00000000
0000fffe`793adc30  00000000`00000000
0000fffe`793adc38  00000000`00000000
0000fffe`793adc40  00000000`00000000
0000fffe`793adc48  00000000`00000000
0000fffe`793adc50  00000000`00000000
0000fffe`793adc58  00000000`00000000
0000fffe`793adc60  00000000`00000000
0000fffe`793adc68  00000000`00000000
0000fffe`793adc70  00000000`00000000
0000fffe`793adc78  00000000`00000000
0000fffe`793adc80  0000fffe`793ae770
0000fffe`793adc88  00000000`00404d7c App8!_cxa_throw+0x90
0000fffe`793adc90  00000000`00000000
0000fffe`793adc98  00000000`00000000
0000fffe`793adca0  00000000`00000000
0000fffe`793adca8  00000000`00414ab0 App8!Unwind_RaiseException
0000fffe`793adcb0  40000000`00000000
0000fffe`793adcb8  00000000`00000000
0000fffe`793adcc0  00000000`00000000
0000fffe`793adcc8  00000000`00000000
0000fffe`793adcd0  00000000`00000000
0000fffe`793adcd8  00000000`00000000
```

```
0000fffe`793adce0  00000000`00000000
0000fffe`793adce8  00000000`00000000
0000fffe`793adcf0  00000000`00000000
0000fffe`793adcf8  00000000`00000000
0000fffe`793add00  00000000`00000000
0000fffe`793add08  00000000`00000000
0000fffe`793add10  00000000`00000000
0000fffe`793add18  00000000`00000000
0000fffe`793add20  00000000`00000000
0000fffe`793add28  00000000`00000000
0000fffe`793add30  0000fffe`793ad8b0
0000fffe`793add38  0000fffe`793ad8b8
0000fffe`793add40  0000fffe`793ad8c0
0000fffe`793add48  0000fffe`793ad8c8
0000fffe`793add50  00000000`00000000
0000fffe`793add58  00000000`00000000
0000fffe`793add60  00000000`00000000
0000fffe`793add68  00000000`00000000
0000fffe`793add70  00000000`00000000
0000fffe`793add78  00000000`00000000
0000fffe`793add80  00000000`00000000
0000fffe`793add88  00000000`00000000
0000fffe`793add90  00000000`00000000
0000fffe`793add98  00000000`00000000
0000fffe`793adda0  00000000`00000000
0000fffe`793adda8  00000000`00000000
0000fffe`793addb0  00000000`00000000
0000fffe`793addb8  00000000`00000000
0000fffe`793addc0  00000000`00000000
0000fffe`793addc8  0000fffe`793ae7b0
0000fffe`793addd0  0000fffe`793ae7b8
0000fffe`793addd8  0000fffe`793ae7c0
0000fffe`793adde0  0000fffe`793ad8e8
0000fffe`793adde8  0000fffe`793ad8f0
0000fffe`793addf0  0000fffe`793ad8f8
0000fffe`793addf8  0000fffe`793ad900
0000fffe`793ade00  0000fffe`793ad908
0000fffe`793ade08  0000fffe`793ad910
0000fffe`793ade10  0000fffe`793ad918
0000fffe`793ade18  0000fffe`793ae7e0
0000fffe`793ade20  0000fffe`793ae7e8
0000fffe`793ade28  0000fffe`793ad898
0000fffe`793ade30  00000000`00000000
0000fffe`793ade38  00000000`00000000
0000fffe`793ade40  00000000`00000000
0000fffe`793ade48  00000000`00000000
0000fffe`793ade50  00000000`00000000
0000fffe`793ade58  00000000`00000000
0000fffe`793ade60  00000000`00000000
0000fffe`793ade68  00000000`00000000
0000fffe`793ade70  00000000`00000000
0000fffe`793ade78  00000000`00000000
0000fffe`793ade80  00000000`00000000
0000fffe`793ade88  00000000`00000000
0000fffe`793ade90  00000000`00000000
0000fffe`793ade98  00000000`00000000
0000fffe`793adea0  00000000`00000000
0000fffe`793adea8  00000000`00000000
0000fffe`793adeb0  00000000`00000000
0000fffe`793adeb8  00000000`00000000
```

```
0000fffe`793adec0  00000000`00000000
0000fffe`793adec8  00000000`00000000
0000fffe`793aded0  00000000`00000000
0000fffe`793aded8  00000000`00000000
0000fffe`793adee0  00000000`00000000
0000fffe`793adee8  00000000`00000000
0000fffe`793adef0  00000000`00000000
0000fffe`793adef8  00000000`00000000
0000fffe`793adf00  00000000`00000000
0000fffe`793adf08  00000000`00000000
0000fffe`793adf10  00000000`00000000
0000fffe`793adf18  00000000`00000000
0000fffe`793adf20  00000000`00000000
0000fffe`793adf28  00000000`00000000
0000fffe`793adf30  00000000`00000000
0000fffe`793adf38  00000000`00000000
0000fffe`793adf40  00000000`00000000
0000fffe`793adf48  00000000`00000000
0000fffe`793adf50  00000000`00000000
0000fffe`793adf58  00000000`00000000
0000fffe`793adf60  00000000`00000000
0000fffe`793adf68  00000000`00000000
0000fffe`793adf70  0000fffe`793ad920
0000fffe`793adf78  0000fffe`793ad928
0000fffe`793adf80  0000fffe`793ad930
0000fffe`793adf88  0000fffe`793ad938
0000fffe`793adf90  0000fffe`793ad940
0000fffe`793adf98  0000fffe`793ad948
0000fffe`793adfa0  0000fffe`793ad950
0000fffe`793adfa8  0000fffe`793ad958
0000fffe`793adfb0  00000000`00000000
0000fffe`793adfb8  00000000`00000000
0000fffe`793adfc0  00000000`00000000
0000fffe`793adfc8  00000000`00000000
0000fffe`793adfd0  00000000`00000000
0000fffe`793adfd8  00000000`00000000
0000fffe`793adfe0  00000000`00000000
0000fffe`793adfe8  00000000`00000000
0000fffe`793adff0  00000000`00000000
0000fffe`793adff8  00000000`00000000
0000fffe`793ae000  00000000`00000000
0000fffe`793ae008  00000000`00000000
0000fffe`793ae010  00000000`00000000
0000fffe`793ae018  00000000`00000000
0000fffe`793ae020  00000000`00000000
0000fffe`793ae028  00000000`00000000
0000fffe`793ae030  00000000`00000000
0000fffe`793ae038  00000000`00000000
0000fffe`793ae040  0000fffe`793ae7f0
0000fffe`793ae048  00000000`004034c0 App8!Z5procHv+0x14
0000fffe`793ae050  00000000`004c57d8 App8!$d+0x1
0000fffe`793ae058  00000000`00000000
0000fffe`793ae060  00000000`00000000
0000fffe`793ae068  00000000`004034ac App8!Z5procHv
0000fffe`793ae070  40000000`00000000
0000fffe`793ae078  00000000`00000000
0000fffe`793ae080  00000000`00000000
0000fffe`793ae088  00000000`00000000
0000fffe`793ae090  00000000`00000000
0000fffe`793ae098  00000000`00000000
```

```
0000fffe`793ae0a0  00000000`00000000
0000fffe`793ae0a8  00000000`00000000
0000fffe`793ae0b0  00000000`00000000
0000fffe`793ae0b8  00000000`00000000
0000fffe`793ae0c0  00000000`00000000
0000fffe`793ae0c8  00000000`00000000
0000fffe`793ae0d0  00000000`00000000
0000fffe`793ae0d8  00000000`00000000
0000fffe`793ae0e0  00000000`00000000
0000fffe`793ae0e8  00000000`00000000
0000fffe`793ae0f0  00000000`00000000
0000fffe`793ae0f8  00000000`00000000
0000fffe`793ae100  00000000`00000000
0000fffe`793ae108  00000000`00000000
0000fffe`793ae110  00000000`00000000
0000fffe`793ae118  00000000`00000000
0000fffe`793ae120  00000000`00000000
0000fffe`793ae128  00000000`00000000
0000fffe`793ae130  00000000`00000000
0000fffe`793ae138  00000000`00000000
0000fffe`793ae140  00000000`00000000
0000fffe`793ae148  00000000`00000000
0000fffe`793ae150  00000000`00000000
0000fffe`793ae158  00000000`00000000
0000fffe`793ae160  00000000`00000000
0000fffe`793ae168  00000000`00000000
0000fffe`793ae170  00000000`00000000
0000fffe`793ae178  00000000`00000000
0000fffe`793ae180  00000000`00000000
0000fffe`793ae188  00000000`00000000
0000fffe`793ae190  00000000`00000000
0000fffe`793ae198  00000000`00000000
0000fffe`793ae1a0  00000000`00000000
0000fffe`793ae1a8  00000000`00000000
0000fffe`793ae1b0  00000000`00000000
0000fffe`793ae1b8  00000000`00000000
0000fffe`793ae1c0  00000000`00000000
0000fffe`793ae1c8  00000000`00000000
0000fffe`793ae1d0  00000000`00000000
0000fffe`793ae1d8  00000000`00000000
0000fffe`793ae1e0  00000000`00000000
0000fffe`793ae1e8  00000000`00000000
0000fffe`793ae1f0  00000000`00000000
0000fffe`793ae1f8  00000000`00000000
0000fffe`793ae200  00000000`00000000
0000fffe`793ae208  00000000`00000000
0000fffe`793ae210  00000000`00000000
0000fffe`793ae218  00000000`00000000
0000fffe`793ae220  ffffffff`fffffff0
0000fffe`793ae228  00000000`00000001
0000fffe`793ae230  00000000`00000000
0000fffe`793ae238  00000000`00000000
0000fffe`793ae240  00000000`00000000
0000fffe`793ae248  00000000`00000000
0000fffe`793ae250  00000000`00000000
0000fffe`793ae258  00000000`00000000
0000fffe`793ae260  00000000`00000000
0000fffe`793ae268  00000000`00000000
0000fffe`793ae270  00000000`00000000
0000fffe`793ae278  00000000`00000000
```

```
0000fffe`793ae280  00000000`00000000
0000fffe`793ae288  00000000`00000000
0000fffe`793ae290  00000000`00000000
0000fffe`793ae298  00000000`00000000
0000fffe`793ae2a0  00000000`00000000
0000fffe`793ae2a8  00000000`00000000
0000fffe`793ae2b0  00000000`00000000
0000fffe`793ae2b8  00000000`00000000
0000fffe`793ae2c0  ffffffff`ffffffe0
0000fffe`793ae2c8  00000000`00000001
0000fffe`793ae2d0  ffffffff`ffffffe8
0000fffe`793ae2d8  00000000`00000001
0000fffe`793ae2e0  00000000`00000000
0000fffe`793ae2e8  00000000`00000000
0000fffe`793ae2f0  00000000`00000000
0000fffe`793ae2f8  00000000`00000000
0000fffe`793ae300  00000000`00000000
0000fffe`793ae308  00000000`00000000
0000fffe`793ae310  00000000`00000000
0000fffe`793ae318  00000000`00000000
0000fffe`793ae320  00000000`00000000
0000fffe`793ae328  00000000`00000000
0000fffe`793ae330  00000000`00000000
0000fffe`793ae338  00000000`00000000
0000fffe`793ae340  00000000`00000000
0000fffe`793ae348  00000000`00000000
0000fffe`793ae350  00000000`00000000
0000fffe`793ae358  00000000`00000000
0000fffe`793ae360  00000000`00000000
0000fffe`793ae368  00000000`00000000
0000fffe`793ae370  00000000`00000000
0000fffe`793ae378  00000000`00000000
0000fffe`793ae380  00000000`00000000
0000fffe`793ae388  00000000`00000000
0000fffe`793ae390  00000000`00000000
0000fffe`793ae398  00000000`00000000
0000fffe`793ae3a0  00000000`00000000
0000fffe`793ae3a8  00000000`00000000
0000fffe`793ae3b0  00000000`00000000
0000fffe`793ae3b8  00000000`00000000
0000fffe`793ae3c0  00000000`00000000
0000fffe`793ae3c8  00000000`00000000
0000fffe`793ae3d0  00000000`00000000
0000fffe`793ae3d8  00000000`00000000
0000fffe`793ae3e0  00000000`00000000
0000fffe`793ae3e8  00000000`00000000
0000fffe`793ae3f0  00000000`00000000
0000fffe`793ae3f8  00000000`00000000
0000fffe`793ae400  00000000`00000000
0000fffe`793ae408  00000000`00000000
0000fffe`793ae410  00000000`00000000
0000fffe`793ae418  00000000`00000000
0000fffe`793ae420  00000000`00000000
0000fffe`793ae428  00000000`00000000
0000fffe`793ae430  00000000`00000000
0000fffe`793ae438  00000000`00000000
0000fffe`793ae440  00000000`00000000
0000fffe`793ae448  00000000`00000000
0000fffe`793ae450  00000000`00000000
0000fffe`793ae458  00000000`00000000
```

```
0000fffe`793ae460  00000000`00000000
0000fffe`793ae468  00000000`00000000
0000fffe`793ae470  00000000`00000000
0000fffe`793ae478  00000000`00000000
0000fffe`793ae480  00000000`00000000
0000fffe`793ae488  00000000`00000000
0000fffe`793ae490  0000fffe`793ae600
0000fffe`793ae498  00000000`00438e08 App8!sleep+0xe4
0000fffe`793ae4a0  00000000`ffffffff
0000fffe`793ae4a8  00000000`00010000
0000fffe`793ae4b0  00000000`00000000
0000fffe`793ae4b8  00000000`00000000
0000fffe`793ae4c0  00000000`00000000
0000fffe`793ae4c8  00000000`00000000
0000fffe`793ae4d0  00000000`00000000
0000fffe`793ae4d8  00000000`00000000
0000fffe`793ae4e0  00000000`00000000
0000fffe`793ae4e8  00000000`00000000
0000fffe`793ae4f0  00000000`00000000
0000fffe`793ae4f8  00000000`00000000
0000fffe`793ae500  00000000`00000000
0000fffe`793ae508  00000000`00000000
0000fffe`793ae510  00000000`00000000
0000fffe`793ae518  00000000`00000000
0000fffe`793ae520  00000000`00000000
0000fffe`793ae528  00000000`00000000
0000fffe`793ae530  00000000`00000000
0000fffe`793ae538  00000000`00000000
0000fffe`793ae540  00000000`00000000
0000fffe`793ae548  00000000`00000000
[...]
```

```
0:001> ub 00000000`004145cc
App8!Unwind_RaiseException_Phase2+0x70:
00000000`004145ac 71001c1f cmp         w0,#7
00000000`004145b0 54000120 beq         App8!Unwind_RaiseException_Phase2+0x98
(00000000`004145d4)
00000000`004145b4 7100201f cmp         w0,#8
00000000`004145b8 540000c1 bne         App8!Unwind_RaiseException_Phase2+0x94
(00000000`004145d0)
00000000`004145bc 35000195 cbnz        w21,App8!Unwind_RaiseException_Phase2+0xb0
(00000000`004145ec)
00000000`004145c0 aa1303e0 mov         x0,x19
00000000`004145c4 aa1403e1 mov         x1,x20
00000000`004145c8 97ffffb2 bl          App8!uw_update_context (00000000`00414490)
```

10. We close logging before exiting WinDbg Preview:

```
0:001> .logclose
Closing open log file 'C:\ALCDA2\A64\App8\App8.log'
```

Exercise A9

- **Goal:** Learn how to identify heap leaks
- **Patterns:** Memory Leak (Process Heap); Module Hint
- \ALCDA-Dumps\Exercise-A9-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A9-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A9-A64-WinDbg.pdf

Exercise A9 (x64, GDB)

Goal: Learn how to identify heap leaks.

Patterns: Memory Leak (Process Heap); Module Hint.

1. The application *App9* was found to consume more and more memory. Several core memory dumps were saved at different times with corresponding *pmap* logs. Load *App9.core.2.230* dump file and *App9* executable from the x64/App9 directory:

```
~/ALCDA2/x64/App9$ gdb -c App9.core.2.230 -se App9
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App9...done.
[New LWP 230]
[New LWP 231]
[New LWP 232]
[New LWP 233]
[New LWP 234]
[New LWP 235]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App9'.
#0  0x0000000000441ad0 in nanosleep ()
[Current thread is 1 (Thread 0x1778880 (LWP 230))]
```

8. Set logging to a file in case of lengthy output from some commands:

```
(gdb) set logging on App9.log
Copying output to App9.log.
```

2. Notice the size of the largest section and quit gdb:

```
(gdb) maintenance info sections
Exec file:
    `/home/coredump/ALCDA2/x64/App9/App9', file type elf64-x86-64.
 [0]     0x00400200->0x00400220 at 0x00000200: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]     0x00400220->0x00400244 at 0x00000220: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]     0x00400248->0x004004d0 at 0x00000248: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]     0x00401000->0x00401017 at 0x00001000: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
 [4]     0x00401018->0x004010f0 at 0x00001018: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
 [5]     0x004010f0->0x00493490 at 0x000010f0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]     0x00493490->0x00494037 at 0x00093490: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]     0x00494038->0x00494041 at 0x00094038: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [8]     0x00495000->0x004af73c at 0x00095000: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
```

```
 [9]     0x004af740->0x004bbbd0 at 0x000af740: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]     0x004bbbd0->0x004bbc7c at 0x000bbbd0: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]     0x004bd0b0->0x004bd0d8 at 0x000bc0b0: .tdata ALLOC LOAD DATA HAS_CONTENTS
 [12]     0x004bd0d8->0x004bd120 at 0x000bc0d8: .tbss ALLOC
 [13]     0x004bd0d8->0x004bd0e0 at 0x000bc0d8: .preinit_array ALLOC LOAD DATA HAS_CONTENTS
 [14]     0x004bd0e0->0x004bd0f0 at 0x000bc0e0: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [15]     0x004bd0f0->0x004bd100 at 0x000bc0f0: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [16]     0x004bd100->0x004bfef4 at 0x000bc100: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [17]     0x004bfef8->0x004c0000 at 0x000beef8: .got ALLOC LOAD DATA HAS_CONTENTS
 [18]     0x004c0000->0x004c00f0 at 0x000bf000: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [19]     0x004c0100->0x004c1c30 at 0x000bf100: .data ALLOC LOAD DATA HAS_CONTENTS
 [20]     0x004c1c30->0x004c1c90 at 0x000c0c30: __libc_subfreeres ALLOC LOAD DATA HAS_CONTENTS
 [21]     0x004c1ca0->0x004c2408 at 0x000c0ca0: __libc_IO_vtables ALLOC LOAD DATA HAS_CONTENTS
 [22]     0x004c2408->0x004c2410 at 0x000c1408: __libc_atexit ALLOC LOAD DATA HAS_CONTENTS
 [23]     0x004c2420->0x004c8528 at 0x000c1410: .bss ALLOC
 [24]     0x004c8528->0x004c8558 at 0x000c1410: __libc_freeres_ptrs ALLOC
 [25]     0x00000000->0x00000038 at 0x000c1410: .comment READONLY HAS_CONTENTS
 [26]     0x00000000->0x00000420 at 0x000c1450: .debug_aranges READONLY HAS_CONTENTS
 [27]     0x00000000->0x000372ad at 0x000c1870: .debug_info READONLY HAS_CONTENTS
 [28]     0x00000000->0x000057e8 at 0x000f8b1d: .debug_abbrev READONLY HAS_CONTENTS
 [29]     0x00000000->0x0000aa2b at 0x000fe305: .debug_line READONLY HAS_CONTENTS
 [30]     0x00000000->0x00004d08 at 0x00108d30: .debug_str READONLY HAS_CONTENTS
 [31]     0x00000000->0x0000d4b8 at 0x0010da38: .debug_loc READONLY HAS_CONTENTS
 [32]     0x00000000->0x000024c0 at 0x0011aef0: .debug_ranges READONLY HAS_CONTENTS
Core file:
    `/home/coredump/ALCDA2/x64/App9/App9.core.2.230', file type elf64-x86-64.
 [0]     0x00000000->0x00002ecc at 0x000004a0: note0 READONLY HAS_CONTENTS
 [1]     0x00000000->0x000000d8 at 0x000005c0: .reg/230 HAS_CONTENTS
 [2]     0x00000000->0x000000d8 at 0x000005c0: .reg HAS_CONTENTS
 [3]     0x00000000->0x00000200 at 0x000006b4: .reg2/230 HAS_CONTENTS
 [4]     0x00000000->0x00000200 at 0x000006b4: .reg2 HAS_CONTENTS
 [5]     0x00000000->0x00000340 at 0x000008c8: .reg-xstate/230 HAS_CONTENTS
 [6]     0x00000000->0x00000340 at 0x000008c8: .reg-xstate HAS_CONTENTS
 [7]     0x00000000->0x00000080 at 0x00000c1c: .note.linuxcore.siginfo/230 HAS_CONTENTS
 [8]     0x00000000->0x00000080 at 0x00000c1c: .note.linuxcore.siginfo HAS_CONTENTS
 [9]     0x00000000->0x000000d8 at 0x00000d20: .reg/231 HAS_CONTENTS
 [10]     0x00000000->0x00000200 at 0x00000e14: .reg2/231 HAS_CONTENTS
 [11]     0x00000000->0x00000340 at 0x00001028: .reg-xstate/231 HAS_CONTENTS
 [12]     0x00000000->0x00000080 at 0x0000137c: .note.linuxcore.siginfo/231 HAS_CONTENTS
 [13]     0x00000000->0x000000d8 at 0x00001480: .reg/232 HAS_CONTENTS
 [14]     0x00000000->0x00000200 at 0x00001574: .reg2/232 HAS_CONTENTS
 [15]     0x00000000->0x00000340 at 0x00001788: .reg-xstate/232 HAS_CONTENTS
 [16]     0x00000000->0x00000080 at 0x00001adc: .note.linuxcore.siginfo/232 HAS_CONTENTS
 [17]     0x00000000->0x000000d8 at 0x00001be0: .reg/233 HAS_CONTENTS
 [18]     0x00000000->0x00000200 at 0x00001cd4: .reg2/233 HAS_CONTENTS
 [19]     0x00000000->0x00000340 at 0x00001ee8: .reg-xstate/233 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
 [20]     0x00000000->0x00000080 at 0x0000223c: .note.linuxcore.siginfo/233 HAS_CONTENTS
 [21]     0x00000000->0x000000d8 at 0x00002340: .reg/234 HAS_CONTENTS
 [22]     0x00000000->0x00000200 at 0x00002434: .reg2/234 HAS_CONTENTS
 [23]     0x00000000->0x00000340 at 0x00002648: .reg-xstate/234 HAS_CONTENTS
 [24]     0x00000000->0x00000080 at 0x0000299c: .note.linuxcore.siginfo/234 HAS_CONTENTS
 [25]     0x00000000->0x000000d8 at 0x00002aa0: .reg/235 HAS_CONTENTS
 [26]     0x00000000->0x00000200 at 0x00002b94: .reg2/235 HAS_CONTENTS
 [27]     0x00000000->0x00000340 at 0x00002da8: .reg-xstate/235 HAS_CONTENTS
 [28]     0x00000000->0x00000080 at 0x000030fc: .note.linuxcore.siginfo/235 HAS_CONTENTS
 [29]     0x00000000->0x00000140 at 0x00003190: .auxv HAS_CONTENTS
 [30]     0x00000000->0x00000088 at 0x000032e4: .note.linuxcore.file/235 HAS_CONTENTS
 [31]     0x00000000->0x00000088 at 0x000032e4: .note.linuxcore.file HAS_CONTENTS
 [32]     0x00401000->0x00495000 at 0x0000336c: load1 ALLOC LOAD READONLY CODE HAS_CONTENTS
 [33]     0x004bd000->0x004c3000 at 0x0009736c: load2 ALLOC LOAD HAS_CONTENTS
 [34]     0x004c3000->0x004c9000 at 0x0009d36c: load3 ALLOC LOAD HAS_CONTENTS
 [35]     0x01778000->0x0179b000 at 0x000a336c: load4 ALLOC LOAD HAS_CONTENTS
 [36]     0x7f08dc000000->0x7f08dc227000 at 0x000c636c: load5 ALLOC LOAD HAS_CONTENTS
 [37]     0x7f08dc227000->0x7f08e0000000 at 0x002ed36c: load6 ALLOC LOAD READONLY HAS_CONTENTS
 [38]     0x7f08e4000000->0x7f08e8000000 at 0x040c636c: load7 ALLOC LOAD HAS_CONTENTS
 [39]     0x7f08eb528000->0x7f08eb529000 at 0x080c636c: load8 ALLOC LOAD READONLY HAS_CONTENTS
```

```
 [40]     0x7f08eb529000->0x7f08ebd29000 at 0x080c736c: load9 ALLOC LOAD HAS_CONTENTS
 [41]     0x7f08ebd29000->0x7f08ebd2a000 at 0x088c736c: load10 ALLOC LOAD READONLY HAS_CONTENTS
 [42]     0x7f08ebd2a000->0x7f08ec52a000 at 0x088c836c: load11 ALLOC LOAD HAS_CONTENTS
 [43]     0x7f08ec52a000->0x7f08ec52b000 at 0x090c836c: load12 ALLOC LOAD READONLY HAS_CONTENTS
 [44]     0x7f08ec52b000->0x7f08ecd2b000 at 0x090c936c: load13 ALLOC LOAD HAS_CONTENTS
 [45]     0x7f08ecd2b000->0x7f08ecd2c000 at 0x098c936c: load14 ALLOC LOAD READONLY HAS_CONTENTS
 [46]     0x7f08ecd2c000->0x7f08ed52c000 at 0x098ca36c: load15 ALLOC LOAD HAS_CONTENTS
 [47]     0x7f08ed52c000->0x7f08ed52d000 at 0x0a0ca36c: load16 ALLOC LOAD READONLY HAS_CONTENTS
 [48]     0x7f08ed52d000->0x7f08edd2d000 at 0x0a0cb36c: load17 ALLOC LOAD HAS_CONTENTS
 [49]     0x7ffe4333f000->0x7ffe43360000 at 0x0a8cb36c: load18 ALLOC LOAD HAS_CONTENTS
 [50]     0x7ffe43385000->0x7ffe43386000 at 0x0a8ec36c: load19 ALLOC LOAD READONLY CODE HAS_CONTENTS

(gdb) q
```

3. Load *App9.core.3.230* dump file and *App9* executable from x64/App9 directory:

```
~/ALCDA2/x64/App9$ gdb -c App9.core.3.230 -se App9
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App9...done.
[New LWP 230]
[New LWP 231]
[New LWP 232]
[New LWP 233]
[New LWP 234]
[New LWP 235]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App9'.
#0  0x0000000000441ad0 in nanosleep ()
[Current thread is 1 (Thread 0x1778880 (LWP 230))]
```

9. Set logging to a file in case of lengthy verbose output from some commands:

```
(gdb) set logging on App9.log
Copying output to App9.log.
```

4. Notice that another large section appeared after some time.

```
(gdb) maintenance info sections
Exec file:
    `/home/coredump/ALCDA2/x64/App9/App9', file type elf64-x86-64.
 [0]     0x00400200->0x00400220 at 0x00000200: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]     0x00400220->0x00400244 at 0x00000220: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]     0x00400248->0x004004d0 at 0x00000248: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]     0x00401000->0x00401017 at 0x00001000: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
 [4]     0x00401018->0x004010f0 at 0x00001018: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
 [5]     0x004010f0->0x00493490 at 0x000010f0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]     0x00493490->0x00494037 at 0x00093490: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]     0x00494038->0x00494041 at 0x00094038: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [8]     0x00495000->0x004af73c at 0x00095000: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
 [9]     0x004af740->0x004bbbd0 at 0x000af740: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]     0x004bbbd0->0x004bbc7c at 0x000bbbd0: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]     0x004bd0b0->0x004bd0d8 at 0x000bc0b0: .tdata ALLOC LOAD DATA HAS_CONTENTS
 [12]     0x004bd0d8->0x004bd120 at 0x000bc0d8: .tbss ALLOC
 [13]     0x004bd0d8->0x004bd0e0 at 0x000bc0d8: .preinit_array ALLOC LOAD DATA HAS_CONTENTS
 [14]     0x004bd0e0->0x004bd0f0 at 0x000bc0e0: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [15]     0x004bd0f0->0x004bd100 at 0x000bc0f0: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [16]     0x004bd100->0x004bfef4 at 0x000bc100: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [17]     0x004bfef8->0x004c0000 at 0x000beef8: .got ALLOC LOAD DATA HAS_CONTENTS
 [18]     0x004c0000->0x004c00f0 at 0x000bf000: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [19]     0x004c0100->0x004c1c30 at 0x000bf100: .data ALLOC LOAD DATA HAS_CONTENTS
 [20]     0x004c1c30->0x004c1c90 at 0x000c0c30: __libc_subfreeres ALLOC LOAD DATA HAS_CONTENTS
 [21]     0x004c1ca0->0x004c2408 at 0x000c0ca0: __libc_IO_vtables ALLOC LOAD DATA HAS_CONTENTS
 [22]     0x004c2408->0x004c2410 at 0x000c1408: __libc_atexit ALLOC LOAD DATA HAS_CONTENTS
 [23]     0x004c2420->0x004c8528 at 0x000c1410: .bss ALLOC
 [24]     0x004c8528->0x004c8558 at 0x000c1410: __libc_freeres_ptrs ALLOC
 [25]     0x00000000->0x00000038 at 0x000c1410: .comment READONLY HAS_CONTENTS
 [26]     0x00000000->0x00000420 at 0x000c1450: .debug_aranges READONLY HAS_CONTENTS
 [27]     0x00000000->0x000372ad at 0x000c1870: .debug_info READONLY HAS_CONTENTS
 [28]     0x00000000->0x000057e8 at 0x000f8b1d: .debug_abbrev READONLY HAS_CONTENTS
 [29]     0x00000000->0x0000aa2b at 0x000fe305: .debug_line READONLY HAS_CONTENTS
 [30]     0x00000000->0x00004d08 at 0x00108d30: .debug_str READONLY HAS_CONTENTS
 [31]     0x00000000->0x0000d4b8 at 0x0010da38: .debug_loc READONLY HAS_CONTENTS
 [32]     0x00000000->0x000024c0 at 0x0011aef0: .debug_ranges READONLY HAS_CONTENTS
Core file:
    `/home/coredump/ALCDA2/x64/App9/App9.core.3.230', file type elf64-x86-64.
 [0]     0x00000000->0x00002ecc at 0x000004a0: note0 READONLY HAS_CONTENTS
 [1]     0x00000000->0x000000d8 at 0x000005c0: .reg/230 HAS_CONTENTS
 [2]     0x00000000->0x000000d8 at 0x000005c0: .reg HAS_CONTENTS
 [3]     0x00000000->0x00000200 at 0x000006b4: .reg2/230 HAS_CONTENTS
 [4]     0x00000000->0x00000200 at 0x000006b4: .reg2 HAS_CONTENTS
 [5]     0x00000000->0x00000340 at 0x000008c8: .reg-xstate/230 HAS_CONTENTS
 [6]     0x00000000->0x00000340 at 0x000008c8: .reg-xstate HAS_CONTENTS
 [7]     0x00000000->0x00000080 at 0x00000c1c: .note.linuxcore.siginfo/230 HAS_CONTENTS
 [8]     0x00000000->0x00000080 at 0x00000c1c: .note.linuxcore.siginfo HAS_CONTENTS
 [9]     0x00000000->0x000000d8 at 0x00000d20: .reg/231 HAS_CONTENTS
 [10]     0x00000000->0x00000200 at 0x00000e14: .reg2/231 HAS_CONTENTS
 [11]     0x00000000->0x00000340 at 0x00001028: .reg-xstate/231 HAS_CONTENTS
 [12]     0x00000000->0x00000080 at 0x0000137c: .note.linuxcore.siginfo/231 HAS_CONTENTS
 [13]     0x00000000->0x000000d8 at 0x00001480: .reg/232 HAS_CONTENTS
 [14]     0x00000000->0x00000200 at 0x00001574: .reg2/232 HAS_CONTENTS
 [15]     0x00000000->0x00000340 at 0x00001788: .reg-xstate/232 HAS_CONTENTS
 [16]     0x00000000->0x00000080 at 0x00001adc: .note.linuxcore.siginfo/232 HAS_CONTENTS
 [17]     0x00000000->0x000000d8 at 0x00001be0: .reg/233 HAS_CONTENTS
 [18]     0x00000000->0x00000200 at 0x00001cd4: .reg2/233 HAS_CONTENTS
 [19]     0x00000000->0x00000340 at 0x00001ee8: .reg-xstate/233 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
 [20]     0x00000000->0x00000080 at 0x0000223c: .note.linuxcore.siginfo/233 HAS_CONTENTS
 [21]     0x00000000->0x000000d8 at 0x00002340: .reg/234 HAS_CONTENTS
 [22]     0x00000000->0x00000200 at 0x00002434: .reg2/234 HAS_CONTENTS
 [23]     0x00000000->0x00000340 at 0x00002648: .reg-xstate/234 HAS_CONTENTS
 [24]     0x00000000->0x00000080 at 0x0000299c: .note.linuxcore.siginfo/234 HAS_CONTENTS
```

```
[25]     0x00000000->0x000000d8 at 0x00002aa0: .reg/235 HAS_CONTENTS
[26]     0x00000000->0x00000200 at 0x00002b94: .reg2/235 HAS_CONTENTS
[27]     0x00000000->0x00000340 at 0x00002da8: .reg-xstate/235 HAS_CONTENTS
[28]     0x00000000->0x00000080 at 0x000030fc: .note.linuxcore.siginfo/235 HAS_CONTENTS
[29]     0x00000000->0x00000140 at 0x00003190: .auxv HAS_CONTENTS
[30]     0x00000000->0x00000088 at 0x000032e4: .note.linuxcore.file/235 HAS_CONTENTS
[31]     0x00000000->0x00000088 at 0x000032e4: .note.linuxcore.file HAS_CONTENTS
[32]     0x00401000->0x00495000 at 0x0000336c: load1 ALLOC LOAD READONLY CODE HAS_CONTENTS
[33]     0x004bd000->0x004c3000 at 0x0009736c: load2 ALLOC LOAD HAS_CONTENTS
[34]     0x004c3000->0x004c9000 at 0x0009d36c: load3 ALLOC LOAD HAS_CONTENTS
[35]     0x01778000->0x0179b000 at 0x000a336c: load4 ALLOC LOAD HAS_CONTENTS
[36]     0x7f08dc000000->0x7f08e0300000 at 0x000c636c: load5 ALLOC LOAD HAS_CONTENTS
[37]     0x7f08e0300000->0x7f08e4000000 at 0x043c636c: load6 ALLOC LOAD READONLY HAS_CONTENTS
[38]     0x7f08e4000000->0x7f08e8000000 at 0x080c636c: load7 ALLOC LOAD HAS_CONTENTS
[39]     0x7f08eb528000->0x7f08eb529000 at 0x0c0c636c: load8 ALLOC LOAD READONLY HAS_CONTENTS
[40]     0x7f08eb529000->0x7f08ebd29000 at 0x0c0c736c: load9 ALLOC LOAD HAS_CONTENTS
[41]     0x7f08ebd29000->0x7f08ebd2a000 at 0x0c8c736c: load10 ALLOC LOAD READONLY HAS_CONTENTS
[42]     0x7f08ebd2a000->0x7f08ec52a000 at 0x0c8c836c: load11 ALLOC LOAD HAS_CONTENTS
[43]     0x7f08ec52a000->0x7f08ec52b000 at 0x0d0c836c: load12 ALLOC LOAD READONLY HAS_CONTENTS
[44]     0x7f08ec52b000->0x7f08ecd2b000 at 0x0d0c936c: load13 ALLOC LOAD HAS_CONTENTS
[45]     0x7f08ecd2b000->0x7f08ecd2c000 at 0x0d8c936c: load14 ALLOC LOAD READONLY HAS_CONTENTS
[46]     0x7f08ecd2c000->0x7f08ed52c000 at 0x0d8ca36c: load15 ALLOC LOAD HAS_CONTENTS
[47]     0x7f08ed52c000->0x7f08ed52d000 at 0x0e0ca36c: load16 ALLOC LOAD READONLY HAS_CONTENTS
[48]     0x7f08ed52d000->0x7f08edd2d000 at 0x0e0cb36c: load17 ALLOC LOAD HAS_CONTENTS
[49]     0x7ffe4333f000->0x7ffe43360000 at 0x0e8cb36c: load18 ALLOC LOAD HAS_CONTENTS
[50]     0x7ffe43385000->0x7ffe43386000 at 0x0e8ec36c: load19 ALLOC LOAD READONLY CODE HAS_CONTENTS
```

5. Examine segment contents for any execution residue and hints (we choose some smaller address range from the section address range):

```
(gdb) x/1000a 0x7f08dc000000
0x7f08dc000000: 0x7f08e4000020  0x7f08e4000000
0x7f08dc000010: 0x4000000       0x4000000
0x7f08dc000020: 0x0     0x115
0x7f08dc000030: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc000040: 0x0     0x0
0x7f08dc000050: 0x401bad <procD>        0x0
0x7f08dc000060: 0x0     0x0
0x7f08dc000070: 0x0     0x0
0x7f08dc000080: 0x0     0x0
0x7f08dc000090: 0x0     0x0
0x7f08dc0000a0: 0x0     0x0
0x7f08dc0000b0: 0x0     0x0
0x7f08dc0000c0: 0x0     0x0
0x7f08dc0000d0: 0x0     0x0
0x7f08dc0000e0: 0x0     0x0
0x7f08dc0000f0: 0x0     0x0
0x7f08dc000100: 0x0     0x0
0x7f08dc000110: 0x0     0x0
0x7f08dc000120: 0x0     0x0
0x7f08dc000130: 0x0     0x115
0x7f08dc000140: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc000150: 0x0     0x0
0x7f08dc000160: 0x401bad <procD>        0x0
0x7f08dc000170: 0x0     0x0
0x7f08dc000180: 0x0     0x0
0x7f08dc000190: 0x0     0x0
0x7f08dc0001a0: 0x0     0x0
0x7f08dc0001b0: 0x0     0x0
0x7f08dc0001c0: 0x0     0x0
0x7f08dc0001d0: 0x0     0x0
0x7f08dc0001e0: 0x0     0x0
```

```
0x7f08dc0001f0: 0x0     0x0
0x7f08dc000200: 0x0     0x0
0x7f08dc000210: 0x0     0x0
0x7f08dc000220: 0x0     0x0
0x7f08dc000230: 0x0     0x0
0x7f08dc000240: 0x0     0x115
0x7f08dc000250: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc000260: 0x0     0x0
0x7f08dc000270: 0x401bad <procD>        0x0
0x7f08dc000280: 0x0     0x0
0x7f08dc000290: 0x0     0x0
0x7f08dc0002a0: 0x0     0x0
0x7f08dc0002b0: 0x0     0x0
0x7f08dc0002c0: 0x0     0x0
0x7f08dc0002d0: 0x0     0x0
0x7f08dc0002e0: 0x0     0x0
0x7f08dc0002f0: 0x0     0x0
0x7f08dc000300: 0x0     0x0
0x7f08dc000310: 0x0     0x0
0x7f08dc000320: 0x0     0x0
0x7f08dc000330: 0x0     0x0
0x7f08dc000340: 0x0     0x0
0x7f08dc000350: 0x0     0x115
0x7f08dc000360: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc000370: 0x0     0x0
0x7f08dc000380: 0x401bad <procD>        0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f08dc000390: 0x0     0x0
0x7f08dc0003a0: 0x0     0x0
0x7f08dc0003b0: 0x0     0x0
0x7f08dc0003c0: 0x0     0x0
0x7f08dc0003d0: 0x0     0x0
0x7f08dc0003e0: 0x0     0x0
0x7f08dc0003f0: 0x0     0x0
0x7f08dc000400: 0x0     0x0
0x7f08dc000410: 0x0     0x0
0x7f08dc000420: 0x0     0x0
0x7f08dc000430: 0x0     0x0
0x7f08dc000440: 0x0     0x0
0x7f08dc000450: 0x0     0x0
0x7f08dc000460: 0x0     0x115
0x7f08dc000470: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc000480: 0x0     0x0
0x7f08dc000490: 0x401bad <procD>        0x0
0x7f08dc0004a0: 0x0     0x0
0x7f08dc0004b0: 0x0     0x0
0x7f08dc0004c0: 0x0     0x0
0x7f08dc0004d0: 0x0     0x0
0x7f08dc0004e0: 0x0     0x0
0x7f08dc0004f0: 0x0     0x0
0x7f08dc000500: 0x0     0x0
0x7f08dc000510: 0x0     0x0
0x7f08dc000520: 0x0     0x0
0x7f08dc000530: 0x0     0x0
0x7f08dc000540: 0x0     0x0
0x7f08dc000550: 0x0     0x0
0x7f08dc000560: 0x0     0x0
0x7f08dc000570: 0x0     0x115
0x7f08dc000580: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc000590: 0x0     0x0
```

```
0x7f08dc0005a0: 0x401bad <procD>        0x0
0x7f08dc0005b0: 0x0     0x0
0x7f08dc0005c0: 0x0     0x0
0x7f08dc0005d0: 0x0     0x0
0x7f08dc0005e0: 0x0     0x0
0x7f08dc0005f0: 0x0     0x0
0x7f08dc000600: 0x0     0x0
0x7f08dc000610: 0x0     0x0
0x7f08dc000620: 0x0     0x0
0x7f08dc000630: 0x0     0x0
0x7f08dc000640: 0x0     0x0
0x7f08dc000650: 0x0     0x0
0x7f08dc000660: 0x0     0x0
0x7f08dc000670: 0x0     0x0
0x7f08dc000680: 0x0     0x115
0x7f08dc000690: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc0006a0: 0x0     0x0
0x7f08dc0006b0: 0x401bad <procD>        0x0
0x7f08dc0006c0: 0x0     0x0
0x7f08dc0006d0: 0x0     0x0
0x7f08dc0006e0: 0x0     0x0
0x7f08dc0006f0: 0x0     0x0
0x7f08dc000700: 0x0     0x0
0x7f08dc000710: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f08dc000720: 0x0     0x0
0x7f08dc000730: 0x0     0x0
0x7f08dc000740: 0x0     0x0
0x7f08dc000750: 0x0     0x0
0x7f08dc000760: 0x0     0x0
0x7f08dc000770: 0x0     0x0
0x7f08dc000780: 0x0     0x0
0x7f08dc000790: 0x0     0x115
0x7f08dc0007a0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc0007b0: 0x0     0x0
0x7f08dc0007c0: 0x401bad <procD>        0x0
0x7f08dc0007d0: 0x0     0x0
0x7f08dc0007e0: 0x0     0x0
0x7f08dc0007f0: 0x0     0x0
0x7f08dc000800: 0x0     0x0
0x7f08dc000810: 0x0     0x0
0x7f08dc000820: 0x0     0x0
0x7f08dc000830: 0x0     0x0
0x7f08dc000840: 0x0     0x0
0x7f08dc000850: 0x0     0x0
0x7f08dc000860: 0x0     0x0
0x7f08dc000870: 0x0     0x0
0x7f08dc000880: 0x0     0x0
0x7f08dc000890: 0x0     0x0
0x7f08dc0008a0: 0x0     0x115
0x7f08dc0008b0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc0008c0: 0x0     0x0
0x7f08dc0008d0: 0x401bad <procD>        0x0
0x7f08dc0008e0: 0x0     0x0
0x7f08dc0008f0: 0x0     0x0
0x7f08dc000900: 0x0     0x0
0x7f08dc000910: 0x0     0x0
0x7f08dc000920: 0x0     0x0
0x7f08dc000930: 0x0     0x0
0x7f08dc000940: 0x0     0x0
```

```
0x7f08dc000950: 0x0     0x0
0x7f08dc000960: 0x0     0x0
0x7f08dc000970: 0x0     0x0
0x7f08dc000980: 0x0     0x0
0x7f08dc000990: 0x0     0x0
0x7f08dc0009a0: 0x0     0x0
0x7f08dc0009b0: 0x0     0x115
0x7f08dc0009c0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc0009d0: 0x0     0x0
0x7f08dc0009e0: 0x401bad <procD>        0x0
0x7f08dc0009f0: 0x0     0x0
0x7f08dc000a00: 0x0     0x0
0x7f08dc000a10: 0x0     0x0
0x7f08dc000a20: 0x0     0x0
0x7f08dc000a30: 0x0     0x0
0x7f08dc000a40: 0x0     0x0
0x7f08dc000a50: 0x0     0x0
0x7f08dc000a60: 0x0     0x0
0x7f08dc000a70: 0x0     0x0
0x7f08dc000a80: 0x0     0x0
0x7f08dc000a90: 0x0     0x0
0x7f08dc000aa0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f08dc000ab0: 0x0     0x0
0x7f08dc000ac0: 0x0     0x115
0x7f08dc000ad0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc000ae0: 0x0     0x0
0x7f08dc000af0: 0x401bad <procD>        0x0
0x7f08dc000b00: 0x0     0x0
0x7f08dc000b10: 0x0     0x0
0x7f08dc000b20: 0x0     0x0
0x7f08dc000b30: 0x0     0x0
0x7f08dc000b40: 0x0     0x0
0x7f08dc000b50: 0x0     0x0
0x7f08dc000b60: 0x0     0x0
0x7f08dc000b70: 0x0     0x0
0x7f08dc000b80: 0x0     0x0
0x7f08dc000b90: 0x0     0x0
0x7f08dc000ba0: 0x0     0x0
0x7f08dc000bb0: 0x0     0x0
0x7f08dc000bc0: 0x0     0x0
0x7f08dc000bd0: 0x0     0x115
0x7f08dc000be0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc000bf0: 0x0     0x0
0x7f08dc000c00: 0x401bad <procD>        0x0
0x7f08dc000c10: 0x0     0x0
0x7f08dc000c20: 0x0     0x0
0x7f08dc000c30: 0x0     0x0
0x7f08dc000c40: 0x0     0x0
0x7f08dc000c50: 0x0     0x0
0x7f08dc000c60: 0x0     0x0
0x7f08dc000c70: 0x0     0x0
0x7f08dc000c80: 0x0     0x0
0x7f08dc000c90: 0x0     0x0
0x7f08dc000ca0: 0x0     0x0
0x7f08dc000cb0: 0x0     0x0
0x7f08dc000cc0: 0x0     0x0
0x7f08dc000cd0: 0x0     0x0
0x7f08dc000ce0: 0x0     0x115
0x7f08dc000cf0: 0x657461636f6c6c61      0x79726f6d656d2064
```

```
0x7f08dc000d00: 0x0     0x0
0x7f08dc000d10: 0x401bad <procD>        0x0
0x7f08dc000d20: 0x0     0x0
0x7f08dc000d30: 0x0     0x0
0x7f08dc000d40: 0x0     0x0
0x7f08dc000d50: 0x0     0x0
0x7f08dc000d60: 0x0     0x0
0x7f08dc000d70: 0x0     0x0
0x7f08dc000d80: 0x0     0x0
0x7f08dc000d90: 0x0     0x0
0x7f08dc000da0: 0x0     0x0
0x7f08dc000db0: 0x0     0x0
0x7f08dc000dc0: 0x0     0x0
0x7f08dc000dd0: 0x0     0x0
0x7f08dc000de0: 0x0     0x0
0x7f08dc000df0: 0x0     0x115
0x7f08dc000e00: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc000e10: 0x0     0x0
0x7f08dc000e20: 0x401bad <procD>        0x0
0x7f08dc000e30: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f08dc000e40: 0x0     0x0
0x7f08dc000e50: 0x0     0x0
0x7f08dc000e60: 0x0     0x0
0x7f08dc000e70: 0x0     0x0
0x7f08dc000e80: 0x0     0x0
0x7f08dc000e90: 0x0     0x0
0x7f08dc000ea0: 0x0     0x0
0x7f08dc000eb0: 0x0     0x0
0x7f08dc000ec0: 0x0     0x0
0x7f08dc000ed0: 0x0     0x0
0x7f08dc000ee0: 0x0     0x0
0x7f08dc000ef0: 0x0     0x0
0x7f08dc000f00: 0x0     0x115
0x7f08dc000f10: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc000f20: 0x0     0x0
0x7f08dc000f30: 0x401bad <procD>        0x0
0x7f08dc000f40: 0x0     0x0
0x7f08dc000f50: 0x0     0x0
0x7f08dc000f60: 0x0     0x0
0x7f08dc000f70: 0x0     0x0
0x7f08dc000f80: 0x0     0x0
0x7f08dc000f90: 0x0     0x0
0x7f08dc000fa0: 0x0     0x0
0x7f08dc000fb0: 0x0     0x0
0x7f08dc000fc0: 0x0     0x0
0x7f08dc000fd0: 0x0     0x0
0x7f08dc000fe0: 0x0     0x0
0x7f08dc000ff0: 0x0     0x0
0x7f08dc001000: 0x0     0x0
0x7f08dc001010: 0x0     0x115
0x7f08dc001020: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001030: 0x0     0x0
0x7f08dc001040: 0x401bad <procD>        0x0
0x7f08dc001050: 0x0     0x0
0x7f08dc001060: 0x0     0x0
0x7f08dc001070: 0x0     0x0
0x7f08dc001080: 0x0     0x0
0x7f08dc001090: 0x0     0x0
0x7f08dc0010a0: 0x0     0x0
```

```
0x7f08dc0010b0: 0x0     0x0
0x7f08dc0010c0: 0x0     0x0
0x7f08dc0010d0: 0x0     0x0
0x7f08dc0010e0: 0x0     0x0
0x7f08dc0010f0: 0x0     0x0
0x7f08dc001100: 0x0     0x0
0x7f08dc001110: 0x0     0x0
0x7f08dc001120: 0x0     0x115
0x7f08dc001130: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001140: 0x0     0x0
0x7f08dc001150: 0x401bad <procD>        0x0
0x7f08dc001160: 0x0     0x0
0x7f08dc001170: 0x0     0x0
0x7f08dc001180: 0x0     0x0
0x7f08dc001190: 0x0     0x0
0x7f08dc0011a0: 0x0     0x0
0x7f08dc0011b0: 0x0     0x0
0x7f08dc0011c0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f08dc0011d0: 0x0     0x0
0x7f08dc0011e0: 0x0     0x0
0x7f08dc0011f0: 0x0     0x0
0x7f08dc001200: 0x0     0x0
0x7f08dc001210: 0x0     0x0
0x7f08dc001220: 0x0     0x0
0x7f08dc001230: 0x0     0x115
0x7f08dc001240: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001250: 0x0     0x0
0x7f08dc001260: 0x401bad <procD>        0x0
0x7f08dc001270: 0x0     0x0
0x7f08dc001280: 0x0     0x0
0x7f08dc001290: 0x0     0x0
0x7f08dc0012a0: 0x0     0x0
0x7f08dc0012b0: 0x0     0x0
0x7f08dc0012c0: 0x0     0x0
0x7f08dc0012d0: 0x0     0x0
0x7f08dc0012e0: 0x0     0x0
0x7f08dc0012f0: 0x0     0x0
0x7f08dc001300: 0x0     0x0
0x7f08dc001310: 0x0     0x0
0x7f08dc001320: 0x0     0x0
0x7f08dc001330: 0x0     0x0
0x7f08dc001340: 0x0     0x115
0x7f08dc001350: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001360: 0x0     0x0
0x7f08dc001370: 0x401bad <procD>        0x0
0x7f08dc001380: 0x0     0x0
0x7f08dc001390: 0x0     0x0
0x7f08dc0013a0: 0x0     0x0
0x7f08dc0013b0: 0x0     0x0
0x7f08dc0013c0: 0x0     0x0
0x7f08dc0013d0: 0x0     0x0
0x7f08dc0013e0: 0x0     0x0
0x7f08dc0013f0: 0x0     0x0
0x7f08dc001400: 0x0     0x0
0x7f08dc001410: 0x0     0x0
0x7f08dc001420: 0x0     0x0
0x7f08dc001430: 0x0     0x0
0x7f08dc001440: 0x0     0x0
0x7f08dc001450: 0x0     0x115
```

```
0x7f08dc001460: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001470: 0x0     0x0
0x7f08dc001480: 0x401bad <procD>        0x0
0x7f08dc001490: 0x0     0x0
0x7f08dc0014a0: 0x0     0x0
0x7f08dc0014b0: 0x0     0x0
0x7f08dc0014c0: 0x0     0x0
0x7f08dc0014d0: 0x0     0x0
0x7f08dc0014e0: 0x0     0x0
0x7f08dc0014f0: 0x0     0x0
0x7f08dc001500: 0x0     0x0
0x7f08dc001510: 0x0     0x0
0x7f08dc001520: 0x0     0x0
0x7f08dc001530: 0x0     0x0
0x7f08dc001540: 0x0     0x0
0x7f08dc001550: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f08dc001560: 0x0     0x115
0x7f08dc001570: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001580: 0x0     0x0
0x7f08dc001590: 0x401bad <procD>        0x0
0x7f08dc0015a0: 0x0     0x0
0x7f08dc0015b0: 0x0     0x0
0x7f08dc0015c0: 0x0     0x0
0x7f08dc0015d0: 0x0     0x0
0x7f08dc0015e0: 0x0     0x0
0x7f08dc0015f0: 0x0     0x0
0x7f08dc001600: 0x0     0x0
0x7f08dc001610: 0x0     0x0
0x7f08dc001620: 0x0     0x0
0x7f08dc001630: 0x0     0x0
0x7f08dc001640: 0x0     0x0
0x7f08dc001650: 0x0     0x0
0x7f08dc001660: 0x0     0x0
0x7f08dc001670: 0x0     0x115
0x7f08dc001680: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001690: 0x0     0x0
0x7f08dc0016a0: 0x401bad <procD>        0x0
0x7f08dc0016b0: 0x0     0x0
0x7f08dc0016c0: 0x0     0x0
0x7f08dc0016d0: 0x0     0x0
0x7f08dc0016e0: 0x0     0x0
0x7f08dc0016f0: 0x0     0x0
0x7f08dc001700: 0x0     0x0
0x7f08dc001710: 0x0     0x0
0x7f08dc001720: 0x0     0x0
0x7f08dc001730: 0x0     0x0
0x7f08dc001740: 0x0     0x0
0x7f08dc001750: 0x0     0x0
0x7f08dc001760: 0x0     0x0
0x7f08dc001770: 0x0     0x0
0x7f08dc001780: 0x0     0x115
0x7f08dc001790: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc0017a0: 0x0     0x0
0x7f08dc0017b0: 0x401bad <procD>        0x0
0x7f08dc0017c0: 0x0     0x0
0x7f08dc0017d0: 0x0     0x0
0x7f08dc0017e0: 0x0     0x0
0x7f08dc0017f0: 0x0     0x0
0x7f08dc001800: 0x0     0x0
```

```
0x7f08dc001810: 0x0     0x0
0x7f08dc001820: 0x0     0x0
0x7f08dc001830: 0x0     0x0
0x7f08dc001840: 0x0     0x0
0x7f08dc001850: 0x0     0x0
0x7f08dc001860: 0x0     0x0
0x7f08dc001870: 0x0     0x0
0x7f08dc001880: 0x0     0x0
0x7f08dc001890: 0x0     0x115
0x7f08dc0018a0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc0018b0: 0x0     0x0
0x7f08dc0018c0: 0x401bad <procD>        0x0
0x7f08dc0018d0: 0x0     0x0
0x7f08dc0018e0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f08dc0018f0: 0x0     0x0
0x7f08dc001900: 0x0     0x0
0x7f08dc001910: 0x0     0x0
0x7f08dc001920: 0x0     0x0
0x7f08dc001930: 0x0     0x0
0x7f08dc001940: 0x0     0x0
0x7f08dc001950: 0x0     0x0
0x7f08dc001960: 0x0     0x0
0x7f08dc001970: 0x0     0x0
0x7f08dc001980: 0x0     0x0
0x7f08dc001990: 0x0     0x0
0x7f08dc0019a0: 0x0     0x115
0x7f08dc0019b0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc0019c0: 0x0     0x0
0x7f08dc0019d0: 0x401bad <procD>        0x0
0x7f08dc0019e0: 0x0     0x0
0x7f08dc0019f0: 0x0     0x0
0x7f08dc001a00: 0x0     0x0
0x7f08dc001a10: 0x0     0x0
0x7f08dc001a20: 0x0     0x0
0x7f08dc001a30: 0x0     0x0
0x7f08dc001a40: 0x0     0x0
0x7f08dc001a50: 0x0     0x0
0x7f08dc001a60: 0x0     0x0
0x7f08dc001a70: 0x0     0x0
0x7f08dc001a80: 0x0     0x0
0x7f08dc001a90: 0x0     0x0
0x7f08dc001aa0: 0x0     0x0
0x7f08dc001ab0: 0x0     0x115
0x7f08dc001ac0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001ad0: 0x0     0x0
0x7f08dc001ae0: 0x401bad <procD>        0x0
0x7f08dc001af0: 0x0     0x0
0x7f08dc001b00: 0x0     0x0
0x7f08dc001b10: 0x0     0x0
0x7f08dc001b20: 0x0     0x0
0x7f08dc001b30: 0x0     0x0
0x7f08dc001b40: 0x0     0x0
0x7f08dc001b50: 0x0     0x0
0x7f08dc001b60: 0x0     0x0
0x7f08dc001b70: 0x0     0x0
0x7f08dc001b80: 0x0     0x0
0x7f08dc001b90: 0x0     0x0
0x7f08dc001ba0: 0x0     0x0
0x7f08dc001bb0: 0x0     0x0
```

```
0x7f08dc001bc0: 0x0     0x115
0x7f08dc001bd0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001be0: 0x0     0x0
0x7f08dc001bf0: 0x401bad <procD>        0x0
0x7f08dc001c00: 0x0     0x0
0x7f08dc001c10: 0x0     0x0
0x7f08dc001c20: 0x0     0x0
0x7f08dc001c30: 0x0     0x0
0x7f08dc001c40: 0x0     0x0
0x7f08dc001c50: 0x0     0x0
0x7f08dc001c60: 0x0     0x0
0x7f08dc001c70: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0x7f08dc001c80: 0x0     0x0
0x7f08dc001c90: 0x0     0x0
0x7f08dc001ca0: 0x0     0x0
0x7f08dc001cb0: 0x0     0x0
0x7f08dc001cc0: 0x0     0x0
0x7f08dc001cd0: 0x0     0x115
0x7f08dc001ce0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001cf0: 0x0     0x0
0x7f08dc001d00: 0x401bad <procD>        0x0
0x7f08dc001d10: 0x0     0x0
0x7f08dc001d20: 0x0     0x0
0x7f08dc001d30: 0x0     0x0
0x7f08dc001d40: 0x0     0x0
0x7f08dc001d50: 0x0     0x0
0x7f08dc001d60: 0x0     0x0
0x7f08dc001d70: 0x0     0x0
0x7f08dc001d80: 0x0     0x0
0x7f08dc001d90: 0x0     0x0
0x7f08dc001da0: 0x0     0x0
0x7f08dc001db0: 0x0     0x0
0x7f08dc001dc0: 0x0     0x0
0x7f08dc001dd0: 0x0     0x0
0x7f08dc001de0: 0x0     0x115
0x7f08dc001df0: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001e00: 0x0     0x0
0x7f08dc001e10: 0x401bad <procD>        0x0
0x7f08dc001e20: 0x0     0x0
0x7f08dc001e30: 0x0     0x0
0x7f08dc001e40: 0x0     0x0
0x7f08dc001e50: 0x0     0x0
0x7f08dc001e60: 0x0     0x0
0x7f08dc001e70: 0x0     0x0
0x7f08dc001e80: 0x0     0x0
0x7f08dc001e90: 0x0     0x0
0x7f08dc001ea0: 0x0     0x0
0x7f08dc001eb0: 0x0     0x0
0x7f08dc001ec0: 0x0     0x0
0x7f08dc001ed0: 0x0     0x0
0x7f08dc001ee0: 0x0     0x0
0x7f08dc001ef0: 0x0     0x115
0x7f08dc001f00: 0x657461636f6c6c61      0x79726f6d656d2064
0x7f08dc001f10: 0x0     0x0
0x7f08dc001f20: 0x401bad <procD>        0x0
0x7f08dc001f30: 0x0     0x0

(gdb) x/s 0x7f08dc001f00
0x7f08dc001f00: "allocated memory"
```

6. Compare pmap logs *App9.pmap.1.230*, *App9.pmap.2.230*, and *App9.pmap.3.230* (the first one was saved before the leak started, and the other two correspond to core dumps we looked at):

```
230:   ./App9
0000000000400000      4K r---- App9
0000000000401000    592K r-x-- App9
0000000000495000    156K r---- App9
00000000004bd000     24K rw--- App9
00000000004c3000     24K rw---   [ anon ]
0000000001778000    140K rw---   [ anon ]
00007f08e4000000   1332K rw---   [ anon ]
00007f08e414d000  64204K -----   [ anon ]
00007f08eb528000      4K -----   [ anon ]
00007f08eb529000   8192K rw---   [ anon ]
00007f08ebd29000      4K -----   [ anon ]
00007f08ebd2a000   8192K rw---   [ anon ]
00007f08ec52a000      4K -----   [ anon ]
00007f08ec52b000   8192K rw---   [ anon ]
00007f08ecd2b000      4K -----   [ anon ]
00007f08ecd2c000   8192K rw---   [ anon ]
00007f08ed52c000      4K -----   [ anon ]
00007f08ed52d000   8192K rw---   [ anon ]
00007ffe4333f000    132K rw---   [ stack ]
00007ffe43381000     16K r----   [ anon ]
00007ffe43385000      4K r-x--   [ anon ]
 total           107608K

230:   ./App9
0000000000400000      4K r---- App9
0000000000401000    592K r-x-- App9
0000000000495000    156K r---- App9
00000000004bd000     24K rw--- App9
00000000004c3000     24K rw---   [ anon ]
0000000001778000    140K rw---   [ anon ]
00007f08dc000000   2204K rw---   [ anon ]
00007f08dc227000  63332K -----   [ anon ]
00007f08e4000000  65536K rw---   [ anon ]
00007f08eb528000      4K -----   [ anon ]
00007f08eb529000   8192K rw---   [ anon ]
00007f08ebd29000      4K -----   [ anon ]
00007f08ebd2a000   8192K rw---   [ anon ]
00007f08ec52a000      4K -----   [ anon ]
00007f08ec52b000   8192K rw---   [ anon ]
00007f08ecd2b000      4K -----   [ anon ]
00007f08ecd2c000   8192K rw---   [ anon ]
00007f08ed52c000      4K -----   [ anon ]
00007f08ed52d000   8192K rw---   [ anon ]
00007ffe4333f000    132K rw---   [ stack ]
00007ffe43381000     16K r----   [ anon ]
00007ffe43385000      4K r-x--   [ anon ]
 total           173144K

230:   ./App9
0000000000400000      4K r---- App9
0000000000401000    592K r-x-- App9
0000000000495000    156K r---- App9
00000000004bd000     24K rw--- App9
00000000004c3000     24K rw---   [ anon ]
0000000001778000    140K rw---   [ anon ]
00007f08dc000000  68608K rw---   [ anon ]
```

```
00007f08e0300000  62464K -----   [ anon ]
00007f08e4000000  65536K rw---   [ anon ]
00007f08eb528000      4K -----   [ anon ]
00007f08eb529000   8192K rw---   [ anon ]
00007f08ebd29000      4K -----   [ anon ]
00007f08ebd2a000   8192K rw---   [ anon ]
00007f08ec52a000      4K -----   [ anon ]
00007f08ec52b000   8192K rw---   [ anon ]
00007f08ecd2b000      4K -----   [ anon ]
00007f08ecd2c000   8192K rw---   [ anon ]
00007f08ed52c000      4K -----   [ anon ]
00007f08ed52d000   8192K rw---   [ anon ]
00007ffe4333f000    132K rw---   [ stack ]
00007ffe43381000     16K r----   [ anon ]
00007ffe43385000      4K r-x--   [ anon ]
 total           238680K
```

Exercise A9 (A64, GDB)

Goal: Learn how to identify heap leaks.

Patterns: Memory Leak (Process Heap); Module Hint.

1. The application *App9* was found to consume more and more memory. Several core memory dumps were saved at different times with corresponding *pmap* logs. Load *App9.core.2.12057* dump file and *App9* executable from the A64/App9 directory:

```
~/ALCDA2/A64/App9$ gdb -c App9.core.2.12057 -se App9
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App9...
(No debugging symbols found in App9)

warning: Can't open file /home/opc/ALCDA2/App9/App9 during file-backed mapping note processing
[New LWP 12058]
[New LWP 12059]
[New LWP 12060]
[New LWP 12061]
[New LWP 12062]
[New LWP 12057]
Core was generated by `./App9'.
#0  0x000000000040ca84 in nanosleep ()
[Current thread is 1 (LWP 12058)]
```

2. Set logging to a file in case of lengthy output from some commands:

```
(gdb) set logging file App9.log

(gdb) set logging enabled on
Copying output to App9.log.
Copying debug output to App9.log.

(gdb) set style enabled off
```

3. Notice the size of the largest section and quit GDB:

```
(gdb) maintenance info sections
Exec file: `/home/ubuntu/ALCDA2/A64/App9/App9', file type elf64-littleaarch64.
 [0]      0x00400190->0x004001b0 at 0x00000190: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]      0x004001b0->0x004001d4 at 0x000001b0: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]      0x004001d8->0x00400250 at 0x000001d8: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]      0x00400250->0x00400264 at 0x00000250: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
```

```
 [4]      0x00400270->0x004002c0 at 0x00000270: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
 [5]      0x004002c0->0x00487158 at 0x000002c0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]      0x00487158->0x00488e28 at 0x00087158: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]      0x00488e28->0x00489278 at 0x00088e28: __libc_thread_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [8]      0x00489278->0x00489288 at 0x00089278: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [9]      0x00489290->0x004a178d at 0x00089290: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]     0x004a178d->0x004a178e at 0x000a178d: .stapsdt.base ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]     0x004a1790->0x004a1ec8 at 0x000a1790: __libc_IO_vtables ALLOC LOAD READONLY DATA HAS_CONTENTS
 [12]     0x004a1ec8->0x004a1f30 at 0x000a1ec8: __libc_subfreeres ALLOC LOAD READONLY DATA HAS_CONTENTS
 [13]     0x004a1f30->0x004a1f38 at 0x000a1f30: __libc_atexit ALLOC LOAD READONLY DATA HAS_CONTENTS
 [14]     0x004a1f38->0x004a1f48 at 0x000a1f38: __libc_thread_subfreeres ALLOC LOAD READONLY DATA HAS_CONTENTS
 [15]     0x004a1f48->0x004b05ec at 0x000a1f48: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [16]     0x004b05ec->0x004b07a9 at 0x000b05ec: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [17]     0x004cfb20->0x004cfb48 at 0x000bfb20: .tdata ALLOC LOAD DATA HAS_CONTENTS
 [18]     0x004cfb48->0x004cfb98 at 0x000bfb48: .tbss ALLOC
 [19]     0x004cfb48->0x004cfb50 at 0x000bfb48: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [20]     0x004cfb50->0x004cfb60 at 0x000bfb50: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [21]     0x004cfb60->0x004cfb68 at 0x000bfb60: .jcr ALLOC LOAD DATA HAS_CONTENTS
 [22]     0x004cfb68->0x004cff24 at 0x000bfb68: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [23]     0x004cff28->0x004cffe8 at 0x000bff28: .got ALLOC LOAD DATA HAS_CONTENTS
 [24]     0x004cffe8->0x004d0028 at 0x000bffe8: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [25]     0x004d0030->0x004d1580 at 0x000c0030: .data ALLOC LOAD DATA HAS_CONTENTS
 [26]     0x004d1580->0x004d8050 at 0x000c1580: .bss ALLOC
 [27]     0x004d8050->0x004d8088 at 0x000c1580: __libc_freeres_ptrs ALLOC
 [28]     0x00000000->0x00000031 at 0x000c1580: .comment READONLY HAS_CONTENTS
 [29]     0x00000000->0x00001cb0 at 0x000c15b4: .note.stapsdt READONLY HAS_CONTENTS
Core file: `/home/ubuntu/ALCDA2/A64/App9/App9.core.2.12057', file type elf64-littleaarch64.
 [0]      0x00000000->0x00001c94 at 0x00000468: note0 READONLY HAS_CONTENTS
 [1]      0x00000000->0x00000110 at 0x00000588: .reg/12058 HAS_CONTENTS
 [2]      0x00000000->0x00000110 at 0x00000588: .reg HAS_CONTENTS
 [3]      0x00000000->0x00000210 at 0x000006b4: .reg2/12058 HAS_CONTENTS
 [4]      0x00000000->0x00000210 at 0x000006b4: .reg2 HAS_CONTENTS
 [5]      0x00000000->0x00000080 at 0x000008d8: .note.linuxcore.siginfo/12058 HAS_CONTENTS
 [6]      0x00000000->0x00000080 at 0x000008d8: .note.linuxcore.siginfo HAS_CONTENTS
 [7]      0x00000000->0x00000110 at 0x000009dc: .reg/12059 HAS_CONTENTS
 [8]      0x00000000->0x00000210 at 0x00000b08: .reg2/12059 HAS_CONTENTS
 [9]      0x00000000->0x00000080 at 0x00000d2c: .note.linuxcore.siginfo/12059 HAS_CONTENTS
 [10]     0x00000000->0x00000110 at 0x00000e30: .reg/12060 HAS_CONTENTS
 [11]     0x00000000->0x00000210 at 0x00000f5c: .reg2/12060 HAS_CONTENTS
 [12]     0x00000000->0x00000080 at 0x00001180: .note.linuxcore.siginfo/12060 HAS_CONTENTS
 [13]     0x00000000->0x00000110 at 0x00001284: .reg/12061 HAS_CONTENTS
 [14]     0x00000000->0x00000210 at 0x000013b0: .reg2/12061 HAS_CONTENTS
 [15]     0x00000000->0x00000080 at 0x000015d4: .note.linuxcore.siginfo/12061 HAS_CONTENTS
 [16]     0x00000000->0x00000110 at 0x000016d8: .reg/12062 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
 [17]     0x00000000->0x00000210 at 0x00001804: .reg2/12062 HAS_CONTENTS
 [18]     0x00000000->0x00000080 at 0x00001a28: .note.linuxcore.siginfo/12062 HAS_CONTENTS
 [19]     0x00000000->0x00000110 at 0x00001b2c: .reg/12057 HAS_CONTENTS
 [20]     0x00000000->0x00000210 at 0x00001c58: .reg2/12057 HAS_CONTENTS
 [21]     0x00000000->0x00000080 at 0x00001e7c: .note.linuxcore.siginfo/12057 HAS_CONTENTS
 [22]     0x00000000->0x00000160 at 0x00001f10: .auxv HAS_CONTENTS
 [23]     0x00000000->0x00000076 at 0x00002084: .note.linuxcore.file/12057 HAS_CONTENTS
 [24]     0x00000000->0x00000076 at 0x00002084: .note.linuxcore.file HAS_CONTENTS
 [25]     0x00400000->0x004c0000 at 0x000020fc: load1 ALLOC LOAD READONLY CODE HAS_CONTENTS
 [26]     0x004c0000->0x004e0000 at 0x000c20fc: load2 ALLOC LOAD HAS_CONTENTS
 [27]     0x2f860000->0x2f8a0000 at 0x000e20fc: load3 ALLOC LOAD HAS_CONTENTS
 [28]     0xfffce8000000->0xfffce8230000 at 0x001220fc: load4 ALLOC LOAD HAS_CONTENTS
 [29]     0xfffce8230000->0xfffcec000000 at 0x003520fc: load5 ALLOC LOAD READONLY HAS_CONTENTS
 [30]     0xffffcf0000000->0xffffcf4000000 at 0x041220fc: load6 ALLOC LOAD HAS_CONTENTS
 [31]     0xffffcf7400000->0xffffcf7410000 at 0x081220fc: load7 ALLOC LOAD READONLY HAS_CONTENTS
 [32]     0xffffcf7410000->0xffffcf7c10000 at 0x081320fc: load8 ALLOC LOAD HAS_CONTENTS
 [33]     0xffffcf7c10000->0xffffcf7c20000 at 0x089320fc: load9 ALLOC LOAD READONLY HAS_CONTENTS
 [34]     0xffffcf7c20000->0xffffcf8420000 at 0x089420fc: load10 ALLOC LOAD HAS_CONTENTS
 [35]     0xffffcf8420000->0xffffcf8430000 at 0x091420fc: load11 ALLOC LOAD READONLY HAS_CONTENTS
 [36]     0xffffcf8430000->0xffffcf8c30000 at 0x091520fc: load12 ALLOC LOAD HAS_CONTENTS
 [37]     0xffffcf8c30000->0xffffcf8c40000 at 0x099520fc: load13 ALLOC LOAD READONLY HAS_CONTENTS
 [38]     0xffffcf8c40000->0xffffcf9440000 at 0x099620fc: load14 ALLOC LOAD HAS_CONTENTS
 [39]     0xffffcf9440000->0xffffcf9450000 at 0x0a1620fc: load15 ALLOC LOAD READONLY HAS_CONTENTS
 [40]     0xffffcf9450000->0xffffcf9c50000 at 0x0a1720fc: load16 ALLOC LOAD HAS_CONTENTS
 [41]     0xffffcf9c60000->0xffffcf9c70000 at 0x0a9720fc: load17 ALLOC LOAD READONLY CODE HAS_CONTENTS
 [42]     0xffffc2f60000->0xffffc2f90000 at 0x0a9820fc: load18 ALLOC LOAD HAS_CONTENTS

(gdb) q
```

4. Load *App9.core.3.12057* dump file and *App9* executable from A64/App9 directory:

```
~/ALCDA2/A64/App9$ gdb -c App9.core.3.12057 -se App9
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App9...
(No debugging symbols found in App9)

warning: Can't open file /home/opc/ALCDA2/App9/App9 during file-backed mapping note processing
[New LWP 12058]
[New LWP 12059]
[New LWP 12060]
[New LWP 12061]
[New LWP 12062]
[New LWP 12057]
Core was generated by `./App9'.
#0  0x000000000040ca84 in nanosleep ()
[Current thread is 1 (LWP 12058)]
```

5. Set logging to a file in case of lengthy verbose output from some commands:

```
(gdb) set logging file App9.log

(gdb) set logging enabled on
Copying output to App9.log.
Copying debug output to App9.log.

(gdb) set style enabled off

(gdb) show logging
logging debugredirect:  off: Debug output will go to both the screen and the log file.
logging enabled:  off: Logging is disabled.
logging file:  The current logfile is "gdb.txt".
logging overwrite:  off: Logging appends to the log file.
logging redirect:  off: Output will go to both the screen and the log file.
```

6. Notice that another large section appeared after some time.

```
(gdb) maintenance info sections
Exec file: `/home/ubuntu/ALCDA2/A64/App9/App9', file type elf64-littleaarch64.
 [0]      0x00400190->0x004001b0 at 0x00000190: .note.ABI-tag ALLOC LOAD READONLY DATA HAS_CONTENTS
 [1]      0x004001b0->0x004001d4 at 0x000001b0: .note.gnu.build-id ALLOC LOAD READONLY DATA HAS_CONTENTS
 [2]      0x004001d8->0x00400250 at 0x000001d8: .rela.plt ALLOC LOAD READONLY DATA HAS_CONTENTS
 [3]      0x00400250->0x00400264 at 0x00000250: .init ALLOC LOAD READONLY CODE HAS_CONTENTS
 [4]      0x00400270->0x004002c0 at 0x00000270: .plt ALLOC LOAD READONLY CODE HAS_CONTENTS
 [5]      0x004002c0->0x00487158 at 0x000002c0: .text ALLOC LOAD READONLY CODE HAS_CONTENTS
 [6]      0x00487158->0x00488e28 at 0x00087158: __libc_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
 [7]      0x00488e28->0x00489278 at 0x00088e28: __libc_thread_freeres_fn ALLOC LOAD READONLY CODE HAS_CONTENTS
```

```
 [8]      0x00489278->0x00489288 at 0x00089278: .fini ALLOC LOAD READONLY CODE HAS_CONTENTS
 [9]      0x00489290->0x004a178d at 0x00089290: .rodata ALLOC LOAD READONLY DATA HAS_CONTENTS
 [10]     0x004a178d->0x004a178e at 0x000a178d: .stapsdt.base ALLOC LOAD READONLY DATA HAS_CONTENTS
 [11]     0x004a1790->0x004a1ec8 at 0x000a1790: __libc_IO_vtables ALLOC LOAD READONLY DATA HAS_CONTENTS
 [12]     0x004a1ec8->0x004a1f30 at 0x000a1ec8: __libc_subfreeres ALLOC LOAD READONLY DATA HAS_CONTENTS
 [13]     0x004a1f30->0x004a1f38 at 0x000a1f30: __libc_atexit ALLOC LOAD READONLY DATA HAS_CONTENTS
 [14]     0x004a1f38->0x004a1f48 at 0x000a1f38: __libc_thread_subfreeres ALLOC LOAD READONLY DATA HAS_CONTENTS
 [15]     0x004a1f48->0x004b05ec at 0x000a1f48: .eh_frame ALLOC LOAD READONLY DATA HAS_CONTENTS
 [16]     0x004b05ec->0x004b07a9 at 0x000b05ec: .gcc_except_table ALLOC LOAD READONLY DATA HAS_CONTENTS
 [17]     0x004cfb20->0x004cfb48 at 0x000bfb20: .tdata ALLOC LOAD DATA HAS_CONTENTS
 [18]     0x004cfb48->0x004cfb98 at 0x000bfb48: .tbss ALLOC
 [19]     0x004cfb48->0x004cfb50 at 0x000bfb48: .init_array ALLOC LOAD DATA HAS_CONTENTS
 [20]     0x004cfb50->0x004cfb60 at 0x000bfb50: .fini_array ALLOC LOAD DATA HAS_CONTENTS
 [21]     0x004cfb60->0x004cfb68 at 0x000bfb60: .jcr ALLOC LOAD DATA HAS_CONTENTS
 [22]     0x004cfb68->0x004cff24 at 0x000bfb68: .data.rel.ro ALLOC LOAD DATA HAS_CONTENTS
 [23]     0x004cff28->0x004cffe8 at 0x000bff28: .got ALLOC LOAD DATA HAS_CONTENTS
 [24]     0x004cffe8->0x004d0028 at 0x000bffe8: .got.plt ALLOC LOAD DATA HAS_CONTENTS
 [25]     0x004d0030->0x004d1580 at 0x000c0030: .data ALLOC LOAD DATA HAS_CONTENTS
 [26]     0x004d1580->0x004d8050 at 0x000c1580: .bss ALLOC
 [27]     0x004d8050->0x004d8088 at 0x000c1580: __libc_freeres_ptrs ALLOC
 [28]     0x00000000->0x00000031 at 0x000c1580: .comment READONLY HAS_CONTENTS
 [29]     0x00000000->0x00001cb0 at 0x000c15b4: .note.stapsdt READONLY HAS_CONTENTS
Core file: `/home/ubuntu/ALCDA2/A64/App9/App9.core.3.12057', file type elf64-littleaarch64.
 [0]      0x00000000->0x00001c94 at 0x00000468: note0 READONLY HAS_CONTENTS
 [1]      0x00000000->0x00000110 at 0x00000588: .reg/12058 HAS_CONTENTS
 [2]      0x00000000->0x00000110 at 0x00000588: .reg HAS_CONTENTS
 [3]      0x00000000->0x00000210 at 0x000006b4: .reg2/12058 HAS_CONTENTS
 [4]      0x00000000->0x00000210 at 0x000006b4: .reg2 HAS_CONTENTS
 [5]      0x00000000->0x00000080 at 0x000008d8: .note.linuxcore.siginfo/12058 HAS_CONTENTS
 [6]      0x00000000->0x00000080 at 0x000008d8: .note.linuxcore.siginfo HAS_CONTENTS
 [7]      0x00000000->0x00000110 at 0x000009dc: .reg/12059 HAS_CONTENTS
 [8]      0x00000000->0x00000210 at 0x00000b08: .reg2/12059 HAS_CONTENTS
 [9]      0x00000000->0x00000080 at 0x00000d2c: .note.linuxcore.siginfo/12059 HAS_CONTENTS
 [10]     0x00000000->0x00000110 at 0x00000e30: .reg/12060 HAS_CONTENTS
 [11]     0x00000000->0x00000210 at 0x00000f5c: .reg2/12060 HAS_CONTENTS
 [12]     0x00000000->0x00000080 at 0x00001180: .note.linuxcore.siginfo/12060 HAS_CONTENTS
 [13]     0x00000000->0x00000110 at 0x00001284: .reg/12061 HAS_CONTENTS
 [14]     0x00000000->0x00000210 at 0x000013b0: .reg2/12061 HAS_CONTENTS
 [15]     0x00000000->0x00000080 at 0x000015d4: .note.linuxcore.siginfo/12061 HAS_CONTENTS
 [16]     0x00000000->0x00000110 at 0x000016d8: .reg/12062 HAS_CONTENTS
--Type <RET> for more, q to quit, c to continue without paging--
 [17]     0x00000000->0x00000210 at 0x00001804: .reg2/12062 HAS_CONTENTS
 [18]     0x00000000->0x00000080 at 0x00001a28: .note.linuxcore.siginfo/12062 HAS_CONTENTS
 [19]     0x00000000->0x00000110 at 0x00001b2c: .reg/12057 HAS_CONTENTS
 [20]     0x00000000->0x00000210 at 0x00001c58: .reg2/12057 HAS_CONTENTS
 [21]     0x00000000->0x00000080 at 0x00001e7c: .note.linuxcore.siginfo/12057 HAS_CONTENTS
 [22]     0x00000000->0x00000160 at 0x00001f10: .auxv HAS_CONTENTS
 [23]     0x00000000->0x00000076 at 0x00002084: .note.linuxcore.file/12057 HAS_CONTENTS
 [24]     0x00000000->0x00000076 at 0x00002084: .note.linuxcore.file HAS_CONTENTS
 [25]     0x00400000->0x004c0000 at 0x000020fc: load1 ALLOC LOAD READONLY CODE HAS_CONTENTS
 [26]     0x004c0000->0x004e0000 at 0x000c20fc: load2 ALLOC LOAD HAS_CONTENTS
 [27]     0x2f860000->0x2f8a0000 at 0x000e20fc: load3 ALLOC LOAD HAS_CONTENTS
 [28]     0xfffce8000000->0xfffcec300000 at 0x001220fc: load4 ALLOC LOAD HAS_CONTENTS
 [29]     0xfffcec300000->0xfffcf0000000 at 0x044220fc: load5 ALLOC LOAD READONLY HAS_CONTENTS
 [30]     0xfffcf0000000->0xfffcf4000000 at 0x081220fc: load6 ALLOC LOAD HAS_CONTENTS
 [31]     0xfffcf7400000->0xfffcf7410000 at 0x0c1220fc: load7 ALLOC LOAD READONLY HAS_CONTENTS
 [32]     0xfffcf7410000->0xfffcf7c10000 at 0x0c1320fc: load8 ALLOC LOAD HAS_CONTENTS
 [33]     0xfffcf7c10000->0xfffcf7c20000 at 0x0c9320fc: load9 ALLOC LOAD READONLY HAS_CONTENTS
 [34]     0xfffcf7c20000->0xfffcf8420000 at 0x0c9420fc: load10 ALLOC LOAD HAS_CONTENTS
 [35]     0xfffcf8420000->0xfffcf8430000 at 0x0d1420fc: load11 ALLOC LOAD READONLY HAS_CONTENTS
 [36]     0xfffcf8430000->0xfffcf8c30000 at 0x0d1520fc: load12 ALLOC LOAD HAS_CONTENTS
 [37]     0xfffcf8c30000->0xfffcf8c40000 at 0x0d9520fc: load13 ALLOC LOAD READONLY HAS_CONTENTS
 [38]     0xfffcf8c40000->0xfffcf9440000 at 0x0d9620fc: load14 ALLOC LOAD HAS_CONTENTS
 [39]     0xfffcf9440000->0xfffcf9450000 at 0x0e1620fc: load15 ALLOC LOAD READONLY HAS_CONTENTS
 [40]     0xfffcf9450000->0xfffcf9c50000 at 0x0e1720fc: load16 ALLOC LOAD HAS_CONTENTS
 [41]     0xfffcf9c60000->0xfffcf9c70000 at 0x0e9720fc: load17 ALLOC LOAD READONLY CODE HAS_CONTENTS
 [42]     0xffffc2f60000->0xffffc2f90000 at 0x0e9820fc: load18 ALLOC LOAD HAS_CONTENTS
```

7. Examine segment contents for any execution residue and hints (we choose some smaller address range from the section address range):

```
(gdb) x/1000a 0xfffce8000000
0xfffce8000000: 0xfffcf0000020  0xfffcf0000000
0xfffce8000010: 0x4000000       0x4000000
0xfffce8000020: 0x0     0x115
0xfffce8000030: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8000040: 0x0     0x0
0xfffce8000050: 0x4031e8 <procD>        0x0
0xfffce8000060: 0x0     0x0
0xfffce8000070: 0x0     0x0
0xfffce8000080: 0x0     0x0
0xfffce8000090: 0x0     0x0
0xfffce80000a0: 0x0     0x0
0xfffce80000b0: 0x0     0x0
0xfffce80000c0: 0x0     0x0
0xfffce80000d0: 0x0     0x0
0xfffce80000e0: 0x0     0x0
0xfffce80000f0: 0x0     0x0
0xfffce8000100: 0x0     0x0
0xfffce8000110: 0x0     0x0
0xfffce8000120: 0x0     0x0
0xfffce8000130: 0x0     0x115
0xfffce8000140: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8000150: 0x0     0x0
0xfffce8000160: 0x4031e8 <procD>        0x0
0xfffce8000170: 0x0     0x0
0xfffce8000180: 0x0     0x0
0xfffce8000190: 0x0     0x0
0xfffce80001a0: 0x0     0x0
0xfffce80001b0: 0x0     0x0
0xfffce80001c0: 0x0     0x0
0xfffce80001d0: 0x0     0x0
0xfffce80001e0: 0x0     0x0
0xfffce80001f0: 0x0     0x0
0xfffce8000200: 0x0     0x0
0xfffce8000210: 0x0     0x0
0xfffce8000220: 0x0     0x0
0xfffce8000230: 0x0     0x0
0xfffce8000240: 0x0     0x115
0xfffce8000250: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8000260: 0x0     0x0
0xfffce8000270: 0x4031e8 <procD>        0x0
0xfffce8000280: 0x0     0x0
0xfffce8000290: 0x0     0x0
0xfffce80002a0: 0x0     0x0
0xfffce80002b0: 0x0     0x0
0xfffce80002c0: 0x0     0x0
0xfffce80002d0: 0x0     0x0
0xfffce80002e0: 0x0     0x0
0xfffce80002f0: 0x0     0x0
0xfffce8000300: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffce8000310: 0x0     0x0
0xfffce8000320: 0x0     0x0
0xfffce8000330: 0x0     0x0
0xfffce8000340: 0x0     0x0
0xfffce8000350: 0x0     0x115
0xfffce8000360: 0x657461636f6c6c61      0x79726f6d656d2064
```

```
0xfffce8000370: 0x0     0x0
0xfffce8000380: 0x4031e8 <procD>        0x0
0xfffce8000390: 0x0     0x0
0xfffce80003a0: 0x0     0x0
0xfffce80003b0: 0x0     0x0
0xfffce80003c0: 0x0     0x0
0xfffce80003d0: 0x0     0x0
0xfffce80003e0: 0x0     0x0
0xfffce80003f0: 0x0     0x0
0xfffce8000400: 0x0     0x0
0xfffce8000410: 0x0     0x0
0xfffce8000420: 0x0     0x0
0xfffce8000430: 0x0     0x0
0xfffce8000440: 0x0     0x0
0xfffce8000450: 0x0     0x0
0xfffce8000460: 0x0     0x115
0xfffce8000470: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8000480: 0x0     0x0
0xfffce8000490: 0x4031e8 <procD>        0x0
0xfffce80004a0: 0x0     0x0
0xfffce80004b0: 0x0     0x0
0xfffce80004c0: 0x0     0x0
0xfffce80004d0: 0x0     0x0
0xfffce80004e0: 0x0     0x0
0xfffce80004f0: 0x0     0x0
0xfffce8000500: 0x0     0x0
0xfffce8000510: 0x0     0x0
0xfffce8000520: 0x0     0x0
0xfffce8000530: 0x0     0x0
0xfffce8000540: 0x0     0x0
0xfffce8000550: 0x0     0x0
0xfffce8000560: 0x0     0x0
0xfffce8000570: 0x0     0x115
0xfffce8000580: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8000590: 0x0     0x0
0xfffce80005a0: 0x4031e8 <procD>        0x0
0xfffce80005b0: 0x0     0x0
0xfffce80005c0: 0x0     0x0
0xfffce80005d0: 0x0     0x0
0xfffce80005e0: 0x0     0x0
0xfffce80005f0: 0x0     0x0
0xfffce8000600: 0x0     0x0
0xfffce8000610: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffce8000620: 0x0     0x0
0xfffce8000630: 0x0     0x0
0xfffce8000640: 0x0     0x0
0xfffce8000650: 0x0     0x0
0xfffce8000660: 0x0     0x0
0xfffce8000670: 0x0     0x0
0xfffce8000680: 0x0     0x115
0xfffce8000690: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce80006a0: 0x0     0x0
0xfffce80006b0: 0x4031e8 <procD>        0x0
0xfffce80006c0: 0x0     0x0
0xfffce80006d0: 0x0     0x0
0xfffce80006e0: 0x0     0x0
0xfffce80006f0: 0x0     0x0
0xfffce8000700: 0x0     0x0
0xfffce8000710: 0x0     0x0
```

```
0xfffce8000720: 0x0     0x0
0xfffce8000730: 0x0     0x0
0xfffce8000740: 0x0     0x0
0xfffce8000750: 0x0     0x0
0xfffce8000760: 0x0     0x0
0xfffce8000770: 0x0     0x0
0xfffce8000780: 0x0     0x0
0xfffce8000790: 0x0     0x115
0xfffce80007a0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce80007b0: 0x0     0x0
0xfffce80007c0: 0x4031e8 <procD>        0x0
0xfffce80007d0: 0x0     0x0
0xfffce80007e0: 0x0     0x0
0xfffce80007f0: 0x0     0x0
0xfffce8000800: 0x0     0x0
0xfffce8000810: 0x0     0x0
0xfffce8000820: 0x0     0x0
0xfffce8000830: 0x0     0x0
0xfffce8000840: 0x0     0x0
0xfffce8000850: 0x0     0x0
0xfffce8000860: 0x0     0x0
0xfffce8000870: 0x0     0x0
0xfffce8000880: 0x0     0x0
0xfffce8000890: 0x0     0x0
0xfffce80008a0: 0x0     0x115
0xfffce80008b0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce80008c0: 0x0     0x0
0xfffce80008d0: 0x4031e8 <procD>        0x0
0xfffce80008e0: 0x0     0x0
0xfffce80008f0: 0x0     0x0
0xfffce8000900: 0x0     0x0
0xfffce8000910: 0x0     0x0
0xfffce8000920: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffce8000930: 0x0     0x0
0xfffce8000940: 0x0     0x0
0xfffce8000950: 0x0     0x0
0xfffce8000960: 0x0     0x0
0xfffce8000970: 0x0     0x0
0xfffce8000980: 0x0     0x0
0xfffce8000990: 0x0     0x0
0xfffce80009a0: 0x0     0x0
0xfffce80009b0: 0x0     0x115
0xfffce80009c0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce80009d0: 0x0     0x0
0xfffce80009e0: 0x4031e8 <procD>        0x0
0xfffce80009f0: 0x0     0x0
0xfffce8000a00: 0x0     0x0
0xfffce8000a10: 0x0     0x0
0xfffce8000a20: 0x0     0x0
0xfffce8000a30: 0x0     0x0
0xfffce8000a40: 0x0     0x0
0xfffce8000a50: 0x0     0x0
0xfffce8000a60: 0x0     0x0
0xfffce8000a70: 0x0     0x0
0xfffce8000a80: 0x0     0x0
0xfffce8000a90: 0x0     0x0
0xfffce8000aa0: 0x0     0x0
0xfffce8000ab0: 0x0     0x0
0xfffce8000ac0: 0x0     0x115
```

```
0xfffce8000ad0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8000ae0: 0x0     0x0
0xfffce8000af0: 0x4031e8 <procD>        0x0
0xfffce8000b00: 0x0     0x0
0xfffce8000b10: 0x0     0x0
0xfffce8000b20: 0x0     0x0
0xfffce8000b30: 0x0     0x0
0xfffce8000b40: 0x0     0x0
0xfffce8000b50: 0x0     0x0
0xfffce8000b60: 0x0     0x0
0xfffce8000b70: 0x0     0x0
0xfffce8000b80: 0x0     0x0
0xfffce8000b90: 0x0     0x0
0xfffce8000ba0: 0x0     0x0
0xfffce8000bb0: 0x0     0x0
0xfffce8000bc0: 0x0     0x0
0xfffce8000bd0: 0x0     0x115
0xfffce8000be0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8000bf0: 0x0     0x0
0xfffce8000c00: 0x4031e8 <procD>        0x0
0xfffce8000c10: 0x0     0x0
0xfffce8000c20: 0x0     0x0
0xfffce8000c30: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffce8000c40: 0x0     0x0
0xfffce8000c50: 0x0     0x0
0xfffce8000c60: 0x0     0x0
0xfffce8000c70: 0x0     0x0
0xfffce8000c80: 0x0     0x0
0xfffce8000c90: 0x0     0x0
0xfffce8000ca0: 0x0     0x0
0xfffce8000cb0: 0x0     0x0
0xfffce8000cc0: 0x0     0x0
0xfffce8000cd0: 0x0     0x0
0xfffce8000ce0: 0x0     0x115
0xfffce8000cf0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8000d00: 0x0     0x0
0xfffce8000d10: 0x4031e8 <procD>        0x0
0xfffce8000d20: 0x0     0x0
0xfffce8000d30: 0x0     0x0
0xfffce8000d40: 0x0     0x0
0xfffce8000d50: 0x0     0x0
0xfffce8000d60: 0x0     0x0
0xfffce8000d70: 0x0     0x0
0xfffce8000d80: 0x0     0x0
0xfffce8000d90: 0x0     0x0
0xfffce8000da0: 0x0     0x0
0xfffce8000db0: 0x0     0x0
0xfffce8000dc0: 0x0     0x0
0xfffce8000dd0: 0x0     0x0
0xfffce8000de0: 0x0     0x0
0xfffce8000df0: 0x0     0x115
0xfffce8000e00: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8000e10: 0x0     0x0
0xfffce8000e20: 0x4031e8 <procD>        0x0
0xfffce8000e30: 0x0     0x0
0xfffce8000e40: 0x0     0x0
0xfffce8000e50: 0x0     0x0
0xfffce8000e60: 0x0     0x0
0xfffce8000e70: 0x0     0x0
```

```
0xfffce8000e80: 0x0     0x0
0xfffce8000e90: 0x0     0x0
0xfffce8000ea0: 0x0     0x0
0xfffce8000eb0: 0x0     0x0
0xfffce8000ec0: 0x0     0x0
0xfffce8000ed0: 0x0     0x0
0xfffce8000ee0: 0x0     0x0
0xfffce8000ef0: 0x0     0x0
0xfffce8000f00: 0x0     0x115
0xfffce8000f10: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8000f20: 0x0     0x0
0xfffce8000f30: 0x4031e8 <procD>        0x0
0xfffce8000f40: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffce8000f50: 0x0     0x0
0xfffce8000f60: 0x0     0x0
0xfffce8000f70: 0x0     0x0
0xfffce8000f80: 0x0     0x0
0xfffce8000f90: 0x0     0x0
0xfffce8000fa0: 0x0     0x0
0xfffce8000fb0: 0x0     0x0
0xfffce8000fc0: 0x0     0x0
0xfffce8000fd0: 0x0     0x0
0xfffce8000fe0: 0x0     0x0
0xfffce8000ff0: 0x0     0x0
0xfffce8001000: 0x0     0x0
0xfffce8001010: 0x0     0x115
0xfffce8001020: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001030: 0x0     0x0
0xfffce8001040: 0x4031e8 <procD>        0x0
0xfffce8001050: 0x0     0x0
0xfffce8001060: 0x0     0x0
0xfffce8001070: 0x0     0x0
0xfffce8001080: 0x0     0x0
0xfffce8001090: 0x0     0x0
0xfffce80010a0: 0x0     0x0
0xfffce80010b0: 0x0     0x0
0xfffce80010c0: 0x0     0x0
0xfffce80010d0: 0x0     0x0
0xfffce80010e0: 0x0     0x0
0xfffce80010f0: 0x0     0x0
0xfffce8001100: 0x0     0x0
0xfffce8001110: 0x0     0x0
0xfffce8001120: 0x0     0x115
0xfffce8001130: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001140: 0x0     0x0
0xfffce8001150: 0x4031e8 <procD>        0x0
0xfffce8001160: 0x0     0x0
0xfffce8001170: 0x0     0x0
0xfffce8001180: 0x0     0x0
0xfffce8001190: 0x0     0x0
0xfffce80011a0: 0x0     0x0
0xfffce80011b0: 0x0     0x0
0xfffce80011c0: 0x0     0x0
0xfffce80011d0: 0x0     0x0
0xfffce80011e0: 0x0     0x0
0xfffce80011f0: 0x0     0x0
0xfffce8001200: 0x0     0x0
0xfffce8001210: 0x0     0x0
0xfffce8001220: 0x0     0x0
```

```
0xfffce8001230: 0x0     0x115
0xfffce8001240: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001250: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffce8001260: 0x4031e8 <procD>        0x0
0xfffce8001270: 0x0     0x0
0xfffce8001280: 0x0     0x0
0xfffce8001290: 0x0     0x0
0xfffce80012a0: 0x0     0x0
0xfffce80012b0: 0x0     0x0
0xfffce80012c0: 0x0     0x0
0xfffce80012d0: 0x0     0x0
0xfffce80012e0: 0x0     0x0
0xfffce80012f0: 0x0     0x0
0xfffce8001300: 0x0     0x0
0xfffce8001310: 0x0     0x0
0xfffce8001320: 0x0     0x0
0xfffce8001330: 0x0     0x0
0xfffce8001340: 0x0     0x115
0xfffce8001350: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001360: 0x0     0x0
0xfffce8001370: 0x4031e8 <procD>        0x0
0xfffce8001380: 0x0     0x0
0xfffce8001390: 0x0     0x0
0xfffce80013a0: 0x0     0x0
0xfffce80013b0: 0x0     0x0
0xfffce80013c0: 0x0     0x0
0xfffce80013d0: 0x0     0x0
0xfffce80013e0: 0x0     0x0
0xfffce80013f0: 0x0     0x0
0xfffce8001400: 0x0     0x0
0xfffce8001410: 0x0     0x0
0xfffce8001420: 0x0     0x0
0xfffce8001430: 0x0     0x0
0xfffce8001440: 0x0     0x0
0xfffce8001450: 0x0     0x115
0xfffce8001460: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001470: 0x0     0x0
0xfffce8001480: 0x4031e8 <procD>        0x0
0xfffce8001490: 0x0     0x0
0xfffce80014a0: 0x0     0x0
0xfffce80014b0: 0x0     0x0
0xfffce80014c0: 0x0     0x0
0xfffce80014d0: 0x0     0x0
0xfffce80014e0: 0x0     0x0
0xfffce80014f0: 0x0     0x0
0xfffce8001500: 0x0     0x0
0xfffce8001510: 0x0     0x0
0xfffce8001520: 0x0     0x0
0xfffce8001530: 0x0     0x0
0xfffce8001540: 0x0     0x0
0xfffce8001550: 0x0     0x0
0xfffce8001560: 0x0     0x115
--Type <RET> for more, q to quit, c to continue without paging--
0xfffce8001570: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001580: 0x0     0x0
0xfffce8001590: 0x4031e8 <procD>        0x0
0xfffce80015a0: 0x0     0x0
0xfffce80015b0: 0x0     0x0
0xfffce80015c0: 0x0     0x0
```

```
0xfffce80015d0: 0x0     0x0
0xfffce80015e0: 0x0     0x0
0xfffce80015f0: 0x0     0x0
0xfffce8001600: 0x0     0x0
0xfffce8001610: 0x0     0x0
0xfffce8001620: 0x0     0x0
0xfffce8001630: 0x0     0x0
0xfffce8001640: 0x0     0x0
0xfffce8001650: 0x0     0x0
0xfffce8001660: 0x0     0x0
0xfffce8001670: 0x0     0x115
0xfffce8001680: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001690: 0x0     0x0
0xfffce80016a0: 0x4031e8 <procD>        0x0
0xfffce80016b0: 0x0     0x0
0xfffce80016c0: 0x0     0x0
0xfffce80016d0: 0x0     0x0
0xfffce80016e0: 0x0     0x0
0xfffce80016f0: 0x0     0x0
0xfffce8001700: 0x0     0x0
0xfffce8001710: 0x0     0x0
0xfffce8001720: 0x0     0x0
0xfffce8001730: 0x0     0x0
0xfffce8001740: 0x0     0x0
0xfffce8001750: 0x0     0x0
0xfffce8001760: 0x0     0x0
0xfffce8001770: 0x0     0x0
0xfffce8001780: 0x0     0x115
0xfffce8001790: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce80017a0: 0x0     0x0
0xfffce80017b0: 0x4031e8 <procD>        0x0
0xfffce80017c0: 0x0     0x0
0xfffce80017d0: 0x0     0x0
0xfffce80017e0: 0x0     0x0
0xfffce80017f0: 0x0     0x0
0xfffce8001800: 0x0     0x0
0xfffce8001810: 0x0     0x0
0xfffce8001820: 0x0     0x0
0xfffce8001830: 0x0     0x0
0xfffce8001840: 0x0     0x0
0xfffce8001850: 0x0     0x0
0xfffce8001860: 0x0     0x0
0xfffce8001870: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffce8001880: 0x0     0x0
0xfffce8001890: 0x0     0x115
0xfffce80018a0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce80018b0: 0x0     0x0
0xfffce80018c0: 0x4031e8 <procD>        0x0
0xfffce80018d0: 0x0     0x0
0xfffce80018e0: 0x0     0x0
0xfffce80018f0: 0x0     0x0
0xfffce8001900: 0x0     0x0
0xfffce8001910: 0x0     0x0
0xfffce8001920: 0x0     0x0
0xfffce8001930: 0x0     0x0
0xfffce8001940: 0x0     0x0
0xfffce8001950: 0x0     0x0
0xfffce8001960: 0x0     0x0
0xfffce8001970: 0x0     0x0
```

```
0xfffce8001980: 0x0     0x0
0xfffce8001990: 0x0     0x0
0xfffce80019a0: 0x0     0x115
0xfffce80019b0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce80019c0: 0x0     0x0
0xfffce80019d0: 0x4031e8 <procD>        0x0
0xfffce80019e0: 0x0     0x0
0xfffce80019f0: 0x0     0x0
0xfffce8001a00: 0x0     0x0
0xfffce8001a10: 0x0     0x0
0xfffce8001a20: 0x0     0x0
0xfffce8001a30: 0x0     0x0
0xfffce8001a40: 0x0     0x0
0xfffce8001a50: 0x0     0x0
0xfffce8001a60: 0x0     0x0
0xfffce8001a70: 0x0     0x0
0xfffce8001a80: 0x0     0x0
0xfffce8001a90: 0x0     0x0
0xfffce8001aa0: 0x0     0x0
0xfffce8001ab0: 0x0     0x115
0xfffce8001ac0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001ad0: 0x0     0x0
0xfffce8001ae0: 0x4031e8 <procD>        0x0
0xfffce8001af0: 0x0     0x0
0xfffce8001b00: 0x0     0x0
0xfffce8001b10: 0x0     0x0
0xfffce8001b20: 0x0     0x0
0xfffce8001b30: 0x0     0x0
0xfffce8001b40: 0x0     0x0
0xfffce8001b50: 0x0     0x0
0xfffce8001b60: 0x0     0x0
0xfffce8001b70: 0x0     0x0
0xfffce8001b80: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffce8001b90: 0x0     0x0
0xfffce8001ba0: 0x0     0x0
0xfffce8001bb0: 0x0     0x0
0xfffce8001bc0: 0x0     0x115
0xfffce8001bd0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001be0: 0x0     0x0
0xfffce8001bf0: 0x4031e8 <procD>        0x0
0xfffce8001c00: 0x0     0x0
0xfffce8001c10: 0x0     0x0
0xfffce8001c20: 0x0     0x0
0xfffce8001c30: 0x0     0x0
0xfffce8001c40: 0x0     0x0
0xfffce8001c50: 0x0     0x0
0xfffce8001c60: 0x0     0x0
0xfffce8001c70: 0x0     0x0
0xfffce8001c80: 0x0     0x0
0xfffce8001c90: 0x0     0x0
0xfffce8001ca0: 0x0     0x0
0xfffce8001cb0: 0x0     0x0
0xfffce8001cc0: 0x0     0x0
0xfffce8001cd0: 0x0     0x115
0xfffce8001ce0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001cf0: 0x0     0x0
0xfffce8001d00: 0x4031e8 <procD>        0x0
0xfffce8001d10: 0x0     0x0
0xfffce8001d20: 0x0     0x0
```

```
0xfffce8001d30: 0x0     0x0
0xfffce8001d40: 0x0     0x0
0xfffce8001d50: 0x0     0x0
0xfffce8001d60: 0x0     0x0
0xfffce8001d70: 0x0     0x0
0xfffce8001d80: 0x0     0x0
0xfffce8001d90: 0x0     0x0
0xfffce8001da0: 0x0     0x0
0xfffce8001db0: 0x0     0x0
0xfffce8001dc0: 0x0     0x0
0xfffce8001dd0: 0x0     0x0
0xfffce8001de0: 0x0     0x115
0xfffce8001df0: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001e00: 0x0     0x0
0xfffce8001e10: 0x4031e8 <procD>        0x0
0xfffce8001e20: 0x0     0x0
0xfffce8001e30: 0x0     0x0
0xfffce8001e40: 0x0     0x0
0xfffce8001e50: 0x0     0x0
0xfffce8001e60: 0x0     0x0
0xfffce8001e70: 0x0     0x0
0xfffce8001e80: 0x0     0x0
0xfffce8001e90: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffce8001ea0: 0x0     0x0
0xfffce8001eb0: 0x0     0x0
0xfffce8001ec0: 0x0     0x0
0xfffce8001ed0: 0x0     0x0
0xfffce8001ee0: 0x0     0x0
0xfffce8001ef0: 0x0     0x115
0xfffce8001f00: 0x657461636f6c6c61      0x79726f6d656d2064
0xfffce8001f10: 0x0     0x0
0xfffce8001f20: 0x4031e8 <procD>        0x0
0xfffce8001f30: 0x0     0x0

(gdb) x/s 0xfffce8001f00
0xfffce8001f00: "allocated memory"
```

8. Compare pmap logs *App9.pmap.1.12057*, *App9.pmap.2.12057*, and *App9.pmap.3.12057* (the first one was saved before the leak started, and the other two correspond to core dumps we looked at):

```
12057:   ./App9
0000000000400000    768K r-x-- App9
00000000004c0000    128K rw--- App9
000000002f860000    256K rw---   [ anon ]
0000fffcf0000000   1344K rw---   [ anon ]
0000fffcf0150000  64192K -----   [ anon ]
0000fffcf7400000     64K -----   [ anon ]
0000fffcf7410000   8192K rw---   [ anon ]
0000fffcf7c10000     64K -----   [ anon ]
0000fffcf7c20000   8192K rw---   [ anon ]
0000fffcf8420000     64K -----   [ anon ]
0000fffcf8430000   8192K rw---   [ anon ]
0000fffcf8c30000     64K -----   [ anon ]
0000fffcf8c40000   8192K rw---   [ anon ]
0000fffcf9440000     64K -----   [ anon ]
0000fffcf9450000   8192K rw---   [ anon ]
0000fffcf9c50000     64K r----   [ anon ]
0000fffcf9c60000     64K r-x--   [ anon ]
0000ffffc2f60000    192K rw---   [ stack ]
```

```
 total           108288K

12057:   ./App9
0000000000400000    768K r-x-- App9
00000000004c0000    128K rw--- App9
000000002f860000    256K rw---   [ anon ]
0000fffce8000000   2240K rw---   [ anon ]
0000fffce8230000  63296K -----   [ anon ]
0000fffcf0000000  65536K rw---   [ anon ]
0000fffcf7400000     64K -----   [ anon ]
0000fffcf7410000   8192K rw---   [ anon ]
0000fffcf7c10000     64K -----   [ anon ]
0000fffcf7c20000   8192K rw---   [ anon ]
0000fffcf8420000     64K -----   [ anon ]
0000fffcf8430000   8192K rw---   [ anon ]
0000fffcf8c30000     64K -----   [ anon ]
0000fffcf8c40000   8192K rw---   [ anon ]
0000fffcf9440000     64K -----   [ anon ]
0000fffcf9450000   8192K rw---   [ anon ]
0000fffcf9c50000     64K r----   [ anon ]
0000fffcf9c60000     64K r-x--   [ anon ]
0000ffffc2f60000    192K rw---   [ stack ]
 total           173824K

12057:   ./App9
0000000000400000    768K r-x-- App9
00000000004c0000    128K rw--- App9
000000002f860000    256K rw---   [ anon ]
0000fffce8000000  68608K rw---   [ anon ]
0000fffcec300000  62464K -----   [ anon ]
0000fffcf0000000  65536K rw---   [ anon ]
0000fffcf7400000     64K -----   [ anon ]
0000fffcf7410000   8192K rw---   [ anon ]
0000fffcf7c10000     64K -----   [ anon ]
0000fffcf7c20000   8192K rw---   [ anon ]
0000fffcf8420000     64K -----   [ anon ]
0000fffcf8430000   8192K rw---   [ anon ]
0000fffcf8c30000     64K -----   [ anon ]
0000fffcf8c40000   8192K rw---   [ anon ]
0000fffcf9440000     64K -----   [ anon ]
0000fffcf9450000   8192K rw---   [ anon ]
0000fffcf9c50000     64K r----   [ anon ]
0000fffcf9c60000     64K r-x--   [ anon ]
0000ffffc2f60000    192K rw---   [ stack ]
 total           239360K
```

Exercise A9 (A64, WinDbg Preview)

Goal: Learn how to identify heap leaks.

Patterns: Memory Leak (Process Heap); Module Hint.

1. Launch WinDbg Preview.

2. The application *App9* was found to consume more and more memory. Several core memory dumps were saved at different times with corresponding *pmap* logs. Load *App9.core.2.12057* dump file from the A64\App9 folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App9\App9.core.2.12057]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
*** WARNING: Unable to verify timestamp for App9
App9+0xca84:
00000000`0040ca84 d4000001 svc         #0
```

3. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App9\App9.log
Opened log file 'C:\ALCDA2\A64\App9\App9.log'
```

4. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App9\
Symbol search path is: srv*;C:\ALCDA2\A64\App9\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app9\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App9\
*** WARNING: Unable to verify timestamp for App9
```

```
0:000> .reload
..
*** WARNING: Unable to verify timestamp for App9

************* Symbol Loading Error Summary **************
Module name            Error
App9                   The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

5. Notice the size of the largest PAGE_READWRITE region, close logging, and quit WinDbg Preview:

```
0:000> !address

Mapping file section regions...
Mapping module regions...

        BaseAddress      EndAddress+1        RegionSize     Type       State                 Protect             Usage
----------------------------------------------------------------------------------------------------------------------
+        0`00000000        0`00400000        0`00400000                                                       <unknown>
+        0`00400000        0`004c0000        0`000c0000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ          Image      [App9;
"/home/opc/ALCDA2/App9/App9"]
+        0`004c0000        0`004e0000        0`00020000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             Image      [App9;
"/home/opc/ALCDA2/App9/App9"]
+        0`004e0000        0`2f860000        0`2f380000                                                       <unknown>
+        0`2f860000        0`2f8a0000        0`00040000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+        0`2f8a0000     fffc`e8000000     fffc`b8760000                                                       <unknown>
+     fffc`e8000000     fffc`e8230000        0`00230000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     fffc`e8230000     fffc`ec000000        0`03dd0000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY              <unknown>  [................]
+     fffc`ec000000     fffc`f0000000        0`04000000                                                       <unknown>
+     fffc`f0000000     fffc`f4000000        0`04000000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     fffc`f4000000     fffc`f7400000        0`03400000                                                       <unknown>
+     fffc`f7400000     fffc`f7410000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY              <unknown>  [................]
+     fffc`f7410000     fffc`f7c10000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     fffc`f7c10000     fffc`f7c20000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY              <unknown>  [................]
+     fffc`f7c20000     fffc`f8420000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     fffc`f8420000     fffc`f8430000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY              <unknown>  [................]
+     fffc`f8430000     fffc`f8c30000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     fffc`f8c30000     fffc`f8c40000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY              <unknown>  [................]
+     fffc`f8c40000     fffc`f9440000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     fffc`f9440000     fffc`f9450000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY              <unknown>  [................]
+     fffc`f9450000     fffc`f9c50000        0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
+     fffc`f9c50000     fffc`f9c60000        0`00010000                                                       <unknown>
+     fffc`f9c60000     fffc`f9c70000        0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ          Image      [linux_vdso_so; "linux-
vdso.so.1"]
+     fffc`f9c70000     ffff`c2f60000        2`c92f0000                                                       <unknown>
+     ffff`c2f60000     ffff`c2f90000        0`00030000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE             <unknown>  [................]
```

```
0:000> .logclose
Closing open log file 'C:\ALCDA2\A64\App9\App9.log'
```

6. Open another instance of WinDbg Preview and load *App9.core.3.12057* dump file from A64\App9 folder. Set up symbol path, reload symbols, and set append logging to the same log file as previously:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App9\App9.core.3.12057]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
```

```
System Uptime: not available
Process Uptime: not available
..
*** WARNING: Unable to verify timestamp for App9
App9+0xca84:
00000000`0040ca84 d4000001 svc         #0

0:000> .sympath+ C:\ALCDA2\A64\App9\
Symbol search path is: srv*;C:\ALCDA2\A64\App9\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app9\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App9\
*** WARNING: Unable to verify timestamp for App9

0:000> .reload
..
*** WARNING: Unable to verify timestamp for App9

************* Symbol Loading Error Summary **************
Module name            Error
App9                   The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.


0:000> .logappend C:\ALCDA2\A64\App9\App9.log
Opened log file 'C:\ALCDA2\A64\App9\App9.log'
```

Note: We ignore warnings and errors as they are not relevant for now.

7. Notice that another PAGE_READWRITE large region appeared after some time.

```
0:000> !address

Mapping file section regions...
Mapping module regions...

        BaseAddress      EndAddress+1        RegionSize     Type       State                 Protect             Usage
----------------------------------------------------------------------------------------------------------------------------
+       0`00000000       0`00400000       0`00400000                                                          <unknown>
+       0`00400000       0`004c0000       0`000c0000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ                Image      [App9;
"/home/opc/ALCDA2/App9/App9"]
+       0`004c0000       0`004e0000       0`00020000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                   Image      [App9;
"/home/opc/ALCDA2/App9/App9"]
+       0`004e0000       0`2f860000       0`2f380000                                                          <unknown>
+       0`2f860000       0`2f8a0000       0`00040000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                   <unknown>  [................]
+       0`2f8a0000     fffc`e8000000     fffc`b8760000                                                       <unknown>
+     fffc`e8000000    fffc`ec300000       0`04300000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                 <unknown>  [................]
+     fffc`ec300000    fffc`f0000000       0`03d00000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                  <unknown>  [................]
+     fffc`f0000000    fffc`f4000000       0`04000000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                 <unknown>  [................]
+     fffc`f4000000    fffc`f7400000       0`03400000                                                       <unknown>
+     fffc`f7400000    fffc`f7410000       0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                  <unknown>  [................]
+     fffc`f7410000    fffc`f7c10000       0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                 <unknown>  [................]
+     fffc`f7c10000    fffc`f7c20000       0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                  <unknown>  [................]
+     fffc`f7c20000    fffc`f8420000       0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                 <unknown>  [................]
+     fffc`f8420000    fffc`f8430000       0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                  <unknown>  [................]
+     fffc`f8430000    fffc`f8c30000       0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                 <unknown>  [................]
+     fffc`f8c30000    fffc`f8c40000       0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                  <unknown>  [................]
+     fffc`f8c40000    fffc`f9440000       0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                 <unknown>  [................]
+     fffc`f9440000    fffc`f9450000       0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_READONLY                  <unknown>  [................]
+     fffc`f9450000    fffc`f9c50000       0`00800000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                 <unknown>  [................]
+     fffc`f9c50000    fffc`f9c60000       0`00010000                                                       <unknown>
+     fffc`f9c60000    fffc`f9c70000       0`00010000 MEM_PRIVATE MEM_COMMIT  PAGE_EXECUTE_READ                Image      [linux_vdso_so; "linux-
vdso.so.1"]
+     fffc`f9c70000    ffff`c2f60000       2`c92f0000                                                       <unknown>
+     ffff`c2f60000    ffff`c2f90000       0`00030000 MEM_PRIVATE MEM_COMMIT  PAGE_READWRITE                 <unknown>  [................]
```

8. Examine region contents for any execution residue and hints (we choose some smaller address range from the section address range):

```
0:000> dps fffc`e8000000 fffc`e8000000+1000
0000fffc`e8000000  0000fffc`f0000020
0000fffc`e8000008  0000fffc`f0000000
0000fffc`e8000010  00000000`04000000
0000fffc`e8000018  00000000`04000000
0000fffc`e8000020  00000000`00000000
0000fffc`e8000028  00000000`00000115
0000fffc`e8000030  65746163`6f6c6c61
0000fffc`e8000038  79726f6d`656d2064
0000fffc`e8000040  00000000`00000000
0000fffc`e8000048  00000000`00000000
0000fffc`e8000050  00000000`004031e8 App9!procD
0000fffc`e8000058  00000000`00000000
0000fffc`e8000060  00000000`00000000
0000fffc`e8000068  00000000`00000000
0000fffc`e8000070  00000000`00000000
0000fffc`e8000078  00000000`00000000
0000fffc`e8000080  00000000`00000000
0000fffc`e8000088  00000000`00000000
0000fffc`e8000090  00000000`00000000
0000fffc`e8000098  00000000`00000000
0000fffc`e80000a0  00000000`00000000
0000fffc`e80000a8  00000000`00000000
0000fffc`e80000b0  00000000`00000000
0000fffc`e80000b8  00000000`00000000
0000fffc`e80000c0  00000000`00000000
0000fffc`e80000c8  00000000`00000000
0000fffc`e80000d0  00000000`00000000
0000fffc`e80000d8  00000000`00000000
0000fffc`e80000e0  00000000`00000000
0000fffc`e80000e8  00000000`00000000
0000fffc`e80000f0  00000000`00000000
0000fffc`e80000f8  00000000`00000000
0000fffc`e8000100  00000000`00000000
0000fffc`e8000108  00000000`00000000
0000fffc`e8000110  00000000`00000000
0000fffc`e8000118  00000000`00000000
0000fffc`e8000120  00000000`00000000
0000fffc`e8000128  00000000`00000000
0000fffc`e8000130  00000000`00000000
0000fffc`e8000138  00000000`00000115
0000fffc`e8000140  65746163`6f6c6c61
0000fffc`e8000148  79726f6d`656d2064
0000fffc`e8000150  00000000`00000000
0000fffc`e8000158  00000000`00000000
0000fffc`e8000160  00000000`004031e8 App9!procD
0000fffc`e8000168  00000000`00000000
0000fffc`e8000170  00000000`00000000
0000fffc`e8000178  00000000`00000000
0000fffc`e8000180  00000000`00000000
0000fffc`e8000188  00000000`00000000
0000fffc`e8000190  00000000`00000000
0000fffc`e8000198  00000000`00000000
0000fffc`e80001a0  00000000`00000000
0000fffc`e80001a8  00000000`00000000
0000fffc`e80001b0  00000000`00000000
0000fffc`e80001b8  00000000`00000000
```

```
0000fffc`e80001c0  00000000`00000000
0000fffc`e80001c8  00000000`00000000
0000fffc`e80001d0  00000000`00000000
0000fffc`e80001d8  00000000`00000000
0000fffc`e80001e0  00000000`00000000
0000fffc`e80001e8  00000000`00000000
0000fffc`e80001f0  00000000`00000000
0000fffc`e80001f8  00000000`00000000
0000fffc`e8000200  00000000`00000000
0000fffc`e8000208  00000000`00000000
0000fffc`e8000210  00000000`00000000
0000fffc`e8000218  00000000`00000000
0000fffc`e8000220  00000000`00000000
0000fffc`e8000228  00000000`00000000
0000fffc`e8000230  00000000`00000000
0000fffc`e8000238  00000000`00000000
0000fffc`e8000240  00000000`00000000
0000fffc`e8000248  00000000`00000115
0000fffc`e8000250  65746163`6f6c6c61
0000fffc`e8000258  79726f6d`656d2064
0000fffc`e8000260  00000000`00000000
0000fffc`e8000268  00000000`00000000
0000fffc`e8000270  00000000`004031e8 App9!procD
0000fffc`e8000278  00000000`00000000
0000fffc`e8000280  00000000`00000000
0000fffc`e8000288  00000000`00000000
0000fffc`e8000290  00000000`00000000
0000fffc`e8000298  00000000`00000000
0000fffc`e80002a0  00000000`00000000
0000fffc`e80002a8  00000000`00000000
0000fffc`e80002b0  00000000`00000000
0000fffc`e80002b8  00000000`00000000
0000fffc`e80002c0  00000000`00000000
0000fffc`e80002c8  00000000`00000000
0000fffc`e80002d0  00000000`00000000
0000fffc`e80002d8  00000000`00000000
0000fffc`e80002e0  00000000`00000000
0000fffc`e80002e8  00000000`00000000
0000fffc`e80002f0  00000000`00000000
0000fffc`e80002f8  00000000`00000000
0000fffc`e8000300  00000000`00000000
0000fffc`e8000308  00000000`00000000
0000fffc`e8000310  00000000`00000000
0000fffc`e8000318  00000000`00000000
0000fffc`e8000320  00000000`00000000
0000fffc`e8000328  00000000`00000000
0000fffc`e8000330  00000000`00000000
0000fffc`e8000338  00000000`00000000
0000fffc`e8000340  00000000`00000000
0000fffc`e8000348  00000000`00000000
0000fffc`e8000350  00000000`00000000
0000fffc`e8000358  00000000`00000115
0000fffc`e8000360  65746163`6f6c6c61
0000fffc`e8000368  79726f6d`656d2064
0000fffc`e8000370  00000000`00000000
0000fffc`e8000378  00000000`00000000
0000fffc`e8000380  00000000`004031e8 App9!procD
0000fffc`e8000388  00000000`00000000
0000fffc`e8000390  00000000`00000000
0000fffc`e8000398  00000000`00000000
```

```
0000fffc`e80003a0  00000000`00000000
0000fffc`e80003a8  00000000`00000000
0000fffc`e80003b0  00000000`00000000
0000fffc`e80003b8  00000000`00000000
0000fffc`e80003c0  00000000`00000000
0000fffc`e80003c8  00000000`00000000
0000fffc`e80003d0  00000000`00000000
0000fffc`e80003d8  00000000`00000000
0000fffc`e80003e0  00000000`00000000
0000fffc`e80003e8  00000000`00000000
0000fffc`e80003f0  00000000`00000000
0000fffc`e80003f8  00000000`00000000
0000fffc`e8000400  00000000`00000000
0000fffc`e8000408  00000000`00000000
0000fffc`e8000410  00000000`00000000
0000fffc`e8000418  00000000`00000000
0000fffc`e8000420  00000000`00000000
0000fffc`e8000428  00000000`00000000
0000fffc`e8000430  00000000`00000000
0000fffc`e8000438  00000000`00000000
0000fffc`e8000440  00000000`00000000
0000fffc`e8000448  00000000`00000000
0000fffc`e8000450  00000000`00000000
0000fffc`e8000458  00000000`00000000
0000fffc`e8000460  00000000`00000000
0000fffc`e8000468  00000000`00000115
0000fffc`e8000470  65746163`6f6c6c61
0000fffc`e8000478  79726f6d`656d2064
0000fffc`e8000480  00000000`00000000
0000fffc`e8000488  00000000`00000000
0000fffc`e8000490  00000000`004031e8 App9!procD
0000fffc`e8000498  00000000`00000000
0000fffc`e80004a0  00000000`00000000
0000fffc`e80004a8  00000000`00000000
0000fffc`e80004b0  00000000`00000000
0000fffc`e80004b8  00000000`00000000
0000fffc`e80004c0  00000000`00000000
0000fffc`e80004c8  00000000`00000000
0000fffc`e80004d0  00000000`00000000
0000fffc`e80004d8  00000000`00000000
0000fffc`e80004e0  00000000`00000000
0000fffc`e80004e8  00000000`00000000
0000fffc`e80004f0  00000000`00000000
0000fffc`e80004f8  00000000`00000000
0000fffc`e8000500  00000000`00000000
0000fffc`e8000508  00000000`00000000
0000fffc`e8000510  00000000`00000000
0000fffc`e8000518  00000000`00000000
0000fffc`e8000520  00000000`00000000
0000fffc`e8000528  00000000`00000000
0000fffc`e8000530  00000000`00000000
0000fffc`e8000538  00000000`00000000
0000fffc`e8000540  00000000`00000000
0000fffc`e8000548  00000000`00000000
0000fffc`e8000550  00000000`00000000
0000fffc`e8000558  00000000`00000000
0000fffc`e8000560  00000000`00000000
0000fffc`e8000568  00000000`00000000
0000fffc`e8000570  00000000`00000000
0000fffc`e8000578  00000000`00000115
```

```
0000fffc`e8000580  65746163`6f6c6c61
0000fffc`e8000588  79726f6d`656d2064
0000fffc`e8000590  00000000`00000000
0000fffc`e8000598  00000000`00000000
0000fffc`e80005a0  00000000`004031e8 App9!procD
0000fffc`e80005a8  00000000`00000000
0000fffc`e80005b0  00000000`00000000
0000fffc`e80005b8  00000000`00000000
0000fffc`e80005c0  00000000`00000000
0000fffc`e80005c8  00000000`00000000
0000fffc`e80005d0  00000000`00000000
0000fffc`e80005d8  00000000`00000000
0000fffc`e80005e0  00000000`00000000
0000fffc`e80005e8  00000000`00000000
0000fffc`e80005f0  00000000`00000000
0000fffc`e80005f8  00000000`00000000
0000fffc`e8000600  00000000`00000000
0000fffc`e8000608  00000000`00000000
0000fffc`e8000610  00000000`00000000
0000fffc`e8000618  00000000`00000000
0000fffc`e8000620  00000000`00000000
0000fffc`e8000628  00000000`00000000
0000fffc`e8000630  00000000`00000000
0000fffc`e8000638  00000000`00000000
0000fffc`e8000640  00000000`00000000
0000fffc`e8000648  00000000`00000000
0000fffc`e8000650  00000000`00000000
0000fffc`e8000658  00000000`00000000
0000fffc`e8000660  00000000`00000000
0000fffc`e8000668  00000000`00000000
0000fffc`e8000670  00000000`00000000
0000fffc`e8000678  00000000`00000000
0000fffc`e8000680  00000000`00000000
0000fffc`e8000688  00000000`00000115
0000fffc`e8000690  65746163`6f6c6c61
0000fffc`e8000698  79726f6d`656d2064
0000fffc`e80006a0  00000000`00000000
0000fffc`e80006a8  00000000`00000000
0000fffc`e80006b0  00000000`004031e8 App9!procD
0000fffc`e80006b8  00000000`00000000
0000fffc`e80006c0  00000000`00000000
0000fffc`e80006c8  00000000`00000000
0000fffc`e80006d0  00000000`00000000
0000fffc`e80006d8  00000000`00000000
0000fffc`e80006e0  00000000`00000000
0000fffc`e80006e8  00000000`00000000
0000fffc`e80006f0  00000000`00000000
0000fffc`e80006f8  00000000`00000000
0000fffc`e8000700  00000000`00000000
0000fffc`e8000708  00000000`00000000
0000fffc`e8000710  00000000`00000000
0000fffc`e8000718  00000000`00000000
0000fffc`e8000720  00000000`00000000
0000fffc`e8000728  00000000`00000000
0000fffc`e8000730  00000000`00000000
0000fffc`e8000738  00000000`00000000
0000fffc`e8000740  00000000`00000000
0000fffc`e8000748  00000000`00000000
0000fffc`e8000750  00000000`00000000
0000fffc`e8000758  00000000`00000000
```

```
0000fffc`e8000760  00000000`00000000
0000fffc`e8000768  00000000`00000000
0000fffc`e8000770  00000000`00000000
0000fffc`e8000778  00000000`00000000
0000fffc`e8000780  00000000`00000000
0000fffc`e8000788  00000000`00000000
0000fffc`e8000790  00000000`00000000
0000fffc`e8000798  00000000`00000115
0000fffc`e80007a0  65746163`6f6c6c61
0000fffc`e80007a8  79726f6d`656d2064
0000fffc`e80007b0  00000000`00000000
0000fffc`e80007b8  00000000`00000000
0000fffc`e80007c0  00000000`004031e8 App9!procD
0000fffc`e80007c8  00000000`00000000
0000fffc`e80007d0  00000000`00000000
0000fffc`e80007d8  00000000`00000000
0000fffc`e80007e0  00000000`00000000
0000fffc`e80007e8  00000000`00000000
0000fffc`e80007f0  00000000`00000000
0000fffc`e80007f8  00000000`00000000
0000fffc`e8000800  00000000`00000000
0000fffc`e8000808  00000000`00000000
0000fffc`e8000810  00000000`00000000
0000fffc`e8000818  00000000`00000000
0000fffc`e8000820  00000000`00000000
0000fffc`e8000828  00000000`00000000
0000fffc`e8000830  00000000`00000000
0000fffc`e8000838  00000000`00000000
0000fffc`e8000840  00000000`00000000
0000fffc`e8000848  00000000`00000000
0000fffc`e8000850  00000000`00000000
0000fffc`e8000858  00000000`00000000
0000fffc`e8000860  00000000`00000000
0000fffc`e8000868  00000000`00000000
0000fffc`e8000870  00000000`00000000
0000fffc`e8000878  00000000`00000000
0000fffc`e8000880  00000000`00000000
0000fffc`e8000888  00000000`00000000
0000fffc`e8000890  00000000`00000000
0000fffc`e8000898  00000000`00000000
0000fffc`e80008a0  00000000`00000000
0000fffc`e80008a8  00000000`00000115
0000fffc`e80008b0  65746163`6f6c6c61
0000fffc`e80008b8  79726f6d`656d2064
0000fffc`e80008c0  00000000`00000000
0000fffc`e80008c8  00000000`00000000
0000fffc`e80008d0  00000000`004031e8 App9!procD
0000fffc`e80008d8  00000000`00000000
0000fffc`e80008e0  00000000`00000000
0000fffc`e80008e8  00000000`00000000
0000fffc`e80008f0  00000000`00000000
0000fffc`e80008f8  00000000`00000000
0000fffc`e8000900  00000000`00000000
0000fffc`e8000908  00000000`00000000
0000fffc`e8000910  00000000`00000000
0000fffc`e8000918  00000000`00000000
0000fffc`e8000920  00000000`00000000
0000fffc`e8000928  00000000`00000000
0000fffc`e8000930  00000000`00000000
0000fffc`e8000938  00000000`00000000
```

```
0000fffc`e8000940  00000000`00000000
0000fffc`e8000948  00000000`00000000
0000fffc`e8000950  00000000`00000000
0000fffc`e8000958  00000000`00000000
0000fffc`e8000960  00000000`00000000
0000fffc`e8000968  00000000`00000000
0000fffc`e8000970  00000000`00000000
0000fffc`e8000978  00000000`00000000
0000fffc`e8000980  00000000`00000000
0000fffc`e8000988  00000000`00000000
0000fffc`e8000990  00000000`00000000
0000fffc`e8000998  00000000`00000000
0000fffc`e80009a0  00000000`00000000
0000fffc`e80009a8  00000000`00000000
0000fffc`e80009b0  00000000`00000000
0000fffc`e80009b8  00000000`00000115
0000fffc`e80009c0  65746163`6f6c6c61
0000fffc`e80009c8  79726f6d`656d2064
0000fffc`e80009d0  00000000`00000000
0000fffc`e80009d8  00000000`00000000
0000fffc`e80009e0  00000000`004031e8 App9!procD
0000fffc`e80009e8  00000000`00000000
0000fffc`e80009f0  00000000`00000000
0000fffc`e80009f8  00000000`00000000
0000fffc`e8000a00  00000000`00000000
0000fffc`e8000a08  00000000`00000000
0000fffc`e8000a10  00000000`00000000
0000fffc`e8000a18  00000000`00000000
0000fffc`e8000a20  00000000`00000000
0000fffc`e8000a28  00000000`00000000
0000fffc`e8000a30  00000000`00000000
0000fffc`e8000a38  00000000`00000000
0000fffc`e8000a40  00000000`00000000
0000fffc`e8000a48  00000000`00000000
0000fffc`e8000a50  00000000`00000000
0000fffc`e8000a58  00000000`00000000
0000fffc`e8000a60  00000000`00000000
0000fffc`e8000a68  00000000`00000000
0000fffc`e8000a70  00000000`00000000
0000fffc`e8000a78  00000000`00000000
0000fffc`e8000a80  00000000`00000000
0000fffc`e8000a88  00000000`00000000
0000fffc`e8000a90  00000000`00000000
0000fffc`e8000a98  00000000`00000000
0000fffc`e8000aa0  00000000`00000000
0000fffc`e8000aa8  00000000`00000000
0000fffc`e8000ab0  00000000`00000000
0000fffc`e8000ab8  00000000`00000000
0000fffc`e8000ac0  00000000`00000000
0000fffc`e8000ac8  00000000`00000115
0000fffc`e8000ad0  65746163`6f6c6c61
0000fffc`e8000ad8  79726f6d`656d2064
0000fffc`e8000ae0  00000000`00000000
0000fffc`e8000ae8  00000000`00000000
0000fffc`e8000af0  00000000`004031e8 App9!procD
0000fffc`e8000af8  00000000`00000000
0000fffc`e8000b00  00000000`00000000
0000fffc`e8000b08  00000000`00000000
0000fffc`e8000b10  00000000`00000000
0000fffc`e8000b18  00000000`00000000
```

```
0000fffc`e8000b20  00000000`00000000
0000fffc`e8000b28  00000000`00000000
0000fffc`e8000b30  00000000`00000000
0000fffc`e8000b38  00000000`00000000
0000fffc`e8000b40  00000000`00000000
0000fffc`e8000b48  00000000`00000000
0000fffc`e8000b50  00000000`00000000
0000fffc`e8000b58  00000000`00000000
0000fffc`e8000b60  00000000`00000000
0000fffc`e8000b68  00000000`00000000
0000fffc`e8000b70  00000000`00000000
0000fffc`e8000b78  00000000`00000000
0000fffc`e8000b80  00000000`00000000
0000fffc`e8000b88  00000000`00000000
0000fffc`e8000b90  00000000`00000000
0000fffc`e8000b98  00000000`00000000
0000fffc`e8000ba0  00000000`00000000
0000fffc`e8000ba8  00000000`00000000
0000fffc`e8000bb0  00000000`00000000
0000fffc`e8000bb8  00000000`00000000
0000fffc`e8000bc0  00000000`00000000
0000fffc`e8000bc8  00000000`00000000
0000fffc`e8000bd0  00000000`00000000
0000fffc`e8000bd8  00000000`00000115
0000fffc`e8000be0  65746163`6f6c6c61
0000fffc`e8000be8  79726f6d`656d2064
0000fffc`e8000bf0  00000000`00000000
0000fffc`e8000bf8  00000000`00000000
0000fffc`e8000c00  00000000`004031e8 App9!procD
0000fffc`e8000c08  00000000`00000000
0000fffc`e8000c10  00000000`00000000
0000fffc`e8000c18  00000000`00000000
0000fffc`e8000c20  00000000`00000000
0000fffc`e8000c28  00000000`00000000
0000fffc`e8000c30  00000000`00000000
0000fffc`e8000c38  00000000`00000000
0000fffc`e8000c40  00000000`00000000
0000fffc`e8000c48  00000000`00000000
0000fffc`e8000c50  00000000`00000000
0000fffc`e8000c58  00000000`00000000
0000fffc`e8000c60  00000000`00000000
0000fffc`e8000c68  00000000`00000000
0000fffc`e8000c70  00000000`00000000
0000fffc`e8000c78  00000000`00000000
0000fffc`e8000c80  00000000`00000000
0000fffc`e8000c88  00000000`00000000
0000fffc`e8000c90  00000000`00000000
0000fffc`e8000c98  00000000`00000000
0000fffc`e8000ca0  00000000`00000000
0000fffc`e8000ca8  00000000`00000000
0000fffc`e8000cb0  00000000`00000000
0000fffc`e8000cb8  00000000`00000000
0000fffc`e8000cc0  00000000`00000000
0000fffc`e8000cc8  00000000`00000000
0000fffc`e8000cd0  00000000`00000000
0000fffc`e8000cd8  00000000`00000000
0000fffc`e8000ce0  00000000`00000000
0000fffc`e8000ce8  00000000`00000115
0000fffc`e8000cf0  65746163`6f6c6c61
0000fffc`e8000cf8  79726f6d`656d2064
```

```
0000fffc`e8000d00  00000000`00000000
0000fffc`e8000d08  00000000`00000000
0000fffc`e8000d10  00000000`004031e8 App9!procD
0000fffc`e8000d18  00000000`00000000
0000fffc`e8000d20  00000000`00000000
0000fffc`e8000d28  00000000`00000000
0000fffc`e8000d30  00000000`00000000
0000fffc`e8000d38  00000000`00000000
0000fffc`e8000d40  00000000`00000000
0000fffc`e8000d48  00000000`00000000
0000fffc`e8000d50  00000000`00000000
0000fffc`e8000d58  00000000`00000000
0000fffc`e8000d60  00000000`00000000
0000fffc`e8000d68  00000000`00000000
0000fffc`e8000d70  00000000`00000000
0000fffc`e8000d78  00000000`00000000
0000fffc`e8000d80  00000000`00000000
0000fffc`e8000d88  00000000`00000000
0000fffc`e8000d90  00000000`00000000
0000fffc`e8000d98  00000000`00000000
0000fffc`e8000da0  00000000`00000000
0000fffc`e8000da8  00000000`00000000
0000fffc`e8000db0  00000000`00000000
0000fffc`e8000db8  00000000`00000000
0000fffc`e8000dc0  00000000`00000000
0000fffc`e8000dc8  00000000`00000000
0000fffc`e8000dd0  00000000`00000000
0000fffc`e8000dd8  00000000`00000000
0000fffc`e8000de0  00000000`00000000
0000fffc`e8000de8  00000000`00000000
0000fffc`e8000df0  00000000`00000000
0000fffc`e8000df8  00000000`00000115
0000fffc`e8000e00  65746163`6f6c6c61
0000fffc`e8000e08  79726f6d`656d2064
0000fffc`e8000e10  00000000`00000000
0000fffc`e8000e18  00000000`00000000
0000fffc`e8000e20  00000000`004031e8 App9!procD
0000fffc`e8000e28  00000000`00000000
0000fffc`e8000e30  00000000`00000000
0000fffc`e8000e38  00000000`00000000
0000fffc`e8000e40  00000000`00000000
0000fffc`e8000e48  00000000`00000000
0000fffc`e8000e50  00000000`00000000
0000fffc`e8000e58  00000000`00000000
0000fffc`e8000e60  00000000`00000000
0000fffc`e8000e68  00000000`00000000
0000fffc`e8000e70  00000000`00000000
0000fffc`e8000e78  00000000`00000000
0000fffc`e8000e80  00000000`00000000
0000fffc`e8000e88  00000000`00000000
0000fffc`e8000e90  00000000`00000000
0000fffc`e8000e98  00000000`00000000
0000fffc`e8000ea0  00000000`00000000
0000fffc`e8000ea8  00000000`00000000
0000fffc`e8000eb0  00000000`00000000
0000fffc`e8000eb8  00000000`00000000
0000fffc`e8000ec0  00000000`00000000
0000fffc`e8000ec8  00000000`00000000
0000fffc`e8000ed0  00000000`00000000
0000fffc`e8000ed8  00000000`00000000
```

```
0000fffc`e8000ee0  00000000`00000000
0000fffc`e8000ee8  00000000`00000000
0000fffc`e8000ef0  00000000`00000000
0000fffc`e8000ef8  00000000`00000000
0000fffc`e8000f00  00000000`00000000
0000fffc`e8000f08  00000000`00000115
0000fffc`e8000f10  65746163`6f6c6c61
0000fffc`e8000f18  79726f6d`656d2064
0000fffc`e8000f20  00000000`00000000
0000fffc`e8000f28  00000000`00000000
0000fffc`e8000f30  00000000`004031e8 App9!procD
0000fffc`e8000f38  00000000`00000000
0000fffc`e8000f40  00000000`00000000
0000fffc`e8000f48  00000000`00000000
0000fffc`e8000f50  00000000`00000000
0000fffc`e8000f58  00000000`00000000
0000fffc`e8000f60  00000000`00000000
0000fffc`e8000f68  00000000`00000000
0000fffc`e8000f70  00000000`00000000
0000fffc`e8000f78  00000000`00000000
0000fffc`e8000f80  00000000`00000000
0000fffc`e8000f88  00000000`00000000
0000fffc`e8000f90  00000000`00000000
0000fffc`e8000f98  00000000`00000000
0000fffc`e8000fa0  00000000`00000000
0000fffc`e8000fa8  00000000`00000000
0000fffc`e8000fb0  00000000`00000000
0000fffc`e8000fb8  00000000`00000000
0000fffc`e8000fc0  00000000`00000000
0000fffc`e8000fc8  00000000`00000000
0000fffc`e8000fd0  00000000`00000000
0000fffc`e8000fd8  00000000`00000000
0000fffc`e8000fe0  00000000`00000000
0000fffc`e8000fe8  00000000`00000000
0000fffc`e8000ff0  00000000`00000000
0000fffc`e8000ff8  00000000`00000000
0000fffc`e8001000  00000000`00000000

0:000> da 0000fffc`e8000f10
0000fffc`e8000f10  "allocated memory"
```

9. Compare pmap logs *App9.pmap.1.12057*, *App9.pmap.2.12057*, and *App9.pmap.3.12057* (the first one was saved before the leak started, and the other two correspond to core dumps we looked at):

```
12057:   ./App9
0000000000400000    768K r-x-- App9
00000000004c0000    128K rw--- App9
000000002f860000    256K rw---   [ anon ]
0000fffcf0000000   1344K rw---   [ anon ]
0000fffcf0150000  64192K -----   [ anon ]
0000fffcf7400000     64K -----   [ anon ]
0000fffcf7410000   8192K rw---   [ anon ]
0000fffcf7c10000     64K -----   [ anon ]
0000fffcf7c20000   8192K rw---   [ anon ]
0000fffcf8420000     64K -----   [ anon ]
0000fffcf8430000   8192K rw---   [ anon ]
0000fffcf8c30000     64K -----   [ anon ]
0000fffcf8c40000   8192K rw---   [ anon ]
0000fffcf9440000     64K -----   [ anon ]
0000fffcf9450000   8192K rw---   [ anon ]
```

```
0000fffcf9c50000     64K r----   [ anon ]
0000fffcf9c60000     64K r-x--   [ anon ]
0000ffffc2f60000    192K rw---   [ stack ]
 total           108288K

12057:   ./App9
0000000000400000    768K r-x-- App9
00000000004c0000    128K rw--- App9
000000002f860000    256K rw---   [ anon ]
0000fffce8000000   2240K rw---   [ anon ]
0000fffce8230000  63296K -----   [ anon ]
0000fffcf0000000  65536K rw---   [ anon ]
0000fffcf7400000     64K -----   [ anon ]
0000fffcf7410000   8192K rw---   [ anon ]
0000fffcf7c10000     64K -----   [ anon ]
0000fffcf7c20000   8192K rw---   [ anon ]
0000fffcf8420000     64K -----   [ anon ]
0000fffcf8430000   8192K rw---   [ anon ]
0000fffcf8c30000     64K -----   [ anon ]
0000fffcf8c40000   8192K rw---   [ anon ]
0000fffcf9440000     64K -----   [ anon ]
0000fffcf9450000   8192K rw---   [ anon ]
0000fffcf9c50000     64K r----   [ anon ]
0000fffcf9c60000     64K r-x--   [ anon ]
0000ffffc2f60000    192K rw---   [ stack ]
 total           173824K

12057:   ./App9
0000000000400000    768K r-x-- App9
00000000004c0000    128K rw--- App9
000000002f860000    256K rw---   [ anon ]
0000fffce8000000  68608K rw---   [ anon ]
0000fffcec300000  62464K -----   [ anon ]
0000fffcf0000000  65536K rw---   [ anon ]
0000fffcf7400000     64K -----   [ anon ]
0000fffcf7410000   8192K rw---   [ anon ]
0000fffcf7c10000     64K -----   [ anon ]
0000fffcf7c20000   8192K rw---   [ anon ]
0000fffcf8420000     64K -----   [ anon ]
0000fffcf8430000   8192K rw---   [ anon ]
0000fffcf8c30000     64K -----   [ anon ]
0000fffcf8c40000   8192K rw---   [ anon ]
0000fffcf9440000     64K -----   [ anon ]
0000fffcf9450000   8192K rw---   [ anon ]
0000fffcf9c50000     64K r----   [ anon ]
0000fffcf9c60000     64K r-x--   [ anon ]
0000ffffc2f60000    192K rw---   [ stack ]
 total           239360K
```

11. We close logging before exiting WinDbg Preview:

```
0:000> .logclose
Closing open log file 'C:\ALCDA2\A64\App9\App9.log
```

Exercise A10

- **Goal:** Learn how to identify heap contention wait chains, synchronization issues, advanced disassembly, dump arrays
- **Patterns:** Double Free (Process Heap); High Contention (Process Heap); Wait Chain (General); Critical Region; Self-Diagnosis (User Mode)
- \ALCDA-Dumps\Exercise-A10-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A10-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A10-A64-WinDbg.pdf

Exercise A10 (x64, GDB)

Goal: Learn how to identify heap contention wait chains, synchronization issues, advanced disassembly, dump arrays.

Patterns: Double Free (Process Heap); High Contention (Process Heap); Wait Chain (General); Critical Region; Self-Diagnosis (User Mode).

1. When we launched *App10,* we got this console output, and a core dump was saved:

```
~/ALCDA2/x64/App10$ ./App10
double free or corruption (!prev)
Aborted (core dumped)
```

2. Load *core.App10* dump file and *App10* executable from the x64/App10 directory:

```
~/ALCDA2/x64/App10$ gdb -c core.App10 -se App10
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App10...done.
[New LWP 398]
[New LWP 396]
[New LWP 397]
[New LWP 401]
[New LWP 400]
[New LWP 399]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App10'.
Program terminated with signal SIGABRT, Aborted.
#0  0x000000000040cc6b in raise ()
[Current thread is 1 (Thread 0x7ff7dbab4700 (LWP 398))]
```

3. Check all threads and identify problem top frames:

```
(gdb) info threads
  Id   Target Id                         Frame
* 1    Thread 0x7ff7dbab4700 (LWP 398) 0x000000000040cc6b in raise ()
  2    Thread 0xc56880 (LWP 396)       0x0000000000441b10 in nanosleep ()
  3    Thread 0x7ff7dc2b5700 (LWP 397) 0x00000000004431e7 in mprotect ()
  4    Thread 0x7ff7da2b1700 (LWP 401) __lll_lock_wait_private () at
../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:63
  5    Thread 0x7ff7daab2700 (LWP 400) __lll_lock_wait_private () at
../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:63
  6    Thread 0x7ff7db2b3700 (LWP 399) __lll_lock_wait_private () at
../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:63
```

4. Check thread #4 and find where it was being executed:

```
(gdb) thread 4
[Switching to thread 4 (Thread 0x7ff7da2b1700 (LWP 401))]
#0  __lll_lock_wait_private () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:63
63      in ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S
```

```
(gdb) bt
#0  __lll_lock_wait_private () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:63
#1  0x000000000041a7b0 in malloc ()
#2  0x0000000000401c79 in proc () at pthread_create.c:688
#3  0x0000000000401da3 in bar_five () at pthread_create.c:688
#4  0x0000000000401db4 in foo_five () at pthread_create.c:688
#5  0x0000000000401dcd in thread_five () at pthread_create.c:688
#6  0x00000000004031c3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000044436f in clone ()
```

```
(gdb) disassemble proc
Dump of assembler code for function proc:
   0x0000000000401bad <+0>:     push   %rbp
   0x0000000000401bae <+1>:     mov    %rsp,%rbp
   0x0000000000401bb1 <+4>:     sub    $0x10,%rsp
   0x0000000000401bb5 <+8>:     callq  0x40d8d0 <rand>
   0x0000000000401bba <+13>:    mov    %eax,%ecx
   0x0000000000401bbc <+15>:    mov    $0x68db8bad,%edx
   0x0000000000401bc1 <+20>:    mov    %ecx,%eax
   0x0000000000401bc3 <+22>:    imul   %edx
   0x0000000000401bc5 <+24>:    sar    $0xc,%edx
   0x0000000000401bc8 <+27>:    mov    %ecx,%eax
   0x0000000000401bca <+29>:    sar    $0x1f,%eax
   0x0000000000401bcd <+32>:    sub    %eax,%edx
   0x0000000000401bcf <+34>:    mov    %edx,%eax
   0x0000000000401bd1 <+36>:    mov    %eax,-0x4(%rbp)
   0x0000000000401bd4 <+39>:    mov    -0x4(%rbp),%eax
   0x0000000000401bd7 <+42>:    imul   $0x2710,%eax,%eax
   0x0000000000401bdd <+48>:    sub    %eax,%ecx
   0x0000000000401bdf <+50>:    mov    %ecx,%eax
   0x0000000000401be1 <+52>:    mov    %eax,-0x4(%rbp)
   0x0000000000401be4 <+55>:    callq  0x40d8d0 <rand>
   0x0000000000401be9 <+60>:    mov    %eax,%ecx
   0x0000000000401beb <+62>:    mov    $0x68db8bad,%edx
   0x0000000000401bf0 <+67>:    mov    %ecx,%eax
   0x0000000000401bf2 <+69>:    imul   %edx
   0x0000000000401bf4 <+71>:    sar    $0xc,%edx
   0x0000000000401bf7 <+74>:    mov    %ecx,%eax
   0x0000000000401bf9 <+76>:    sar    $0x1f,%eax
```

```
   0x0000000000401bfc <+79>:    sub    %eax,%edx
   0x0000000000401bfe <+81>:    mov    %edx,%eax
   0x0000000000401c00 <+83>:    mov    %eax,-0x8(%rbp)
   0x0000000000401c03 <+86>:    mov    -0x8(%rbp),%eax
   0x0000000000401c06 <+89>:    imul   $0x2710,%eax,%eax
   0x0000000000401c0c <+95>:    sub    %eax,%ecx
   0x0000000000401c0e <+97>:    mov    %ecx,%eax
   0x0000000000401c10 <+99>:    mov    %eax,-0x8(%rbp)
   0x0000000000401c13 <+102>:   mov    -0x4(%rbp),%eax
   0x0000000000401c16 <+105>:   cltq
   0x0000000000401c18 <+107>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c20 <+115>:   lea    0xc0919(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c27 <+122>:   mov    (%rdx,%rax,1),%rax
   0x0000000000401c2b <+126>:   test   %rax,%rax
   0x0000000000401c2e <+129>:   je     0x401c6c <proc+191>
   0x0000000000401c30 <+131>:   mov    -0x4(%rbp),%eax
   0x0000000000401c33 <+134>:   cltq
   0x0000000000401c35 <+136>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c3d <+144>:   lea    0xc08fc(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c44 <+151>:   mov    (%rdx,%rax,1),%rax
   0x0000000000401c48 <+155>:   mov    %rax,%rdi
   0x0000000000401c4b <+158>:   callq  0x41ac10 <free>
   0x0000000000401c50 <+163>:   mov    -0x4(%rbp),%eax
   0x0000000000401c53 <+166>:   cltq
   0x0000000000401c55 <+168>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c5d <+176>:   lea    0xc08dc(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c64 <+183>:   movq   $0x0,(%rdx,%rax,1)
   0x0000000000401c6c <+191>:   mov    -0x8(%rbp),%eax
   0x0000000000401c6f <+194>:   cltq
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401c71 <+196>:   mov    %rax,%rdi
   0x0000000000401c74 <+199>:   callq  0x41a5d0 <malloc>
   0x0000000000401c79 <+204>:   mov    %rax,%rcx
   0x0000000000401c7c <+207>:   mov    -0x4(%rbp),%eax
   0x0000000000401c7f <+210>:   cltq
   0x0000000000401c81 <+212>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c89 <+220>:   lea    0xc08b0(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c90 <+227>:   mov    %rcx,(%rdx,%rax,1)
   0x0000000000401c94 <+231>:   jmpq   0x401bb5 <proc+8>
End of assembler dump.
```

5. Check thread #5 and find where it was being executed:

```
(gdb) thread 5
[Switching to thread 5 (Thread 0x7ff7daab2700 (LWP 400))]
#0  __lll_lock_wait_private () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:63
63      in ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S
```

```
(gdb) bt
#0  __lll_lock_wait_private () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:63
#1  0x0000000000417a43 in _int_free ()
#2  0x0000000000401c50 in proc () at pthread_create.c:688
#3  0x0000000000401d64 in bar_four () at pthread_create.c:688
#4  0x0000000000401d75 in foo_four () at pthread_create.c:688
#5  0x0000000000401d8e in thread_four () at pthread_create.c:688
#6  0x00000000004031c3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000044436f in clone ()
```

```
(gdb) disassemble proc
Dump of assembler code for function proc:
   0x0000000000401bad <+0>:     push   %rbp
   0x0000000000401bae <+1>:     mov    %rsp,%rbp
   0x0000000000401bb1 <+4>:     sub    $0x10,%rsp
   0x0000000000401bb5 <+8>:     callq  0x40d8d0 <rand>
   0x0000000000401bba <+13>:    mov    %eax,%ecx
   0x0000000000401bbc <+15>:    mov    $0x68db8bad,%edx
   0x0000000000401bc1 <+20>:    mov    %ecx,%eax
   0x0000000000401bc3 <+22>:    imul   %edx
   0x0000000000401bc5 <+24>:    sar    $0xc,%edx
   0x0000000000401bc8 <+27>:    mov    %ecx,%eax
   0x0000000000401bca <+29>:    sar    $0x1f,%eax
   0x0000000000401bcd <+32>:    sub    %eax,%edx
   0x0000000000401bcf <+34>:    mov    %edx,%eax
   0x0000000000401bd1 <+36>:    mov    %eax,-0x4(%rbp)
   0x0000000000401bd4 <+39>:    mov    -0x4(%rbp),%eax
   0x0000000000401bd7 <+42>:    imul   $0x2710,%eax,%eax
   0x0000000000401bdd <+48>:    sub    %eax,%ecx
   0x0000000000401bdf <+50>:    mov    %ecx,%eax
   0x0000000000401be1 <+52>:    mov    %eax,-0x4(%rbp)
   0x0000000000401be4 <+55>:    callq  0x40d8d0 <rand>
   0x0000000000401be9 <+60>:    mov    %eax,%ecx
   0x0000000000401beb <+62>:    mov    $0x68db8bad,%edx
   0x0000000000401bf0 <+67>:    mov    %ecx,%eax
   0x0000000000401bf2 <+69>:    imul   %edx
   0x0000000000401bf4 <+71>:    sar    $0xc,%edx
   0x0000000000401bf7 <+74>:    mov    %ecx,%eax
   0x0000000000401bf9 <+76>:    sar    $0x1f,%eax
   0x0000000000401bfc <+79>:    sub    %eax,%edx
   0x0000000000401bfe <+81>:    mov    %edx,%eax
   0x0000000000401c00 <+83>:    mov    %eax,-0x8(%rbp)
   0x0000000000401c03 <+86>:    mov    -0x8(%rbp),%eax
   0x0000000000401c06 <+89>:    imul   $0x2710,%eax,%eax
   0x0000000000401c0c <+95>:    sub    %eax,%ecx
   0x0000000000401c0e <+97>:    mov    %ecx,%eax
   0x0000000000401c10 <+99>:    mov    %eax,-0x8(%rbp)
   0x0000000000401c13 <+102>:   mov    -0x4(%rbp),%eax
   0x0000000000401c16 <+105>:   cltq
   0x0000000000401c18 <+107>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c20 <+115>:   lea    0xc0919(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c27 <+122>:   mov    (%rdx,%rax,1),%rax
   0x0000000000401c2b <+126>:   test   %rax,%rax
   0x0000000000401c2e <+129>:   je     0x401c6c <proc+191>
   0x0000000000401c30 <+131>:   mov    -0x4(%rbp),%eax
   0x0000000000401c33 <+134>:   cltq
   0x0000000000401c35 <+136>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c3d <+144>:   lea    0xc08fc(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c44 <+151>:   mov    (%rdx,%rax,1),%rax
   0x0000000000401c48 <+155>:   mov    %rax,%rdi
   0x0000000000401c4b <+158>:   callq  0x41ac10 <free>
   0x0000000000401c50 <+163>:   mov    -0x4(%rbp),%eax
   0x0000000000401c53 <+166>:   cltq
   0x0000000000401c55 <+168>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c5d <+176>:   lea    0xc08dc(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c64 <+183>:   movq   $0x0,(%rdx,%rax,1)
   0x0000000000401c6c <+191>:   mov    -0x8(%rbp),%eax
   0x0000000000401c6f <+194>:   cltq
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401c71 <+196>:   mov    %rax,%rdi
```

```
   0x0000000000401c74 <+199>:   callq  0x41a5d0 <malloc>
   0x0000000000401c79 <+204>:   mov    %rax,%rcx
   0x0000000000401c7c <+207>:   mov    -0x4(%rbp),%eax
   0x0000000000401c7f <+210>:   cltq
   0x0000000000401c81 <+212>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c89 <+220>:   lea    0xc08b0(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c90 <+227>:   mov    %rcx,(%rdx,%rax,1)
   0x0000000000401c94 <+231>:   jmpq   0x401bb5 <proc+8>
End of assembler dump.
```

6. Check thread #6 and find where it was being executed:

```
(gdb) thread 6
[Switching to thread 6 (Thread 0x7ff7db2b3700 (LWP 399))]
#0  __lll_lock_wait_private () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:63
63      in ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S
```

```
(gdb) bt
#0  __lll_lock_wait_private () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:63
#1  0x0000000000417a43 in _int_free ()
#2  0x0000000000401c50 in proc () at pthread_create.c:688
#3  0x0000000000401d25 in bar_three () at pthread_create.c:688
#4  0x0000000000401d36 in foo_three () at pthread_create.c:688
#5  0x0000000000401d4f in thread_three () at pthread_create.c:688
#6  0x00000000004031c3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000044436f in clone ()
```

Note: Thread #6 is the same as thread #5. We disassemble *proc* again and put addresses there that we identified from threads #4 and #5:

```
(gdb) disassemble proc
Dump of assembler code for function proc:
   0x0000000000401bad <+0>:     push   %rbp
   0x0000000000401bae <+1>:     mov    %rsp,%rbp
   0x0000000000401bb1 <+4>:     sub    $0x10,%rsp
   0x0000000000401bb5 <+8>:     callq  0x40d8d0 <rand>
   0x0000000000401bba <+13>:    mov    %eax,%ecx
   0x0000000000401bbc <+15>:    mov    $0x68db8bad,%edx
   0x0000000000401bc1 <+20>:    mov    %ecx,%eax
   0x0000000000401bc3 <+22>:    imul   %edx
   0x0000000000401bc5 <+24>:    sar    $0xc,%edx
   0x0000000000401bc8 <+27>:    mov    %ecx,%eax
   0x0000000000401bca <+29>:    sar    $0x1f,%eax
   0x0000000000401bcd <+32>:    sub    %eax,%edx
   0x0000000000401bcf <+34>:    mov    %edx,%eax
   0x0000000000401bd1 <+36>:    mov    %eax,-0x4(%rbp)
   0x0000000000401bd4 <+39>:    mov    -0x4(%rbp),%eax
   0x0000000000401bd7 <+42>:    imul   $0x2710,%eax,%eax
   0x0000000000401bdd <+48>:    sub    %eax,%ecx
   0x0000000000401bdf <+50>:    mov    %ecx,%eax
   0x0000000000401be1 <+52>:    mov    %eax,-0x4(%rbp)
   0x0000000000401be4 <+55>:    callq  0x40d8d0 <rand>
   0x0000000000401be9 <+60>:    mov    %eax,%ecx
   0x0000000000401beb <+62>:    mov    $0x68db8bad,%edx
   0x0000000000401bf0 <+67>:    mov    %ecx,%eax
   0x0000000000401bf2 <+69>:    imul   %edx
   0x0000000000401bf4 <+71>:    sar    $0xc,%edx
   0x0000000000401bf7 <+74>:    mov    %ecx,%eax
   0x0000000000401bf9 <+76>:    sar    $0x1f,%eax
   0x0000000000401bfc <+79>:    sub    %eax,%edx
```

```
   0x0000000000401bfe <+81>:    mov    %edx,%eax
   0x0000000000401c00 <+83>:    mov    %eax,-0x8(%rbp)
   0x0000000000401c03 <+86>:    mov    -0x8(%rbp),%eax
   0x0000000000401c06 <+89>:    imul   $0x2710,%eax,%eax
   0x0000000000401c0c <+95>:    sub    %eax,%ecx
   0x0000000000401c0e <+97>:    mov    %ecx,%eax
   0x0000000000401c10 <+99>:    mov    %eax,-0x8(%rbp)
   0x0000000000401c13 <+102>:   mov    -0x4(%rbp),%eax
   0x0000000000401c16 <+105>:   cltq
   0x0000000000401c18 <+107>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c20 <+115>:   lea    0xc0919(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c27 <+122>:   mov    (%rdx,%rax,1),%rax
   0x0000000000401c2b <+126>:   test   %rax,%rax
   0x0000000000401c2e <+129>:   je     0x401c6c <proc+191>
   0x0000000000401c30 <+131>:   mov    -0x4(%rbp),%eax
   0x0000000000401c33 <+134>:   cltq
   0x0000000000401c35 <+136>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c3d <+144>:   lea    0xc08fc(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c44 <+151>:   mov    (%rdx,%rax,1),%rax
   0x0000000000401c48 <+155>:   mov    %rax,%rdi
   0x0000000000401c4b <+158>:   callq  0x41ac10 <free>
   0x0000000000401c50 <+163>:   mov    -0x4(%rbp),%eax
   0x0000000000401c53 <+166>:   cltq
   0x0000000000401c55 <+168>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c5d <+176>:   lea    0xc08dc(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c64 <+183>:   movq   $0x0,(%rdx,%rax,1)
   0x0000000000401c6c <+191>:   mov    -0x8(%rbp),%eax
   0x0000000000401c6f <+194>:   cltq
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401c71 <+196>:   mov    %rax,%rdi
   0x0000000000401c74 <+199>:   callq  0x41a5d0 <malloc>
   0x0000000000401c79 <+204>:   mov    %rax,%rcx
   0x0000000000401c7c <+207>:   mov    -0x4(%rbp),%eax
   0x0000000000401c7f <+210>:   cltq
   0x0000000000401c81 <+212>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c89 <+220>:   lea    0xc08b0(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c90 <+227>:   mov    %rcx,(%rdx,%rax,1)
   0x0000000000401c94 <+231>:   jmpq   0x401bb5 <proc+8>
End of assembler dump.
```

Note: We see some buffer 0x4c2540 “sandwiched” between *free* and *malloc* calls that internally call “lock” and “unlock” functions.

7. Check thread #3 and find where it was being executed:

```
(gdb) thread 3
[Switching to thread 3 (Thread 0x7ff7dc2b5700 (LWP 397))]
#0  0x00000000004431e7 in mprotect ()
```

```
(gdb) bt
#0  0x00000000004431e7 in mprotect ()
#1  0x000000000041834c in sysmalloc ()
#2  0x00000000004194e1 in _int_malloc ()
#3  0x000000000041a7c2 in malloc ()
#4  0x0000000000401c79 in proc () at pthread_create.c:688
#5  0x0000000000401ca7 in bar_one () at pthread_create.c:688
#6  0x0000000000401cb8 in foo_one () at pthread_create.c:688
#7  0x0000000000401cd1 in thread_one () at pthread_create.c:688
#8  0x00000000004031c3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#9  0x000000000044436f in clone ()
```

Note: Thread #3 is the same as thread #4.

8. Check thread #1 and identify a diagnostic message:

```
(gdb) thread 1
[Switching to thread 1 (Thread 0x7ff7dbab4700 (LWP 398))]
#0  0x000000000040cc6b in raise ()

(gdb) bt
#0  0x000000000040cc6b in raise ()
#1  0x0000000000401241 in abort () at pthread_create.c:688
#2  0x0000000000410828 in __libc_message ()
#3  0x0000000000415fea in malloc_printerr ()
#4  0x00000000004179fc in _int_free ()
#5  0x0000000000401c50 in proc () at pthread_create.c:688
#6  0x0000000000401ce6 in bar_two () at pthread_create.c:688
#7  0x0000000000401cf7 in foo_two () at pthread_create.c:688
#8  0x0000000000401d10 in thread_two () at pthread_create.c:688
#9  0x00000000004031c3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#10 0x000000000044436f in clone ()

(gdb) disassemble __libc_message
Dump of assembler code for function __libc_message:
   0x0000000000410560 <+0>:     push   %rbp
   0x0000000000410561 <+1>:     mov    %rsp,%rbp
   0x0000000000410564 <+4>:     push   %r15
   0x0000000000410566 <+6>:     push   %r14
   0x0000000000410568 <+8>:     push   %r13
   0x000000000041056a <+10>:    mov    %edi,%r13d
   0x000000000041056d <+13>:    push   %r12
   0x000000000041056f <+15>:    push   %rbx
   0x0000000000410570 <+16>:    mov    %rsi,%rbx
   0x0000000000410573 <+19>:    sub    $0x68,%rsp
   0x0000000000410577 <+23>:    mov    %rdx,-0x50(%rbp)
   0x000000000041057b <+27>:    mov    %rcx,-0x48(%rbp)
   0x000000000041057f <+31>:    mov    %r8,-0x40(%rbp)
   0x0000000000410583 <+35>:    mov    %r9,-0x38(%rbp)
   0x0000000000410587 <+39>:    mov    %fs:0x28,%rax
   0x0000000000410590 <+48>:    mov    %rax,-0x68(%rbp)
   0x0000000000410594 <+52>:    xor    %eax,%eax
   0x0000000000410596 <+54>:    lea    0x10(%rbp),%rax
   0x000000000041059a <+58>:    and    $0x2,%edi
   0x000000000041059d <+61>:    movl   $0x10,-0x80(%rbp)
   0x00000000004105a4 <+68>:    mov    %rax,-0x78(%rbp)
   0x00000000004105a8 <+72>:    lea    -0x60(%rbp),%rax
   0x00000000004105ac <+76>:    mov    %rax,-0x70(%rbp)
   0x00000000004105b0 <+80>:    jne    0x410761 <__libc_message+513>
   0x00000000004105b6 <+86>:    movl   $0x2,-0x84(%rbp)
   0x00000000004105c0 <+96>:    movzbl (%rbx),%r12d
   0x00000000004105c4 <+100>:   and    $0x1,%r13d
   0x00000000004105c8 <+104>:   xor    %r14d,%r14d
   0x00000000004105cb <+107>:   mov    $0x20,%r15d
   0x00000000004105d1 <+113>:   mov    %r13d,-0x88(%rbp)
   0x00000000004105d8 <+120>:   xor    %r13d,%r13d
   0x00000000004105db <+123>:   test   %r12b,%r12b
   0x00000000004105de <+126>:   je     0x410737 <__libc_message+471>
   0x00000000004105e4 <+132>:   nopl   0x0(%rax)
   0x00000000004105e8 <+136>:   mov    %r12d,%edx
   0x00000000004105eb <+139>:   mov    %rbx,%rax
   0x00000000004105ee <+142>:   jmp    0x410605 <__libc_message+165>
```

```
   0x00000000004105f0 <+144>:   lea    0x1(%rax),%rdi
   0x00000000004105f4 <+148>:   mov    $0x25,%esi
   0x00000000004105f9 <+153>:   callq  0x401060
   0x00000000004105fe <+158>:   movzbl (%rax),%edx
   0x0000000000410601 <+161>:   test   %dl,%dl
   0x0000000000410603 <+163>:   je     0x410610 <__libc_message+176>
   0x0000000000410605 <+165>:   cmp    $0x25,%dl
   0x0000000000410608 <+168>:   jne    0x4105f0 <__libc_message+144>
   0x000000000041060a <+170>:   cmpb   $0x73,0x1(%rax)
   0x000000000041060e <+174>:   jne    0x4105f0 <__libc_message+144>
   0x0000000000410610 <+176>:   cmp    $0x25,%r12b
   0x0000000000410614 <+180>:   je     0x410650 <__libc_message+240>
   0x0000000000410616 <+182>:   mov    %rax,%r9
   0x0000000000410619 <+185>:   mov    %rbx,%rsi
   0x000000000041061c <+188>:   sub    %rbx,%r9
   0x000000000041061f <+191>:   mov    %rax,%rbx
   0x0000000000410622 <+194>:   sub    %r15,%rsp
   0x0000000000410625 <+197>:   lea    0x1(%r14),%r8d
   0x0000000000410629 <+201>:   lea    0xf(%rsp),%rdx
--Type <RET> for more, q to quit, c to continue without paging--
   0x000000000041062e <+206>:   and    $0xfffffffffffffff0,%rdx
   0x0000000000410632 <+210>:   mov    %rsi,(%rdx)
   0x0000000000410635 <+213>:   mov    %r9,0x8(%rdx)
   0x0000000000410639 <+217>:   mov    %r13,0x10(%rdx)
   0x000000000041063d <+221>:   movzbl (%rbx),%r12d
   0x0000000000410641 <+225>:   test   %r12b,%r12b
   0x0000000000410644 <+228>:   je     0x410690 <__libc_message+304>
   0x0000000000410646 <+230>:   movslq %r8d,%r14
   0x0000000000410649 <+233>:   mov    %rdx,%r13
   0x000000000041064c <+236>:   jmp    0x4105e8 <__libc_message+136>
   0x000000000041064e <+238>:   xchg   %ax,%ax
   0x0000000000410650 <+240>:   cmpb   $0x73,0x1(%rbx)
   0x0000000000410654 <+244>:   jne    0x410616 <__libc_message+182>
   0x0000000000410656 <+246>:   mov    -0x80(%rbp),%eax
   0x0000000000410659 <+249>:   cmp    $0x2f,%eax
   0x000000000041065c <+252>:   ja     0x410750 <__libc_message+496>
   0x0000000000410662 <+258>:   mov    %eax,%edx
   0x0000000000410664 <+260>:   add    $0x8,%eax
   0x0000000000410667 <+263>:   add    -0x70(%rbp),%rdx
   0x000000000041066b <+267>:   mov    %eax,-0x80(%rbp)
   0x000000000041066e <+270>:   mov    (%rdx),%rsi
   0x0000000000410671 <+273>:   add    $0x2,%rbx
   0x0000000000410675 <+277>:   mov    %rsi,%rdi
   0x0000000000410678 <+280>:   mov    %rsi,-0x90(%rbp)
   0x000000000041067f <+287>:   callq  0x4010d8
   0x0000000000410684 <+292>:   mov    -0x90(%rbp),%rsi
   0x000000000041068b <+299>:   mov    %rax,%r9
   0x000000000041068e <+302>:   jmp    0x410622 <__libc_message+194>
   0x0000000000410690 <+304>:   movslq %r8d,%r8
   0x0000000000410693 <+307>:   shl    $0x4,%r14
   0x0000000000410697 <+311>:   xor    %edx,%edx
   0x0000000000410699 <+313>:   mov    %r8,%rax
   0x000000000041069c <+316>:   shl    $0x4,%rax
   0x00000000004106a0 <+320>:   add    $0x10,%rax
   0x00000000004106a4 <+324>:   sub    %rax,%rsp
   0x00000000004106a7 <+327>:   lea    0xf(%rsp),%rbx
   0x00000000004106ac <+332>:   and    $0xfffffffffffffff0,%rbx
   0x00000000004106b0 <+336>:   lea    (%rbx,%r14,1),%rax
   0x00000000004106b4 <+340>:   mov    %rbx,%r12
   0x00000000004106b7 <+343>:   mov    %rax,%rdi
```

```
   0x00000000004106ba <+346>:   sub    %r14,%rdi
   0x00000000004106bd <+349>:   jmp    0x4106d0 <__libc_message+368>
   0x00000000004106bf <+351>:   nop
   0x00000000004106c0 <+352>:   mov    0x0(%r13),%rsi
   0x00000000004106c4 <+356>:   mov    0x8(%r13),%r9
   0x00000000004106c8 <+360>:   sub    $0x10,%rax
   0x00000000004106cc <+364>:   mov    0x10(%r13),%r13
   0x00000000004106d0 <+368>:   mov    %r9,0x8(%rax)
   0x00000000004106d4 <+372>:   add    %rdx,%r9
   0x00000000004106d7 <+375>:   mov    %rsi,(%rax)
   0x00000000004106da <+378>:   mov    %r9,%rdx
   0x00000000004106dd <+381>:   cmp    %rax,%rdi
   0x00000000004106e0 <+384>:   jne    0x4106c0 <__libc_message+352>
   0x00000000004106e2 <+386>:   mov    $0x14,%r10d
   0x00000000004106e8 <+392>:   nopl   0x0(%rax,%rax,1)
   0x00000000004106f0 <+400>:   mov    %r8,%rdx
   0x00000000004106f3 <+403>:   mov    %rbx,%rsi
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004106f6 <+406>:   mov    -0x84(%rbp),%edi
   0x00000000004106fc <+412>:   mov    %r10d,%eax
   0x00000000004106ff <+415>:   syscall
   0x0000000000410701 <+417>:   cmp    $0xfffffffffffffffc,%rax
   0x0000000000410705 <+421>:   je     0x4106f0 <__libc_message+400>
   0x0000000000410707 <+423>:   mov    -0x88(%rbp),%eax
   0x000000000041070d <+429>:   test   %eax,%eax
   0x000000000041070f <+431>:   jne    0x4107a2 <__libc_message+578>
   0x0000000000410715 <+437>:   mov    -0x68(%rbp),%rax
   0x0000000000410719 <+441>:   xor    %fs:0x28,%rax
   0x0000000000410722 <+450>:   jne    0x410828 <__libc_message+712>
   0x0000000000410728 <+456>:   lea    -0x28(%rbp),%rsp
   0x000000000041072c <+460>:   pop    %rbx
   0x000000000041072d <+461>:   pop    %r12
   0x000000000041072f <+463>:   pop    %r13
   0x0000000000410731 <+465>:   pop    %r14
   0x0000000000410733 <+467>:   pop    %r15
   0x0000000000410735 <+469>:   pop    %rbp
   0x0000000000410736 <+470>:   retq
   0x0000000000410737 <+471>:   mov    -0x88(%rbp),%edx
   0x000000000041073d <+477>:   test   %edx,%edx
   0x000000000041073f <+479>:   je     0x410715 <__libc_message+437>
   0x0000000000410741 <+481>:   jmpq   0x410823 <__libc_message+707>
   0x0000000000410746 <+486>:   nopw   %cs:0x0(%rax,%rax,1)
   0x0000000000410750 <+496>:   mov    -0x78(%rbp),%rdx
   0x0000000000410754 <+500>:   lea    0x8(%rdx),%rax
   0x0000000000410758 <+504>:   mov    %rax,-0x78(%rbp)
   0x000000000041075c <+508>:   jmpq   0x41066e <__libc_message+270>
   0x0000000000410761 <+513>:   lea    0x8585c(%rip),%rdi        # 0x495fc4
   0x0000000000410768 <+520>:   callq  0x453da0 <secure_getenv>
   0x000000000041076d <+525>:   test   %rax,%rax
   0x0000000000410770 <+528>:   je     0x41077b <__libc_message+539>
   0x0000000000410772 <+530>:   cmpb   $0x0,(%rax)
   0x0000000000410775 <+533>:   jne    0x4105b6 <__libc_message+86>
   0x000000000041077b <+539>:   mov    $0x902,%esi
   0x0000000000410780 <+544>:   lea    0x85850(%rip),%rdi        # 0x495fd7
   0x0000000000410787 <+551>:   xor    %eax,%eax
   0x0000000000410789 <+553>:   callq  0x442da0 <__open_nocancel>
   0x000000000041078e <+558>:   mov    %eax,-0x84(%rbp)
   0x0000000000410794 <+564>:   cmp    $0xffffffff,%eax
   0x0000000000410797 <+567>:   jne    0x4105c0 <__libc_message+96>
   0x000000000041079d <+573>:   jmpq   0x4105b6 <__libc_message+86>
```

```
   0x00000000004107a2 <+578>:   mov    0xb0aaf(%rip),%rax        # 0x4c1258 <_dl_pagesize>
   0x00000000004107a9 <+585>:   mov    $0xffffffff,%r8d
   0x00000000004107af <+591>:   mov    $0x3,%edx
   0x00000000004107b4 <+596>:   xor    %edi,%edi
   0x00000000004107b6 <+598>:   lea    (%r9,%rax,1),%rcx
   0x00000000004107ba <+602>:   neg    %rax
   0x00000000004107bd <+605>:   xor    %r9d,%r9d
   0x00000000004107c0 <+608>:   and    %rax,%rcx
   0x00000000004107c3 <+611>:   mov    %rcx,%r13
   0x00000000004107c6 <+614>:   mov    $0x22,%ecx
   0x00000000004107cb <+619>:   mov    %r13,%rsi
   0x00000000004107ce <+622>:   callq  0x4430d0 <mmap64>
   0x00000000004107d3 <+627>:   mov    %rax,%r15
   0x00000000004107d6 <+630>:   cmp    $0xffffffffffffffff,%rax
   0x00000000004107da <+634>:   je     0x410823 <__libc_message+707>
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004107dc <+636>:   mov    %r13d,(%rax)
   0x00000000004107df <+639>:   lea    0x10(%rbx,%r14,1),%rbx
   0x00000000004107e4 <+644>:   lea    0x4(%rax),%rax
   0x00000000004107e8 <+648>:   nopl   0x0(%rax,%rax,1)
   0x00000000004107f0 <+656>:   mov    0x8(%r12),%rdx
   0x00000000004107f5 <+661>:   mov    (%r12),%rsi
   0x00000000004107f9 <+665>:   mov    %rax,%rdi
   0x00000000004107fc <+668>:   add    $0x10,%r12
   0x0000000000410800 <+672>:   callq  0x4010b8
   0x0000000000410805 <+677>:   cmp    %r12,%rbx
   0x0000000000410808 <+680>:   jne    0x4107f0 <__libc_message+656>
   0x000000000041080a <+682>:   movb   $0x0,(%rax)
   0x000000000041080d <+685>:   mov    %r15,%rdi
   0x0000000000410810 <+688>:   xchg   %rdi,0xc9b69(%rip)        # 0x4da380 <__abort_msg>
   0x0000000000410817 <+695>:   test   %rdi,%rdi
   0x000000000041081a <+698>:   je     0x410823 <__libc_message+707>
   0x000000000041081c <+700>:   mov    (%rdi),%esi
   0x000000000041081e <+702>:   callq  0x4431b0 <munmap>
   0x0000000000410823 <+707>:   callq  0x401120 <abort>
   0x0000000000410828 <+712>:   callq  0x4449a0 <__stack_chk_fail_local>
End of assembler dump.

(gdb) x/a 0x4da380
0x4da380 <__abort_msg>: 0x7ff7d9ab0000

(gdb) x/10s 0x7ff7d9ab0000
0x7ff7d9ab0000: ""
0x7ff7d9ab0001: "\020"
0x7ff7d9ab0003: ""
0x7ff7d9ab0004: "double free or corruption (!prev)\n"
0x7ff7d9ab0027: ""
0x7ff7d9ab0028: ""
0x7ff7d9ab0029: ""
0x7ff7d9ab002a: ""
0x7ff7d9ab002b: ""
0x7ff7d9ab002c: ""
```

9. Check the address that was being freed:

```
(gdb) bt
#0  0x000000000040cc6b in raise ()
#1  0x0000000000401241 in abort () at pthread_create.c:688
#2  0x0000000000410828 in __libc_message ()
#3  0x0000000000415fea in malloc_printerr ()
#4  0x00000000004179fc in _int_free ()
#5  0x0000000000401c50 in proc () at pthread_create.c:688
#6  0x0000000000401ce6 in bar_two () at pthread_create.c:688
#7  0x0000000000401cf7 in foo_two () at pthread_create.c:688
#8  0x0000000000401d10 in thread_two () at pthread_create.c:688
#9  0x00000000004031c3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#10 0x000000000044436f in clone ()

(gdb) frame 5
#5  0x0000000000401c50 in proc () at pthread_create.c:688
688     pthread_create.c: No such file or directory.

(gdb) disassemble proc
Dump of assembler code for function proc:
   0x0000000000401bad <+0>:     push   %rbp
   0x0000000000401bae <+1>:     mov    %rsp,%rbp
   0x0000000000401bb1 <+4>:     sub    $0x10,%rsp
   0x0000000000401bb5 <+8>:     callq  0x40d8d0 <rand>
   0x0000000000401bba <+13>:    mov    %eax,%ecx
   0x0000000000401bbc <+15>:    mov    $0x68db8bad,%edx
   0x0000000000401bc1 <+20>:    mov    %ecx,%eax
   0x0000000000401bc3 <+22>:    imul   %edx
   0x0000000000401bc5 <+24>:    sar    $0xc,%edx
   0x0000000000401bc8 <+27>:    mov    %ecx,%eax
   0x0000000000401bca <+29>:    sar    $0x1f,%eax
   0x0000000000401bcd <+32>:    sub    %eax,%edx
   0x0000000000401bcf <+34>:    mov    %edx,%eax
   0x0000000000401bd1 <+36>:    mov    %eax,-0x4(%rbp)
   0x0000000000401bd4 <+39>:    mov    -0x4(%rbp),%eax
   0x0000000000401bd7 <+42>:    imul   $0x2710,%eax,%eax
   0x0000000000401bdd <+48>:    sub    %eax,%ecx
   0x0000000000401bdf <+50>:    mov    %ecx,%eax
   0x0000000000401be1 <+52>:    mov    %eax,-0x4(%rbp)
   0x0000000000401be4 <+55>:    callq  0x40d8d0 <rand>
   0x0000000000401be9 <+60>:    mov    %eax,%ecx
   0x0000000000401beb <+62>:    mov    $0x68db8bad,%edx
   0x0000000000401bf0 <+67>:    mov    %ecx,%eax
   0x0000000000401bf2 <+69>:    imul   %edx
   0x0000000000401bf4 <+71>:    sar    $0xc,%edx
   0x0000000000401bf7 <+74>:    mov    %ecx,%eax
   0x0000000000401bf9 <+76>:    sar    $0x1f,%eax
   0x0000000000401bfc <+79>:    sub    %eax,%edx
   0x0000000000401bfe <+81>:    mov    %edx,%eax
   0x0000000000401c00 <+83>:    mov    %eax,-0x8(%rbp)
   0x0000000000401c03 <+86>:    mov    -0x8(%rbp),%eax
   0x0000000000401c06 <+89>:    imul   $0x2710,%eax,%eax
   0x0000000000401c0c <+95>:    sub    %eax,%ecx
   0x0000000000401c0e <+97>:    mov    %ecx,%eax
   0x0000000000401c10 <+99>:    mov    %eax,-0x8(%rbp)
   0x0000000000401c13 <+102>:   mov    -0x4(%rbp),%eax
   0x0000000000401c16 <+105>:   cltq
   0x0000000000401c18 <+107>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c20 <+115>:   lea    0xc0919(%rip),%rax        # 0x4c2540 <pAllocBuf>
```

```
   0x0000000000401c27 <+122>:   mov    (%rdx,%rax,1),%rax
   0x0000000000401c2b <+126>:   test   %rax,%rax
   0x0000000000401c2e <+129>:   je     0x401c6c <proc+191>
   0x0000000000401c30 <+131>:   mov    -0x4(%rbp),%eax
   0x0000000000401c33 <+134>:   cltq
   0x0000000000401c35 <+136>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c3d <+144>:   lea    0xc08fc(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c44 <+151>:   mov    (%rdx,%rax,1),%rax
   0x0000000000401c48 <+155>:   mov    %rax,%rdi
   0x0000000000401c4b <+158>:   callq  0x41ac10 <free>
=> 0x0000000000401c50 <+163>:   mov    -0x4(%rbp),%eax
   0x0000000000401c53 <+166>:   cltq
   0x0000000000401c55 <+168>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c5d <+176>:   lea    0xc08dc(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c64 <+183>:   movq   $0x0,(%rdx,%rax,1)
   0x0000000000401c6c <+191>:   mov    -0x8(%rbp),%eax
   0x0000000000401c6f <+194>:   cltq
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401c71 <+196>:   mov    %rax,%rdi
   0x0000000000401c74 <+199>:   callq  0x41a5d0 <malloc>
   0x0000000000401c79 <+204>:   mov    %rax,%rcx
   0x0000000000401c7c <+207>:   mov    -0x4(%rbp),%eax
   0x0000000000401c7f <+210>:   cltq
   0x0000000000401c81 <+212>:   lea    0x0(,%rax,8),%rdx
   0x0000000000401c89 <+220>:   lea    0xc08b0(%rip),%rax        # 0x4c2540 <pAllocBuf>
   0x0000000000401c90 <+227>:   mov    %rcx,(%rdx,%rax,1)
   0x0000000000401c94 <+231>:   jmpq   0x401bb5 <proc+8>
End of assembler dump.
```

```
(gdb) x/gx 0x4c2540
0x4c2540 <pAllocBuf>:   0x00007ff7c8062b10
```

10. Scaled indexing instruction **mov (%rdx,%rax,1),%rax** suggests that we have an array. Dump the first 1000 elements of array *pAllocBuf* (0x4c2540) found in *proc* function disassembly (this can be done in two different ways):

```
(gdb) print/x *0x4c2540@1000
$0 = {0xc8062b10, 0x7ff7, 0x0 <repeats 14 times>, 0xc809fac0, 0x7ff7, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0xd001a3f0, 0x7ff7, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0xc410eb00, 0x7ff7,
  0x0 <repeats 20 times>, 0xd402ba20, 0x7ff7, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0xd4081a10, 0x7ff7, 0x0, 0x0, 0x0, 0x0, 0xc80dabe0, 0x7ff7, 0x0 <repeats 22 times>, 0xd0047d50,
  0x7ff7, 0x0 <repeats 62 times>, 0xc800bb40, 0x7ff7, 0x0, 0x0, 0xd403af80, 0x7ff7, 0x0
<repeats 20 times>, 0xc8136e10, 0x7ff7, 0x0, 0x0, 0x0, 0x0, 0xd4011980, 0x7ff7, 0x0 <repeats 32
times>,
  0xc4106b00, 0x7ff7, 0xc40eb100, 0x7ff7, 0x0 <repeats 16 times>, 0xd401f6e0, 0x7ff7, 0x0, 0x0,
0xc8029560, 0x7ff7, 0xd4043cf0, 0x7ff7, 0x0 <repeats 48 times>, 0xc80e2760, 0x7ff7, 0x0, 0x0,
  0xc8082fa0, 0x7ff7, 0x0 <repeats 18 times>, 0xc40d0f70, 0x7ff7, 0x0 <repeats 36 times>,
0xc80e0fc0, 0x7ff7, 0x0, 0x0, 0x0, 0x0, 0xc813fdc0, 0x7ff7, 0xc40ddf70, 0x7ff7, 0x0, 0x0, 0x0,
0x0, 0x0,
  0x0, 0x0, 0x0, 0xc8102800, 0x7ff7, 0x0, 0x0...}
```

```
(gdb) print/x *(long *)0x4c2540@1000
$1 = {0x7ff7c8062b10, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x7ff7c809fac0, 0x0, 0x0, 0x0,
0x7ff7d001a3f0, 0x0, 0x0, 0x0, 0x0, 0x7ff7c410eb00, 0x0, 0x0,
  0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x7ff7d402ba20, 0x0, 0x0, 0x0, 0x0, 0x0,
0x7ff7d4081a10, 0x0, 0x0, 0x7ff7c80dabe0, 0x0 <repeats 11 times>,
  0x7ff7d0047d50, 0x0 <repeats 31 times>, 0x7ff7c800bb40, 0x0, 0x7ff7d403af80, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x7ff7c8136e10, 0x0,
  0x0, 0x7ff7d4011980, 0x0 <repeats 16 times>, 0x7ff7c4106b00, 0x7ff7c40eb100, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x7ff7d401f6e0, 0x0,
```

```
  0x7ff7c8029560, 0x7ff7d4043cf0, 0x0 <repeats 24 times>, 0x7ff7c80e2760, 0x0, 0x7ff7c8082fa0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
  0x7ff7c40d0f70, 0x0 <repeats 18 times>, 0x7ff7c80e0fc0, 0x0, 0x0, 0x7ff7c813fdc0,
0x7ff7c40ddf70, 0x0, 0x0, 0x0, 0x0, 0x7ff7c8102800, 0x0, 0x0, 0x0,
  0x0, 0x7ff7c8008660, 0x0 <repeats 18 times>, 0x7ff7c40c4a40, 0x0 <repeats 36 times>,
0x7ff7c4059ca0, 0x0 <repeats 15 times>, 0x7ff7c8056260,
  0x7ff7cc0703a0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x7ff7c40cc3c0, 0x0 <repeats 14
times>...}

(gdb) x/1000gx 0x4c2540
0x4c2540 <pAllocBuf>:   0x00007ff7c8062b10      0x0000000000000000
0x4c2550 <pAllocBuf+16>:        0x0000000000000000      0x0000000000000000
0x4c2560 <pAllocBuf+32>:        0x0000000000000000      0x0000000000000000
0x4c2570 <pAllocBuf+48>:        0x0000000000000000      0x0000000000000000
0x4c2580 <pAllocBuf+64>:        0x00007ff7c809fac0      0x0000000000000000
0x4c2590 <pAllocBuf+80>:        0x0000000000000000      0x0000000000000000
0x4c25a0 <pAllocBuf+96>:        0x00007ff7d001a3f0      0x0000000000000000
0x4c25b0 <pAllocBuf+112>:       0x0000000000000000      0x0000000000000000
0x4c25c0 <pAllocBuf+128>:       0x0000000000000000      0x00007ff7c410eb00
0x4c25d0 <pAllocBuf+144>:       0x0000000000000000      0x0000000000000000
0x4c25e0 <pAllocBuf+160>:       0x0000000000000000      0x0000000000000000
0x4c25f0 <pAllocBuf+176>:       0x0000000000000000      0x0000000000000000
0x4c2600 <pAllocBuf+192>:       0x0000000000000000      0x0000000000000000
0x4c2610 <pAllocBuf+208>:       0x0000000000000000      0x0000000000000000
0x4c2620 <pAllocBuf+224>:       0x00007ff7d402ba20      0x0000000000000000
0x4c2630 <pAllocBuf+240>:       0x0000000000000000      0x0000000000000000
0x4c2640 <pAllocBuf+256>:       0x0000000000000000      0x0000000000000000
0x4c2650 <pAllocBuf+272>:       0x00007ff7d4081a10      0x0000000000000000
0x4c2660 <pAllocBuf+288>:       0x0000000000000000      0x00007ff7c80dabe0
0x4c2670 <pAllocBuf+304>:       0x0000000000000000      0x0000000000000000
0x4c2680 <pAllocBuf+320>:       0x0000000000000000      0x0000000000000000
0x4c2690 <pAllocBuf+336>:       0x0000000000000000      0x0000000000000000
0x4c26a0 <pAllocBuf+352>:       0x0000000000000000      0x0000000000000000
0x4c26b0 <pAllocBuf+368>:       0x0000000000000000      0x0000000000000000
0x4c26c0 <pAllocBuf+384>:       0x0000000000000000      0x00007ff7d0047d50
0x4c26d0 <pAllocBuf+400>:       0x0000000000000000      0x0000000000000000
0x4c26e0 <pAllocBuf+416>:       0x0000000000000000      0x0000000000000000
0x4c26f0 <pAllocBuf+432>:       0x0000000000000000      0x0000000000000000
0x4c2700 <pAllocBuf+448>:       0x0000000000000000      0x0000000000000000
0x4c2710 <pAllocBuf+464>:       0x0000000000000000      0x0000000000000000
0x4c2720 <pAllocBuf+480>:       0x0000000000000000      0x0000000000000000
0x4c2730 <pAllocBuf+496>:       0x0000000000000000      0x0000000000000000
0x4c2740 <pAllocBuf+512>:       0x0000000000000000      0x0000000000000000
0x4c2750 <pAllocBuf+528>:       0x0000000000000000      0x0000000000000000
0x4c2760 <pAllocBuf+544>:       0x0000000000000000      0x0000000000000000
0x4c2770 <pAllocBuf+560>:       0x0000000000000000      0x0000000000000000
0x4c2780 <pAllocBuf+576>:       0x0000000000000000      0x0000000000000000
0x4c2790 <pAllocBuf+592>:       0x0000000000000000      0x0000000000000000
0x4c27a0 <pAllocBuf+608>:       0x0000000000000000      0x0000000000000000
0x4c27b0 <pAllocBuf+624>:       0x0000000000000000      0x0000000000000000
0x4c27c0 <pAllocBuf+640>:       0x0000000000000000      0x00007ff7c800bb40
0x4c27d0 <pAllocBuf+656>:       0x0000000000000000      0x00007ff7d403af80
0x4c27e0 <pAllocBuf+672>:       0x0000000000000000      0x0000000000000000
0x4c27f0 <pAllocBuf+688>:       0x0000000000000000      0x0000000000000000
0x4c2800 <pAllocBuf+704>:       0x0000000000000000      0x0000000000000000
0x4c2810 <pAllocBuf+720>:       0x0000000000000000      0x0000000000000000
0x4c2820 <pAllocBuf+736>:       0x0000000000000000      0x0000000000000000
0x4c2830 <pAllocBuf+752>:       0x00007ff7c8136e10      0x0000000000000000
0x4c2840 <pAllocBuf+768>:       0x0000000000000000      0x00007ff7d4011980
0x4c2850 <pAllocBuf+784>:       0x0000000000000000      0x0000000000000000
```

```
0x4c2860 <pAllocBuf+800>:       0x0000000000000000      0x0000000000000000
0x4c2870 <pAllocBuf+816>:       0x0000000000000000      0x0000000000000000
0x4c2880 <pAllocBuf+832>:       0x0000000000000000      0x0000000000000000
0x4c2890 <pAllocBuf+848>:       0x0000000000000000      0x0000000000000000
0x4c28a0 <pAllocBuf+864>:       0x0000000000000000      0x0000000000000000
0x4c28b0 <pAllocBuf+880>:       0x0000000000000000      0x0000000000000000
0x4c28c0 <pAllocBuf+896>:       0x0000000000000000      0x0000000000000000
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
```

Exercise A10 (A64, GDB)

Goal: Learn how to identify heap contention wait chains, synchronization issues, advanced disassembly, dump arrays.

Patterns: Double Free (Process Heap); High Contention (Process Heap); Wait Chain (General); Critical Region; Self-Diagnosis (User Mode).

1. Load *core.10881* dump file and *App10* executable from the A64/App10 directory:

```
~/ALCDA2/A64/App10$ gdb -c core.10881 -se App10
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App10...
(No debugging symbols found in App10)

warning: Can't open file /home/opc/ALCDA2/App10/App10 during file-backed mapping note
processing
[New LWP 10882]
[New LWP 10881]
[New LWP 10886]
[New LWP 10884]
[New LWP 10885]
[New LWP 10883]
Core was generated by `./App10'.
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x0000000000419a3c in _int_free ()
[Current thread is 1 (LWP 10882)]
```

2. Set logging to a file in case of lengthy output from some commands:

```
(gdb) set logging file App10.log

(gdb) set logging enabled on
Copying output to App10.log.
Copying debug output to App10.log.

(gdb) set style enabled off
```

3. Check all threads and identify problem top frames:

```
(gdb) info threads
  Id   Target Id         Frame
* 1    LWP 10882         0x0000000000419a3c in _int_free ()
  2    LWP 10881         0x000000000040ca44 in nanosleep ()
  3    LWP 10886         0x000000000040bcec in __lll_lock_wait_private ()
  4    LWP 10884         0x000000000040bcc0 in __lll_lock_wait_private ()
  5    LWP 10885         0x000000000040bcec in __lll_lock_wait_private ()
  6    LWP 10883         0x000000000040bcf0 in __lll_lock_wait_private ()
```

4. Check thread #3 and find where it was being executed:

```
(gdb) thread 3
[Switching to thread 3 (LWP 10886)]
#0  0x000000000040bcec in __lll_lock_wait_private ()
```

```
(gdb) bt
#0  0x000000000040bcec in __lll_lock_wait_private ()
#1  0x000000000041a02c in _int_free ()
#2  0x0000000000403254 in proc ()
#3  0x00000000004033a0 in bar_five ()
#4  0x00000000004033b4 in foo_five ()
#5  0x00000000004033cc in thread_five ()
#6  0x0000000000404cc4 in start_thread ()
#7  0x0000000000429c20 in thread_start ()
```

```
(gdb) disassemble proc
Dump of assembler code for function proc:
   0x00000000004031e8 <+0>:     stp     x29, x30, [sp, #-32]!
   0x00000000004031ec <+4>:     mov     x29, sp
   0x00000000004031f0 <+8>:     bl      0x40fadc <rand>
   0x00000000004031f4 <+12>:    mov     w1, #0x2710                     // #10000
   0x00000000004031f8 <+16>:    sdiv    w2, w0, w1
   0x00000000004031fc <+20>:    mov     w1, #0x2710                     // #10000
   0x0000000000403200 <+24>:    mul     w1, w2, w1
   0x0000000000403204 <+28>:    sub     w0, w0, w1
   0x0000000000403208 <+32>:    str     w0, [x29, #28]
   0x000000000040320c <+36>:    bl      0x40fadc <rand>
   0x0000000000403210 <+40>:    mov     w1, #0x2710                     // #10000
   0x0000000000403214 <+44>:    sdiv    w2, w0, w1
   0x0000000000403218 <+48>:    mov     w1, #0x2710                     // #10000
   0x000000000040321c <+52>:    mul     w1, w2, w1
   0x0000000000403220 <+56>:    sub     w0, w0, w1
   0x0000000000403224 <+60>:    str     w0, [x29, #24]
   0x0000000000403228 <+64>:    adrp    x0, 0x4d1000 <main_arena+2168>
   0x000000000040322c <+68>:    add     x0, x0, #0x668
   0x0000000000403230 <+72>:    ldrsw   x1, [x29, #28]
   0x0000000000403234 <+76>:    ldr     x0, [x0, x1, lsl #3]
   0x0000000000403238 <+80>:    cmp     x0, xzr
   0x000000000040323c <+84>:    b.eq    0x403264 <proc+124>  // b.none
   0x0000000000403240 <+88>:    adrp    x0, 0x4d1000 <main_arena+2168>
   0x0000000000403244 <+92>:    add     x0, x0, #0x668
   0x0000000000403248 <+96>:    ldrsw   x1, [x29, #28]
   0x000000000040324c <+100>:   ldr     x0, [x0, x1, lsl #3]
   0x0000000000403250 <+104>:   bl      0x41d5c8 <free>
   0x0000000000403254 <+108>:   adrp    x0, 0x4d1000 <main_arena+2168>
   0x0000000000403258 <+112>:   add     x0, x0, #0x668
   0x000000000040325c <+116>:   ldrsw   x1, [x29, #28]
   0x0000000000403260 <+120>:   str     xzr, [x0, x1, lsl #3]
```

```
   0x0000000000403264 <+124>:   ldrsw   x0, [x29, #24]
   0x0000000000403268 <+128>:   bl      0x41ca90 <malloc>
   0x000000000040326c <+132>:   mov     x2, x0
   0x0000000000403270 <+136>:   adrp    x0, 0x4d1000 <main_arena+2168>
   0x0000000000403274 <+140>:   add     x0, x0, #0x668
   0x0000000000403278 <+144>:   ldrsw   x1, [x29, #28]
   0x000000000040327c <+148>:   str     x2, [x0, x1, lsl #3]
   0x0000000000403280 <+152>:   b       0x4031f0 <proc+8>
End of assembler dump.
```

5. Check threads #4, #5, and #6, and find where it was being executed:

```
(gdb) thread 4
[Switching to thread 4 (LWP 10884)]
#0  0x000000000040bcc0 in __lll_lock_wait_private ()

(gdb) bt
#0  0x000000000040bcc0 in __lll_lock_wait_private ()
#1  0x000000000041a02c in _int_free ()
#2  0x0000000000403254 in proc ()
#3  0x0000000000403318 in bar_three ()
#4  0x000000000040332c in foo_three ()
#5  0x0000000000403344 in thread_three ()
#6  0x0000000000404cc4 in start_thread ()
#7  0x0000000000429c20 in thread_start ()

(gdb) thread 5
[Switching to thread 5 (LWP 10885)]
#0  0x000000000040bcec in __lll_lock_wait_private ()

(gdb) bt
#0  0x000000000040bcec in __lll_lock_wait_private ()
#1  0x000000000041a02c in _int_free ()
#2  0x0000000000403254 in proc ()
#3  0x000000000040335c in bar_four ()
#4  0x0000000000403370 in foo_four ()
#5  0x0000000000403388 in thread_four ()
#6  0x0000000000404cc4 in start_thread ()
#7  0x0000000000429c20 in thread_start ()

(gdb) thread 6
[Switching to thread 6 (LWP 10883)]
#0  0x000000000040bcf0 in __lll_lock_wait_private ()

(gdb) bt
#0  0x000000000040bcf0 in __lll_lock_wait_private ()
#1  0x000000000041a02c in _int_free ()
#2  0x0000000000403254 in proc ()
#3  0x00000000004032d4 in bar_two ()
#4  0x00000000004032e8 in foo_two ()
#5  0x0000000000403300 in thread_two ()
#6  0x0000000000404cc4 in start_thread ()
#7  0x0000000000429c20 in thread_start ()
```

Note: We see that all waiting threads are the same in their return addresses from *free*.

6. Check thread #1 and find where it was being executed:

```
(gdb) thread 1
[Switching to thread 1 (LWP 10882)]
#0  0x0000000000419a3c in _int_free ()

(gdb) bt
#0  0x0000000000419a3c in _int_free ()
#1  0x0000000000403254 in proc ()
#2  0x0000000000403290 in bar_one ()
#3  0x00000000004032a4 in foo_one ()
#4  0x00000000004032bc in thread_one ()
#5  0x0000000000404cc4 in start_thread ()
#6  0x0000000000429c20 in thread_start ()
```

Note: We see that it also has the same return addresses from *free*. It means all these threads are contending for the same free function. However, thread #1 got a segmentation fault signal. Since *free* calls were done from the same *proc* function location, we suspect a double free:

```
(gdb) x/i 0x0000000000419a3c
=> 0x419a3c <_int_free+268>:    ldr     x2, [x19, #8]

(gdb) x $x19+8
   0xfffffffffc12e28:  Cannot access memory at address 0xfffffffffc12e28
```

7. Check the address that was being freed:

```
(gdb) disassemble proc
Dump of assembler code for function proc:
   0x00000000004031e8 <+0>:     stp     x29, x30, [sp, #-32]!
   0x00000000004031ec <+4>:     mov     x29, sp
   0x00000000004031f0 <+8>:     bl      0x40fadc <rand>
   0x00000000004031f4 <+12>:    mov     w1, #0x2710                     // #10000
   0x00000000004031f8 <+16>:    sdiv    w2, w0, w1
   0x00000000004031fc <+20>:    mov     w1, #0x2710                     // #10000
   0x0000000000403200 <+24>:    mul     w1, w2, w1
   0x0000000000403204 <+28>:    sub     w0, w0, w1
   0x0000000000403208 <+32>:    str     w0, [x29, #28]
   0x000000000040320c <+36>:    bl      0x40fadc <rand>
   0x0000000000403210 <+40>:    mov     w1, #0x2710                     // #10000
   0x0000000000403214 <+44>:    sdiv    w2, w0, w1
   0x0000000000403218 <+48>:    mov     w1, #0x2710                     // #10000
   0x000000000040321c <+52>:    mul     w1, w2, w1
   0x0000000000403220 <+56>:    sub     w0, w0, w1
   0x0000000000403224 <+60>:    str     w0, [x29, #24]
   0x0000000000403228 <+64>:    adrp    x0, 0x4d1000 <main_arena+2168>
   0x000000000040322c <+68>:    add     x0, x0, #0x668
   0x0000000000403230 <+72>:    ldrsw   x1, [x29, #28]
   0x0000000000403234 <+76>:    ldr     x0, [x0, x1, lsl #3]
   0x0000000000403238 <+80>:    cmp     x0, xzr
   0x000000000040323c <+84>:    b.eq    0x403264 <proc+124>  // b.none
   0x0000000000403240 <+88>:    adrp    x0, 0x4d1000 <main_arena+2168>
   0x0000000000403244 <+92>:    add     x0, x0, #0x668
   0x0000000000403248 <+96>:    ldrsw   x1, [x29, #28]
   0x000000000040324c <+100>:   ldr     x0, [x0, x1, lsl #3]
   0x0000000000403250 <+104>:   bl      0x41d5c8 <free>
   0x0000000000403254 <+108>:   adrp    x0, 0x4d1000 <main_arena+2168>
   0x0000000000403258 <+112>:   add     x0, x0, #0x668
```

```
   0x000000000040325c <+116>:   ldrsw   x1, [x29, #28]
   0x0000000000403260 <+120>:   str     xzr, [x0, x1, lsl #3]
   0x0000000000403264 <+124>:   ldrsw   x0, [x29, #24]
   0x0000000000403268 <+128>:   bl      0x41ca90 <malloc>
   0x000000000040326c <+132>:   mov     x2, x0
   0x0000000000403270 <+136>:   adrp    x0, 0x4d1000 <main_arena+2168>
   0x0000000000403274 <+140>:   add     x0, x0, #0x668
   0x0000000000403278 <+144>:   ldrsw   x1, [x29, #28]
   0x000000000040327c <+148>:   str     x2, [x0, x1, lsl #3]
   0x0000000000403280 <+152>:   b       0x4031f0 <proc+8>
End of assembler dump.
```

Note: We see that we pass some location from the index array, which starts from the address **0x4d1000 + 0x668**:

```
(gdb) x/gx 0x4d1000 + 0x668
0x4d1668 <pAllocBuf>:   0x0000fffba93b4e00
```

8. Dump the first 100 elements of the identified array:

```
(gdb) print/x *0x4d1668@1000
$0 = {0xa93b4e00, 0xfffb, 0xa90c1af0, 0xfffb, 0xaa939280, 0xfffb, 0xaae8e2d0, 0xfffb,
0xa98aa350, 0xfffb, 0xa8fd48b0,
  0xfffb, 0xa912d220, 0xfffb, 0xaa420ad0, 0xfffb, 0xaae7f930, 0xfffb, 0xa87c2b20, 0xfffb,
0xa83d3e10, 0xfffb, 0x0, 0x0,
  0xaa65da80, 0xfffb, 0xaa869280, 0xfffb, 0xaa5c5f80, 0xfffb, 0xa8459f40, 0xfffb, 0xa81e4e30,
0xfffb, 0xa810f820, 0xfffb,
  0xa9e857a0, 0xfffb, 0xa9deacf0, 0xfffb, 0xa8933250, 0xfffb, 0xa8306260, 0xfffb, 0xaa9303e0,
0xfffb, 0xa85ef530, 0xfffb,
  0xaaefa020, 0xfffb, 0xa8b6f670, 0xfffb, 0xaadd5320, 0xfffb, 0xaaab54f0, 0xfffb, 0xa8ce7fc0,
0xfffb, 0xa8087370, 0xfffb,
  0xa823ab80, 0xfffb, 0xa8927f20, 0xfffb, 0xa9bc66d0, 0xfffb, 0xaa81d670, 0xfffb, 0xa8478270,
0xfffb, 0xa92edf60, 0xfffb,
  0xa8065330, 0xfffb, 0xa893faa0, 0xfffb, 0xaa0fe190, 0xfffb, 0xa8fc1450, 0xfffb, 0xa84c8f20,
0xfffb, 0xa8b79280, 0xfffb,
  0xaabe7040, 0xfffb, 0xa83d9110, 0xfffb, 0xa97dade0, 0xfffb, 0xa8fd0e80, 0xfffb, 0xaa20dc20,
0xfffb, 0xaa56f580, 0xfffb,
  0xa9724a30, 0xfffb, 0xaa241aa0, 0xfffb, 0xa8049e50, 0xfffb, 0xa82f3e40, 0xfffb, 0xa93b3cf0,
0xfffb, 0xaa9978d0, 0xfffb,
  0xa9223440, 0xfffb, 0x0, 0x0, 0xa9cffd00, 0xfffb, 0xa9e323a0, 0xfffb, 0xa9230070, 0xfffb,
0xa8cfc710, 0xfffb, 0xa9e48810,
  0xfffb, 0xa8474ea0, 0xfffb, 0xa9557cf0, 0xfffb, 0xa86617f0, 0xfffb, 0xa8737030, 0xfffb,
0xa92da510, 0xfffb, 0x0, 0x0,
  0xaa1977d0, 0xfffb, 0xa9287350, 0xfffb, 0xa8071640, 0xfffb, 0xa9d15240, 0xfffb, 0xa8ec3f00,
0xfffb, 0xa88273b0, 0xfffb,
  0xbc00d7b0, 0xfffb, 0x0, 0x0, 0xa8d41340, 0xfffb, 0xaa48f100, 0xfffb, 0xa8f9b680, 0xfffb,
0xa8badda0, 0xfffb, 0xa91d85e0,
  0xfffb, 0xa81f7730, 0xfffb, 0xa91ca3f0, 0xfffb, 0xa9688d40, 0xfffb, 0xaa760170, 0xfffb,
0xa878cbd0, 0xfffb, 0xa8edbc30,
  0xfffb, 0xa877fe70, 0xfffb, 0xa861ca80, 0xfffb, 0xa9003de0, 0xfffb, 0xa8cd3b50, 0xfffb,
0xaa310670, 0xfffb, 0xa91095f0,
  0xfffb, 0xa840f270, 0xfffb, 0xa8b094f0, 0xfffb, 0xa84334f0, 0xfffb, 0xaa272b90, 0xfffb,
0xa8e280e0, 0xfffb, 0xa8c32df0,
  0xfffb, 0xaa81b390, 0xfffb, 0xa888cb80, 0xfffb...}

(gdb) print/x *(long *)0x4d1668@1000
$1 = {0xfffba93b4e00, 0xfffba90c1af0, 0xfffbaa939280, 0xfffbaae8e2d0, 0xfffba98aa350,
0xfffba8fd48b0, 0xfffba912d220,
  0xfffbaa420ad0, 0xfffbaae7f930, 0xfffba87c2b20, 0xfffba83d3e10, 0x0, 0xfffbaa65da80,
0xfffbaa869280, 0xfffbaa5c5f80,
```

```
  0xfffba8459f40, 0xfffba81e4e30, 0xfffba810f820, 0xfffba9e857a0, 0xfffba9deacf0,
0xfffba8933250, 0xfffba8306260,
  0xfffbaa9303e0, 0xfffba85ef530, 0xfffbaaefa020, 0xfffba8b6f670, 0xfffbaadd5320,
0xfffbaaab54f0, 0xfffba8ce7fc0,
  0xfffba8087370, 0xfffba823ab80, 0xfffba8927f20, 0xfffba9bc66d0, 0xfffbaa81d670,
0xfffba8478270, 0xfffba92edf60,
  0xfffba8065330, 0xfffba893faa0, 0xfffbaa0fe190, 0xfffba8fc1450, 0xfffba84c8f20,
0xfffba8b79280, 0xfffbaabe7040,
  0xfffba83d9110, 0xfffba97dade0, 0xfffba8fd0e80, 0xfffbaa20dc20, 0xfffbaa56f580,
0xfffba9724a30, 0xfffbaa241aa0,
  0xfffba8049e50, 0xfffba82f3e40, 0xfffba93b3cf0, 0xfffbaa9978d0, 0xfffba9223440, 0x0,
0xfffba9cffd00, 0xfffba9e323a0,
  0xfffba9230070, 0xfffba8cfc710, 0xfffba9e48810, 0xfffba8474ea0, 0xfffba9557cf0,
0xfffba86617f0, 0xfffba8737030,
  0xfffba92da510, 0x0, 0xfffbaa1977d0, 0xfffba9287350, 0xfffba8071640, 0xfffba9d15240,
0xfffba8ec3f00, 0xfffba88273b0,
  0xfffbbc00d7b0, 0x0, 0xfffba8d41340, 0xfffbaa48f100, 0xfffba8f9b680, 0xfffba8badda0,
0xfffba91d85e0, 0xfffba81f7730,
  0xfffba91ca3f0, 0xfffba9688d40, 0xfffbaa760170, 0xfffba878cbd0, 0xfffba8edbc30,
0xfffba877fe70, 0xfffba861ca80,
  0xfffba9003de0, 0xfffba8cd3b50, 0xfffbaa310670, 0xfffba91095f0, 0xfffba840f270,
0xfffba8b094f0, 0xfffba84334f0,
  0xfffbaa272b90, 0xfffba8e280e0, 0xfffba8c32df0, 0xfffbaa81b390, 0xfffba888cb80, 0x0,
0xfffbaa5009a0, 0xfffba8d5aaf0,
  0xfffba895fa50, 0xfffba8726760, 0xfffbaa3bdf80, 0xfffba8088d40, 0xfffba8e63930,
0xfffba8726130, 0xfffbaad1af40,
  0xfffbaa300ed0, 0xfffba992f670, 0xfffbaa194090, 0xfffba9c5a300, 0xfffbaa5ea990,
0xfffba92ab6d0, 0xfffbaa2f3700,
  0xfffba83e0b30, 0xfffba8692620, 0xfffba90d8260, 0xfffbaa62d8a0, 0xfffba939e180,
0xfffba8596f00, 0xfffba9419c70,
  0xfffbaa59f5a0, 0xfffbaa01ffd0, 0xfffba85c1260, 0xfffbaa1a6250, 0xfffba8b66780,
0xfffbaaa696f0, 0xfffba86e3f60,
  0xfffba8bf7240, 0xfffbaa1e3ae0, 0xfffba91c0f30, 0xfffba9fc6b10, 0xfffbaa6e4700,
0xfffbaa660770, 0xfffba9a92e80,
  0xfffbaaac2fc0, 0xfffba8ba0340, 0xfffba9cf6240, 0xfffba8451290, 0xfffba88880c0,
0xfffba92517e0, 0xfffbaae65de0,
  0xfffba84ab520, 0xfffba84d0b70, 0xfffba8264490, 0xfffbaa4749b0, 0xfffba9546660,
0xfffba8a011c0, 0xfffba938f9e0,
  0xfffbaaebe240, 0xfffba9999440, 0xfffba9a72850, 0xfffbaab01a00, 0xfffbaa6795b0,
0xfffbaae9a180, 0xfffbaa39e590,
  0xfffbaa08c140, 0xfffba922c030, 0xfffba9cd6540, 0xfffba81cc7d0, 0xfffba816e2b0,
0xfffbaa8a2070, 0x0, 0xfffbaa2f7740,
  0xfffba98fa130, 0xfffba9811090, 0xfffba85ca940, 0xfffba94f0ed0, 0xfffba922a680,
0xfffba9a709d0, 0xfffba8dc52e0,
  0xfffba9e61de0, 0xfffba856afe0, 0xfffba823e260, 0xfffba896e350, 0xfffba96f19a0,
0xfffba902b650, 0xfffba8095120,
  0xfffba89cc910, 0xfffbaac57210, 0xfffbaa78a580, 0xfffba8bc4120, 0xfffba97b25d0,
0xfffbaa243c10, 0x0, 0xfffbaa7ba1a0,
  0xfffba8b94d90, 0xfffba84977b0, 0xfffba82c4910, 0xfffbaad189f0, 0xfffba8f72680,
0xfffba998bdc0, 0xfffba8219e10,
  0xfffba90c16c0, 0xfffba9535ab0, 0xfffba8a68630, 0xfffba87432d0...}

(gdb) x/1000gx 0x4d1668
0x4d1668 <pAllocBuf>:   0x0000fffba93b4e00      0x0000fffba90c1af0
0x4d1678 <pAllocBuf+16>:        0x0000fffbaa939280      0x0000fffbaae8e2d0
0x4d1688 <pAllocBuf+32>:        0x0000fffba98aa350      0x0000fffba8fd48b0
0x4d1698 <pAllocBuf+48>:        0x0000fffba912d220      0x0000fffbaa420ad0
0x4d16a8 <pAllocBuf+64>:        0x0000fffbaae7f930      0x0000fffba87c2b20
0x4d16b8 <pAllocBuf+80>:        0x0000fffba83d3e10      0x0000000000000000
0x4d16c8 <pAllocBuf+96>:        0x0000fffbaa65da80      0x0000fffbaa869280
0x4d16d8 <pAllocBuf+112>:       0x0000fffbaa5c5f80      0x0000fffba8459f40
```

```
0x4d16e8 <pAllocBuf+128>:       0x0000fffba81e4e30      0x0000fffba810f820
0x4d16f8 <pAllocBuf+144>:       0x0000fffba9e857a0      0x0000fffba9deacf0
0x4d1708 <pAllocBuf+160>:       0x0000fffba8933250      0x0000fffba8306260
0x4d1718 <pAllocBuf+176>:       0x0000fffbaa9303e0      0x0000fffba85ef530
0x4d1728 <pAllocBuf+192>:       0x0000fffbaaefa020      0x0000fffba8b6f670
0x4d1738 <pAllocBuf+208>:       0x0000fffbaadd5320      0x0000fffbaaab54f0
0x4d1748 <pAllocBuf+224>:       0x0000fffba8ce7fc0      0x0000fffba8087370
0x4d1758 <pAllocBuf+240>:       0x0000fffba823ab80      0x0000fffba8927f20
0x4d1768 <pAllocBuf+256>:       0x0000fffba9bc66d0      0x0000fffbaa81d670
0x4d1778 <pAllocBuf+272>:       0x0000fffba8478270      0x0000fffba92edf60
0x4d1788 <pAllocBuf+288>:       0x0000fffba8065330      0x0000fffba893faa0
0x4d1798 <pAllocBuf+304>:       0x0000fffbaa0fe190      0x0000fffba8fc1450
0x4d17a8 <pAllocBuf+320>:       0x0000fffba84c8f20      0x0000fffba8b79280
0x4d17b8 <pAllocBuf+336>:       0x0000fffbaabe7040      0x0000fffba83d9110
0x4d17c8 <pAllocBuf+352>:       0x0000fffba97dade0      0x0000fffba8fd0e80
0x4d17d8 <pAllocBuf+368>:       0x0000fffbaa20dc20      0x0000fffbaa56f580
0x4d17e8 <pAllocBuf+384>:       0x0000fffba9724a30      0x0000fffbaa241aa0
0x4d17f8 <pAllocBuf+400>:       0x0000fffba8049e50      0x0000fffba82f3e40
0x4d1808 <pAllocBuf+416>:       0x0000fffba93b3cf0      0x0000fffbaa9978d0
0x4d1818 <pAllocBuf+432>:       0x0000fffba9223440      0x0000000000000000
0x4d1828 <pAllocBuf+448>:       0x0000fffba9cffd00      0x0000fffba9e323a0
0x4d1838 <pAllocBuf+464>:       0x0000fffba9230070      0x0000fffba8cfc710
0x4d1848 <pAllocBuf+480>:       0x0000fffba9e48810      0x0000fffba8474ea0
0x4d1858 <pAllocBuf+496>:       0x0000fffba9557cf0      0x0000fffba86617f0
0x4d1868 <pAllocBuf+512>:       0x0000fffba8737030      0x0000fffba92da510
0x4d1878 <pAllocBuf+528>:       0x0000000000000000      0x0000fffbaa1977d0
0x4d1888 <pAllocBuf+544>:       0x0000fffba9287350      0x0000fffba8071640
0x4d1898 <pAllocBuf+560>:       0x0000fffba9d15240      0x0000fffba8ec3f00
0x4d18a8 <pAllocBuf+576>:       0x0000fffba88273b0      0x0000fffbbc00d7b0
0x4d18b8 <pAllocBuf+592>:       0x0000000000000000      0x0000fffba8d41340
0x4d18c8 <pAllocBuf+608>:       0x0000fffbaa48f100      0x0000fffba8f9b680
0x4d18d8 <pAllocBuf+624>:       0x0000fffba8badda0      0x0000fffba91d85e0
0x4d18e8 <pAllocBuf+640>:       0x0000fffba81f7730      0x0000fffba91ca3f0
0x4d18f8 <pAllocBuf+656>:       0x0000fffba9688d40      0x0000fffbaa760170
0x4d1908 <pAllocBuf+672>:       0x0000fffba878cbd0      0x0000fffba8edbc30
0x4d1918 <pAllocBuf+688>:       0x0000fffba877fe70      0x0000fffba861ca80
0x4d1928 <pAllocBuf+704>:       0x0000fffba9003de0      0x0000fffba8cd3b50
0x4d1938 <pAllocBuf+720>:       0x0000fffbaa310670      0x0000fffba91095f0
0x4d1948 <pAllocBuf+736>:       0x0000fffba840f270      0x0000fffba8b094f0
0x4d1958 <pAllocBuf+752>:       0x0000fffba84334f0      0x0000fffbaa272b90
0x4d1968 <pAllocBuf+768>:       0x0000fffba8e280e0      0x0000fffba8c32df0
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
```

Exercise A10 (A64, WinDbg Preview)

Goal: Learn how to identify heap contention wait chains, synchronization issues, advanced disassembly, dump arrays.

Patterns: Double Free (Process Heap); High Contention (Process Heap); Wait Chain (General); Critical Region; Self-Diagnosis (User Mode).

1. Launch WinDbg Preview.

2. Load *core.10881* dump file from the A64\App10 folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App10\core.10881]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
(2a81.2a82): Signal SIGSEGV (Segmentation fault) code SEGV_MAPERR (Address not mapped to
object) at 0xfffffffffc12e28*** WARNING: Unable to verify timestamp for App10
App10+0x19a3c:
00000000`00419a3c ?? ???
```

3. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App10\App10.log
Opened log file 'C:\ALCDA2\A64\App10\App10.log'
```

4. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App10\
Symbol search path is: srv*;C:\ALCDA2\A64\App10\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app10\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App10\
*** WARNING: Unable to verify timestamp for App10
```

```
0:000> .reload
..
*** WARNING: Unable to verify timestamp for App10

************* Symbol Loading Error Summary **************
Module name            Error
App10                  The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

5. Check all threads and identify problem top frames:

```
0:000> ~*k 1

Unable to get thread data for thread 0
.  0  Id: 2a81.2a82 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffb`c684e780 00000000`00403254     App10!int_free+0x10c

Unable to get thread data for thread 1
   1  Id: 2a81.2a81 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`c53dc670 00000000`00424d74     App10!_libc_nanosleep+0x24

Unable to get thread data for thread 2
   2  Id: 2a81.2a86 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffb`c480e780 00000000`0041a02c     App10!_lll_lock_wait_private+0x5c

Unable to get thread data for thread 3
   3  Id: 2a81.2a84 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffb`c582e780 00000000`0041a02c     App10!_lll_lock_wait_private+0x30

Unable to get thread data for thread 4
   4  Id: 2a81.2a85 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffb`c501e780 00000000`0041a02c     App10!_lll_lock_wait_private+0x5c

Unable to get thread data for thread 5
   5  Id: 2a81.2a83 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffb`c603e780 00000000`0041a02c     App10!_lll_lock_wait_private+0x60
```

6. Check thread #2 and find where it was being executed:

```
0:000> ~2s
App10!_lll_lock_wait_private+0x5c:
00000000`0040bcec d4000001 svc         #0
```

```
0:002> k
 # Child-SP          RetAddr               Call Site
00 0000fffb`c480e780 00000000`0041a02c     App10!_lll_lock_wait_private+0x5c
01 0000fffb`c480e780 00000000`00403254     App10!int_free+0x6fc
02 0000fffb`c480e800 00000000`004033a0     App10!proc+0x6c
03 0000fffb`c480e820 00000000`004033b4     App10!bar_five+0xc
04 0000fffb`c480e830 00000000`004033cc     App10!foo_five+0xc
05 0000fffb`c480e840 00000000`00404cc4     App10!thread_five+0x10
06 0000fffb`c480e860 00000000`00429c20     App10!start_thread+0xb4
07 0000fffb`c480e990 ffffffff`ffffffff     App10!thread_start+0x30
08 0000fffb`c480e990 00000000`00000000     0xffffffff`ffffffff

0:002> uf proc
App10!proc:
00000000`004031e8 a9be7bfd stp         fp,lr,[sp,#-0x20]!
00000000`004031ec 910003fd mov         fp,sp

App10!proc+0x8:
00000000`004031f0 9400323b bl          App10!rand (00000000`0040fadc)
00000000`004031f4 5284e201 mov         w1,#0x2710
00000000`004031f8 1ac10c02 sdiv        w2,w0,w1
00000000`004031fc 5284e201 mov         w1,#0x2710
00000000`00403200 1b017c41 mul         w1,w2,w1
00000000`00403204 4b010000 sub         w0,w0,w1
00000000`00403208 b9001fa0 str         w0,[fp,#0x1C]
00000000`0040320c 94003234 bl          App10!rand (00000000`0040fadc)
00000000`00403210 5284e201 mov         w1,#0x2710
00000000`00403214 1ac10c02 sdiv        w2,w0,w1
00000000`00403218 5284e201 mov         w1,#0x2710
00000000`0040321c 1b017c41 mul         w1,w2,w1
00000000`00403220 4b010000 sub         w0,w0,w1
00000000`00403224 b9001ba0 str         w0,[fp,#0x18]
00000000`00403228 d0000660 adrp        x0,App10!main_arena+0x878 (00000000`004d1000)
00000000`0040322c 9119a000 add         x0,x0,#0x668
00000000`00403230 b9801fa1 ldrsw       x1,[fp,#0x1C]
00000000`00403234 f8617800 ldr         x0,[x0,x1 lsl #3]
00000000`00403238 eb1f001f cmp         x0,xzr
00000000`0040323c 54000140 beq         App10!proc+0x7c (00000000`00403264)  Branch

App10!proc+0x58:
00000000`00403240 d0000660 adrp        x0,App10!main_arena+0x878 (00000000`004d1000)
00000000`00403244 9119a000 add         x0,x0,#0x668
00000000`00403248 b9801fa1 ldrsw       x1,[fp,#0x1C]
00000000`0040324c f8617800 ldr         x0,[x0,x1 lsl #3]
00000000`00403250 940068de bl          App10!_cfree (00000000`0041d5c8)
00000000`00403254 d0000660 adrp        x0,App10!main_arena+0x878 (00000000`004d1000)
00000000`00403258 9119a000 add         x0,x0,#0x668
00000000`0040325c b9801fa1 ldrsw       x1,[fp,#0x1C]
00000000`00403260 f821781f str         xzr,[x0,x1 lsl #3]

App10!proc+0x7c:
00000000`00403264 b9801ba0 ldrsw       x0,[fp,#0x18]
00000000`00403268 9400660a bl          App10!malloc (00000000`0041ca90)
00000000`0040326c aa0003e2 mov         x2,x0
00000000`00403270 d0000660 adrp        x0,App10!main_arena+0x878 (00000000`004d1000)
00000000`00403274 9119a000 add         x0,x0,#0x668
00000000`00403278 b9801fa1 ldrsw       x1,[fp,#0x1C]
00000000`0040327c f8217802 str         x2,[x0,x1 lsl #3]
00000000`00403280 17ffffdc b           App10!proc+0x8 (00000000`004031f0)  Branch
```

7. Check other remaining threads #3, #4, and #5, and find where they were being executed:

```
0:002> ~3k
 # Child-SP          RetAddr               Call Site
00 0000fffb`c582e780 00000000`0041a02c     App10!_lll_lock_wait_private+0x30
01 0000fffb`c582e780 00000000`00403254     App10!int_free+0x6fc
02 0000fffb`c582e800 00000000`00403318     App10!proc+0x6c
03 0000fffb`c582e820 00000000`0040332c     App10!bar_three+0xc
04 0000fffb`c582e830 00000000`00403344     App10!foo_three+0xc
05 0000fffb`c582e840 00000000`00404cc4     App10!thread_three+0x10
06 0000fffb`c582e860 00000000`00429c20     App10!start_thread+0xb4
07 0000fffb`c582e990 ffffffff`ffffffff     App10!thread_start+0x30
08 0000fffb`c582e990 00000000`00000000     0xffffffff`ffffffff

0:002> ~4k
 # Child-SP          RetAddr               Call Site
00 0000fffb`c501e780 00000000`0041a02c     App10!_lll_lock_wait_private+0x5c
01 0000fffb`c501e780 00000000`00403254     App10!int_free+0x6fc
02 0000fffb`c501e800 00000000`0040335c     App10!proc+0x6c
03 0000fffb`c501e820 00000000`00403370     App10!bar_four+0xc
04 0000fffb`c501e830 00000000`00403388     App10!foo_four+0xc
05 0000fffb`c501e840 00000000`00404cc4     App10!thread_four+0x10
06 0000fffb`c501e860 00000000`00429c20     App10!start_thread+0xb4
07 0000fffb`c501e990 ffffffff`ffffffff     App10!thread_start+0x30
08 0000fffb`c501e990 00000000`00000000     0xffffffff`ffffffff

0:002> ~5k
 # Child-SP          RetAddr               Call Site
00 0000fffb`c603e780 00000000`0041a02c     App10!_lll_lock_wait_private+0x60
01 0000fffb`c603e780 00000000`00403254     App10!int_free+0x6fc
02 0000fffb`c603e800 00000000`004032d4     App10!proc+0x6c
03 0000fffb`c603e820 00000000`004032e8     App10!bar_two+0xc
04 0000fffb`c603e830 00000000`00403300     App10!foo_two+0xc
05 0000fffb`c603e840 00000000`00404cc4     App10!thread_two+0x10
06 0000fffb`c603e860 00000000`00429c20     App10!start_thread+0xb4
07 0000fffb`c603e990 ffffffff`ffffffff     App10!thread_start+0x30
08 0000fffb`c603e990 00000000`00000000     0xffffffff`ffffffff
```

Note: We see that all waiting threads are the same in their return addresses from *free*.

8. Check thread #0 and find where it was being executed:

```
0:002> ~0s
App10!int_free+0x10c:
00000000`00419a3c f9400662 ldr         x2,[x19,#8]

0:000> k
 # Child-SP          RetAddr               Call Site
00 0000fffb`c684e780 00000000`00403254     App10!int_free+0x10c
01 0000fffb`c684e800 00000000`00403290     App10!proc+0x6c
02 0000fffb`c684e820 00000000`004032a4     App10!bar_one+0xc
03 0000fffb`c684e830 00000000`004032bc     App10!foo_one+0xc
04 0000fffb`c684e840 00000000`00404cc4     App10!thread_one+0x10
05 0000fffb`c684e860 00000000`00429c20     App10!start_thread+0xb4
06 0000fffb`c684e990 ffffffff`ffffffff     App10!thread_start+0x30
07 0000fffb`c684e990 00000000`00000000     0xffffffff`ffffffff
```

Note: We see that it also has the same return addresses from *free*. It means all these threads are contending for the same free function. However, thread #0 got a segmentation fault signal. Since *free* calls were done from the same *proc* function location, we suspect a double free:

```
0:000> r
 x0=0000fffb00000000   x1=0000000000000000   x2=0000000000000002   x3=0000000000000000
 x4=0000fffba8000020   x5=0000000000000002   x6=00000000004d1560   x7=000000003d64234e
 x8=0000000000000062   x9=0000fffbc0000690  x10=0000000000000068  x11=0000fffbc00008d0
x12=0000000000000007  x13=0000000000000000  x14=0000000000000004  x15=0000000000000000
x16=0000000000000001  x17=00000000004d0788  x18=0000000000000d18  x19=ffffffffffc12e20
x20=0000000000000e60  x21=0000fffba8000020  x22=0000fffba8903120  x23=0000000000000030
x24=0000fffbc684f770  x25=0000000000000000  x26=00000000004eb1c8  x27=00000000004e9000
x28=0000000000810000   fp=0000fffbc684e780   lr=000000000041a02c   sp=0000fffbc684e780
 pc=0000000000419a3c  psr=80001000 N--- EL0
App10!int_free+0x10c:
00000000`00419a3c f9400662 ldr         x2,[x19,#8]

0:000> dp x19+8
ffffffff`ffc12e28  ????????`???????? ????????`????????
ffffffff`ffc12e38  ????????`???????? ????????`????????
ffffffff`ffc12e48  ????????`???????? ????????`????????
ffffffff`ffc12e58  ????????`???????? ????????`????????
ffffffff`ffc12e68  ????????`???????? ????????`????????
ffffffff`ffc12e78  ????????`???????? ????????`????????
ffffffff`ffc12e88  ????????`???????? ????????`????????
ffffffff`ffc12e98  ????????`???????? ????????`????????
```

9. Check the address that was being freed:

```
0:000> uf proc
App10!proc:
00000000`004031e8 a9be7bfd stp         fp,lr,[sp,#-0x20]!
00000000`004031ec 910003fd mov         fp,sp

App10!proc+0x8:
00000000`004031f0 9400323b bl          App10!rand (00000000`0040fadc)
00000000`004031f4 5284e201 mov         w1,#0x2710
00000000`004031f8 1ac10c02 sdiv        w2,w0,w1
00000000`004031fc 5284e201 mov         w1,#0x2710
00000000`00403200 1b017c41 mul         w1,w2,w1
00000000`00403204 4b010000 sub         w0,w0,w1
00000000`00403208 b9001fa0 str         w0,[fp,#0x1C]
00000000`0040320c 94003234 bl          App10!rand (00000000`0040fadc)
00000000`00403210 5284e201 mov         w1,#0x2710
00000000`00403214 1ac10c02 sdiv        w2,w0,w1
00000000`00403218 5284e201 mov         w1,#0x2710
00000000`0040321c 1b017c41 mul         w1,w2,w1
00000000`00403220 4b010000 sub         w0,w0,w1
00000000`00403224 b9001ba0 str         w0,[fp,#0x18]
00000000`00403228 d0000660 adrp        x0,App10!main_arena+0x878 (00000000`004d1000)
00000000`0040322c 9119a000 add         x0,x0,#0x668
00000000`00403230 b9801fa1 ldrsw       x1,[fp,#0x1C]
00000000`00403234 f8617800 ldr         x0,[x0,x1 lsl #3]
00000000`00403238 eb1f001f cmp         x0,xzr
00000000`0040323c 54000140 beq         App10!proc+0x7c (00000000`00403264)  Branch
```

```
App10!proc+0x58:
00000000`00403240 d0000660 adrp        x0,App10!main_arena+0x878 (00000000`004d1000)
00000000`00403244 9119a000 add         x0,x0,#0x668
00000000`00403248 b9801fa1 ldrsw       x1,[fp,#0x1C]
00000000`0040324c f8617800 ldr         x0,[x0,x1 lsl #3]
00000000`00403250 940068de bl          App10!_cfree (00000000`0041d5c8)
00000000`00403254 d0000660 adrp        x0,App10!main_arena+0x878 (00000000`004d1000)
00000000`00403258 9119a000 add         x0,x0,#0x668
00000000`0040325c b9801fa1 ldrsw       x1,[fp,#0x1C]
00000000`00403260 f821781f str         xzr,[x0,x1 lsl #3]

App10!proc+0x7c:
00000000`00403264 b9801ba0 ldrsw       x0,[fp,#0x18]
00000000`00403268 9400660a bl          App10!malloc (00000000`0041ca90)
00000000`0040326c aa0003e2 mov         x2,x0
00000000`00403270 d0000660 adrp        x0,App10!main_arena+0x878 (00000000`004d1000)
00000000`00403274 9119a000 add         x0,x0,#0x668
00000000`00403278 b9801fa1 ldrsw       x1,[fp,#0x1C]
00000000`0040327c f8217802 str         x2,[x0,x1 lsl #3]
00000000`00403280 17ffffdc b           App10!proc+0x8 (00000000`004031f0)  Branch
```

Note: We see that we pass some location from the index array, which starts from the address `00000000`004d1000 + 0x668`:

```
0:000> ? 00000000`004d1000 + 0x668
Evaluate expression: 5052008 = 00000000`004d1668
```

10. Dump the first 100 elements of the identified array:

```
0:000> dp 00000000`004d1668 L100
00000000`004d1668  0000fffb`a93b4e00 0000fffb`a90c1af0
00000000`004d1678  0000fffb`aa939280 0000fffb`aae8e2d0
00000000`004d1688  0000fffb`a98aa350 0000fffb`a8fd48b0
00000000`004d1698  0000fffb`a912d220 0000fffb`aa420ad0
00000000`004d16a8  0000fffb`aae7f930 0000fffb`a87c2b20
00000000`004d16b8  0000fffb`a83d3e10 00000000`00000000
00000000`004d16c8  0000fffb`aa65da80 0000fffb`aa869280
00000000`004d16d8  0000fffb`aa5c5f80 0000fffb`a8459f40
00000000`004d16e8  0000fffb`a81e4e30 0000fffb`a810f820
00000000`004d16f8  0000fffb`a9e857a0 0000fffb`a9deacf0
00000000`004d1708  0000fffb`a8933250 0000fffb`a8306260
00000000`004d1718  0000fffb`aa9303e0 0000fffb`a85ef530
00000000`004d1728  0000fffb`aaefa020 0000fffb`a8b6f670
00000000`004d1738  0000fffb`aadd5320 0000fffb`aaab54f0
00000000`004d1748  0000fffb`a8ce7fc0 0000fffb`a8087370
00000000`004d1758  0000fffb`a823ab80 0000fffb`a8927f20
00000000`004d1768  0000fffb`a9bc66d0 0000fffb`aa81d670
00000000`004d1778  0000fffb`a8478270 0000fffb`a92edf60
00000000`004d1788  0000fffb`a8065330 0000fffb`a893faa0
00000000`004d1798  0000fffb`aa0fe190 0000fffb`a8fc1450
00000000`004d17a8  0000fffb`a84c8f20 0000fffb`a8b79280
00000000`004d17b8  0000fffb`aabe7040 0000fffb`a83d9110
00000000`004d17c8  0000fffb`a97dade0 0000fffb`a8fd0e80
00000000`004d17d8  0000fffb`aa20dc20 0000fffb`aa56f580
00000000`004d17e8  0000fffb`a9724a30 0000fffb`aa241aa0
00000000`004d17f8  0000fffb`a8049e50 0000fffb`a82f3e40
00000000`004d1808  0000fffb`a93b3cf0 0000fffb`aa9978d0
00000000`004d1818  0000fffb`a9223440 00000000`00000000
00000000`004d1828  0000fffb`a9cffd00 0000fffb`a9e323a0
00000000`004d1838  0000fffb`a9230070 0000fffb`a8cfc710
```

```
00000000`004d1848  0000fffb`a9e48810 0000fffb`a8474ea0
00000000`004d1858  0000fffb`a9557cf0 0000fffb`a86617f0
00000000`004d1868  0000fffb`a8737030 0000fffb`a92da510
00000000`004d1878  00000000`00000000 0000fffb`aa1977d0
00000000`004d1888  0000fffb`a9287350 0000fffb`a8071640
00000000`004d1898  0000fffb`a9d15240 0000fffb`a8ec3f00
00000000`004d18a8  0000fffb`a88273b0 0000fffb`bc00d7b0
00000000`004d18b8  00000000`00000000 0000fffb`a8d41340
00000000`004d18c8  0000fffb`aa48f100 0000fffb`a8f9b680
00000000`004d18d8  0000fffb`a8badda0 0000fffb`a91d85e0
00000000`004d18e8  0000fffb`a81f7730 0000fffb`a91ca3f0
00000000`004d18f8  0000fffb`a9688d40 0000fffb`aa760170
00000000`004d1908  0000fffb`a878cbd0 0000fffb`a8edbc30
00000000`004d1918  0000fffb`a877fe70 0000fffb`a861ca80
00000000`004d1928  0000fffb`a9003de0 0000fffb`a8cd3b50
00000000`004d1938  0000fffb`aa310670 0000fffb`a91095f0
00000000`004d1948  0000fffb`a840f270 0000fffb`a8b094f0
00000000`004d1958  0000fffb`a84334f0 0000fffb`aa272b90
00000000`004d1968  0000fffb`a8e280e0 0000fffb`a8c32df0
00000000`004d1978  0000fffb`aa81b390 0000fffb`a888cb80
00000000`004d1988  00000000`00000000 0000fffb`aa5009a0
00000000`004d1998  0000fffb`a8d5aaf0 0000fffb`a895fa50
00000000`004d19a8  0000fffb`a8726760 0000fffb`aa3bdf80
00000000`004d19b8  0000fffb`a8088d40 0000fffb`a8e63930
00000000`004d19c8  0000fffb`a8726130 0000fffb`aad1af40
00000000`004d19d8  0000fffb`aa300ed0 0000fffb`a992f670
00000000`004d19e8  0000fffb`aa194090 0000fffb`a9c5a300
00000000`004d19f8  0000fffb`aa5ea990 0000fffb`a92ab6d0
00000000`004d1a08  0000fffb`aa2f3700 0000fffb`a83e0b30
00000000`004d1a18  0000fffb`a8692620 0000fffb`a90d8260
00000000`004d1a28  0000fffb`aa62d8a0 0000fffb`a939e180
00000000`004d1a38  0000fffb`a8596f00 0000fffb`a9419c70
00000000`004d1a48  0000fffb`aa59f5a0 0000fffb`aa01ffd0
00000000`004d1a58  0000fffb`a85c1260 0000fffb`aa1a6250
00000000`004d1a68  0000fffb`a8b66780 0000fffb`aaa696f0
00000000`004d1a78  0000fffb`a86e3f60 0000fffb`a8bf7240
00000000`004d1a88  0000fffb`aa1e3ae0 0000fffb`a91c0f30
00000000`004d1a98  0000fffb`a9fc6b10 0000fffb`aa6e4700
00000000`004d1aa8  0000fffb`aa660770 0000fffb`a9a92e80
00000000`004d1ab8  0000fffb`aaac2fc0 0000fffb`a8ba0340
00000000`004d1ac8  0000fffb`a9cf6240 0000fffb`a8451290
00000000`004d1ad8  0000fffb`a88880c0 0000fffb`a92517e0
00000000`004d1ae8  0000fffb`aae65de0 0000fffb`a84ab520
00000000`004d1af8  0000fffb`a84d0b70 0000fffb`a8264490
00000000`004d1b08  0000fffb`aa4749b0 0000fffb`a9546660
00000000`004d1b18  0000fffb`a8a011c0 0000fffb`a938f9e0
00000000`004d1b28  0000fffb`aaebe240 0000fffb`a9999440
00000000`004d1b38  0000fffb`a9a72850 0000fffb`aab01a00
00000000`004d1b48  0000fffb`aa6795b0 0000fffb`aae9a180
00000000`004d1b58  0000fffb`aa39e590 0000fffb`aa08c140
00000000`004d1b68  0000fffb`a922c030 0000fffb`a9cd6540
00000000`004d1b78  0000fffb`a81cc7d0 0000fffb`a816e2b0
00000000`004d1b88  0000fffb`aa8a2070 00000000`00000000
00000000`004d1b98  0000fffb`aa2f7740 0000fffb`a98fa130
00000000`004d1ba8  0000fffb`a9811090 0000fffb`a85ca940
00000000`004d1bb8  0000fffb`a94f0ed0 0000fffb`a922a680
00000000`004d1bc8  0000fffb`a9a709d0 0000fffb`a8dc52e0
00000000`004d1bd8  0000fffb`a9e61de0 0000fffb`a856afe0
00000000`004d1be8  0000fffb`a823e260 0000fffb`a896e350
00000000`004d1bf8  0000fffb`a96f19a0 0000fffb`a902b650
```

```
00000000`004d1c08  0000fffb`a8095120 0000fffb`a89cc910
00000000`004d1c18  0000fffb`aac57210 0000fffb`aa78a580
00000000`004d1c28  0000fffb`a8bc4120 0000fffb`a97b25d0
00000000`004d1c38  0000fffb`aa243c10 00000000`00000000
00000000`004d1c48  0000fffb`aa7ba1a0 0000fffb`a8b94d90
00000000`004d1c58  0000fffb`a84977b0 0000fffb`a82c4910
00000000`004d1c68  0000fffb`aad189f0 0000fffb`a8f72680
00000000`004d1c78  0000fffb`a998bdc0 0000fffb`a8219e10
00000000`004d1c88  0000fffb`a90c16c0 0000fffb`a9535ab0
00000000`004d1c98  0000fffb`a8a68630 0000fffb`a87432d0
00000000`004d1ca8  0000fffb`aa18d820 0000fffb`a8581030
00000000`004d1cb8  0000fffb`a821d160 0000fffb`a8b50240
00000000`004d1cc8  0000fffb`a83dfb40 0000fffb`a82392d0
00000000`004d1cd8  0000fffb`a9594a10 0000fffb`a97fbf90
00000000`004d1ce8  0000fffb`a8602b00 0000fffb`aa13b630
00000000`004d1cf8  0000fffb`aa5fad40 0000fffb`aaf6fc80
00000000`004d1d08  0000fffb`aaf7b830 0000fffb`a8930850
00000000`004d1d18  0000fffb`a8ea2350 0000fffb`aaa0b820
00000000`004d1d28  0000fffb`a97d4630 0000fffb`a828efd0
00000000`004d1d38  0000fffb`aa193cf0 0000fffb`a8593ae0
00000000`004d1d48  0000fffb`a90c8f20 0000fffb`a9cbceb0
00000000`004d1d58  0000fffb`a99ee9c0 0000fffb`a9f4d790
00000000`004d1d68  0000fffb`a91bb0e0 0000fffb`a8501c80
00000000`004d1d78  0000fffb`aac967c0 00000000`00000000
00000000`004d1d88  0000fffb`aa961a70 0000fffb`a80ab010
00000000`004d1d98  0000fffb`a81af730 0000fffb`a9ebcd00
00000000`004d1da8  0000fffb`a8707c50 0000fffb`aa453720
00000000`004d1db8  0000fffb`a87b1330 0000fffb`aa56a710
00000000`004d1dc8  0000fffb`aa66b8c0 0000fffb`a8698260
00000000`004d1dd8  0000fffb`a8fa7d60 0000fffb`a951ba50
00000000`004d1de8  0000fffb`a828f910 0000fffb`a8330ab0
00000000`004d1df8  0000fffb`a848b200 0000fffb`a80f7e10
00000000`004d1e08  0000fffb`aacbc380 0000fffb`a8235540
00000000`004d1e18  0000fffb`a803e4b0 00000000`00000000
00000000`004d1e28  0000fffb`a9c33560 0000fffb`aa200ef0
00000000`004d1e38  0000fffb`a86aca10 0000fffb`a928aff0
00000000`004d1e48  0000fffb`a91a02b0 0000fffb`a8cf69d0
00000000`004d1e58  0000fffb`aaaad110 0000fffb`a8d72110
```

12. We close logging before exiting WinDbg Preview:

```
0:000> .logclose
Closing open log file 'C:\ALCDA2\A64\App10\App10.log
```

Exercise A11

- **Goal:** Learn how to identify synchronization wait chains, deadlocks, hidden and handled exceptions
- **Patterns:** Wait Chain (Mutex Objects); Deadlock (Mutex Objects, User Space); Disassembly Hole (WinDbg)
- \ALCDA-Dumps\Exercise-A11-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A11-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A11-A64-WinDbg.pdf

Exercise A11 (x64, GDB)

Goal: Learn how to identify synchronization wait chains, deadlocks, hidden and handled exceptions.

Patterns: Wait Chain (Mutex Objects); Deadlock (Mutex Objects).

1. Load *App11.core.594* dump file and *App11* executable from the x64/App11 directory:

```
~/ALCDA2/x64/App11$ gdb -c App11.core.594 -se App11
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App11...done.
[New LWP 594]
[New LWP 595]
[New LWP 596]
[New LWP 597]
[New LWP 598]
[New LWP 599]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App11'.
#0  0x0000000000452970 in nanosleep ()
[Current thread is 1 (Thread 0x13b9880 (LWP 594))]
```

2. List all thread stack traces and identify possible wait chain and deadlock:

```
(gdb) thread apply all bt

Thread 6 (Thread 0x7fa8cb7fe700 (LWP 599)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000402024 in bar_five() ()
#3  0x0000000000402030 in foo_five() ()
#4  0x0000000000402044 in thread_five(void*) ()
#5  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000045512f in clone ()

Thread 5 (Thread 0x7fa8cbfff700 (LWP 598)):
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
#1  0x0000000000415294 in __pthread_mutex_lock (mutex=0x4d34a0 <mutexA>) at
../nptl/pthread_mutex_lock.c:80
#2  0x0000000000401f27 in procB() () at pthread_create.c:688
#3  0x0000000000401fef in bar_four() () at pthread_create.c:688
#4  0x0000000000401ffb in foo_four() () at pthread_create.c:688
```

```
#5  0x000000000040200f in thread_four(void*) () at pthread_create.c:688
#6  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045512f in clone ()

Thread 4 (Thread 0x7fa8d086d700 (LWP 597)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000401fbf in bar_three() () at pthread_create.c:688
#3  0x0000000000401fcb in foo_three() () at pthread_create.c:688
#4  0x0000000000401fdf in thread_three(void*) () at pthread_create.c:688
#5  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000045512f in clone ()

Thread 3 (Thread 0x7fa8d106e700 (LWP 596)):
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
#1  0x0000000000415294 in __pthread_mutex_lock (mutex=0x4d34e0 <mutex>) at
../nptl/pthread_mutex_lock.c:80
#2  0x0000000000401eec in procA() () at pthread_create.c:688
#3  0x0000000000401f8a in bar_two() () at pthread_create.c:688
#4  0x0000000000401f96 in foo_two() () at pthread_create.c:688
#5  0x0000000000401faa in thread_two(void*) () at pthread_create.c:688
#6  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045512f in clone ()

Thread 2 (Thread 0x7fa8d186f700 (LWP 595)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000401f5a in bar_one() () at pthread_create.c:688
#3  0x0000000000401f66 in foo_one() () at pthread_create.c:688
#4  0x0000000000401f7a in thread_one(void*) () at pthread_create.c:688
#5  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000045512f in clone ()

Thread 1 (Thread 0x13b9880 (LWP 594)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000402121 in main () at pthread_create.c:688
```

3. Check **thread #5** and its waiting code:

```
(gdb) thread 5
[Switching to thread 5 (Thread 0x7fa8cbfff700 (LWP 598))]
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
103     ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S: No such file or directory.

(gdb) bt
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
#1  0x0000000000415294 in __pthread_mutex_lock (mutex=0x4d34a0 <mutexA>) at
../nptl/pthread_mutex_lock.c:80
#2  0x0000000000401f27 in procB() () at pthread_create.c:688
#3  0x0000000000401fef in bar_four() () at pthread_create.c:688
#4  0x0000000000401ffb in foo_four() () at pthread_create.c:688
#5  0x000000000040200f in thread_four(void*) () at pthread_create.c:688
#6  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045512f in clone ()
```

```
(gdb) disassemble procB
Dump of assembler code for function _Z5procBv:
   0x0000000000401f0b <+0>:     push   %rbp
   0x0000000000401f0c <+1>:     mov    %rsp,%rbp
   0x0000000000401f0f <+4>:     lea    0xd15ca(%rip),%rdi        # 0x4d34e0 <mutexB>
   0x0000000000401f16 <+11>:    callq  0x415240 <__pthread_mutex_lock>
   0x0000000000401f1b <+16>:    lea    0xd157e(%rip),%rdi        # 0x4d34a0 <mutexA>
   0x0000000000401f22 <+23>:    callq  0x415240 <__pthread_mutex_lock>
   0x0000000000401f27 <+28>:    mov    $0x1e,%edi
   0x0000000000401f2c <+33>:    callq  0x4528c0 <sleep>
   0x0000000000401f31 <+38>:    lea    0xd1568(%rip),%rdi        # 0x4d34a0 <mutexA>
   0x0000000000401f38 <+45>:    callq  0x4160b0 <__pthread_mutex_unlock>
   0x0000000000401f3d <+50>:    lea    0xd159c(%rip),%rdi        # 0x4d34e0 <mutexB>
   0x0000000000401f44 <+57>:    callq  0x4160b0 <__pthread_mutex_unlock>
   0x0000000000401f49 <+62>:    nop
   0x0000000000401f4a <+63>:    pop    %rbp
   0x0000000000401f4b <+64>:    retq
End of assembler dump.
```

Note: We see **thread #5** owns mutex **0x4d34e0** but is waiting for mutex **0x4d34a0**.

4. Check **thread #3** and its waiting code:

```
(gdb) thread 3
[Switching to thread 3 (Thread 0x7fa8d106e700 (LWP 596))]
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
103     ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S: No such file or directory.

(gdb) bt
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
#1  0x0000000000415294 in __pthread_mutex_lock (mutex=0x4d34e0 <mutex>) at
../nptl/pthread_mutex_lock.c:80
#2  0x0000000000401eec in procA() ()
#3  0x0000000000401f8a in bar_two() ()
#4  0x0000000000401f96 in foo_two() ()
#5  0x0000000000401faa in thread_two(void*) ()
#6  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045512f in clone ()

(gdb) disassemble procA
Dump of assembler code for function _Z5procAv:
   0x0000000000401eb5 <+0>:     push   %rbp
   0x0000000000401eb6 <+1>:     mov    %rsp,%rbp
   0x0000000000401eb9 <+4>:     lea    0xd15e0(%rip),%rdi        # 0x4d34a0 <mutexA>
   0x0000000000401ec0 <+11>:    callq  0x415240 <__pthread_mutex_lock>
   0x0000000000401ec5 <+16>:    callq  0x401e8d <_Z5procCv>
   0x0000000000401eca <+21>:    lea    0xd15cf(%rip),%rdi        # 0x4d34a0 <mutexA>
   0x0000000000401ed1 <+28>:    callq  0x4160b0 <__pthread_mutex_unlock>
   0x0000000000401ed6 <+33>:    mov    $0x14,%edi
   0x0000000000401edb <+38>:    callq  0x4528c0 <sleep>
   0x0000000000401ee0 <+43>:    lea    0xd15f9(%rip),%rdi        # 0x4d34e0 <mutexB>
   0x0000000000401ee7 <+50>:    callq  0x415240 <__pthread_mutex_lock>
   0x0000000000401eec <+55>:    lea    0xd15ed(%rip),%rdi        # 0x4d34e0 <mutexB>
   0x0000000000401ef3 <+62>:    callq  0x4160b0 <__pthread_mutex_unlock>
   0x0000000000401ef8 <+67>:    jmp    0x401f09 <_Z5procAv+84>
   0x0000000000401efa <+69>:    mov    %rax,%rdi
   0x0000000000401efd <+72>:    callq  0x402ec0 <__cxa_begin_catch>
   0x0000000000401f02 <+77>:    callq  0x402f30 <__cxa_end_catch>
   0x0000000000401f07 <+82>:    jmp    0x401ed6 <_Z5procAv+33>
   0x0000000000401f09 <+84>:    pop    %rbp
```

```
   0x0000000000401f0a <+85>:    retq
End of assembler dump.
```

Note: We see that **thread #3** is waiting for **0x4d34e0** mutex but shouldn't own **0x4d34a0** mutex because it should have unlocked it unless something happened in **procC**. We also notice catch exception processing which transfers execution for the block of code waiting for mutex **0x4d34e0.**

Note: We see C++ function names are mangled, so we can demangle them if necessary (however, it may affect some variable names like **mutexB**):

```
(gdb) set print asm-demangle on

(gdb) disassemble procA
Dump of assembler code for function _Z5procAv:
   0x0000000000401eb5 <+0>:     push   %rbp
   0x0000000000401eb6 <+1>:     mov    %rsp,%rbp
   0x0000000000401eb9 <+4>:     lea    0xd15e0(%rip),%rdi        # 0x4d34a0 <mutexA>
   0x0000000000401ec0 <+11>:    callq  0x415240 <__pthread_mutex_lock>
   0x0000000000401ec5 <+16>:    callq  0x401e8d <procC()>
   0x0000000000401eca <+21>:    lea    0xd15cf(%rip),%rdi        # 0x4d34a0 <mutexA>
   0x0000000000401ed1 <+28>:    callq  0x4160b0 <__pthread_mutex_unlock>
   0x0000000000401ed6 <+33>:    mov    $0x14,%edi
   0x0000000000401edb <+38>:    callq  0x4528c0 <sleep>
   0x0000000000401ee0 <+43>:    lea    0xd15f9(%rip),%rdi        # 0x4d34e0 <mutex>
   0x0000000000401ee7 <+50>:    callq  0x415240 <__pthread_mutex_lock>
   0x0000000000401eec <+55>:    lea    0xd15ed(%rip),%rdi        # 0x4d34e0 <mutex>
   0x0000000000401ef3 <+62>:    callq  0x4160b0 <__pthread_mutex_unlock>
   0x0000000000401ef8 <+67>:    jmp    0x401f09 <procA()+84>
   0x0000000000401efa <+69>:    mov    %rax,%rdi
   0x0000000000401efd <+72>:    callq  0x402ec0 <__cxa_begin_catch>
   0x0000000000401f02 <+77>:    callq  0x402f30 <__cxa_end_catch>
   0x0000000000401f07 <+82>:    jmp    0x401ed6 <procA()+33>
   0x0000000000401f09 <+84>:    pop    %rbp
   0x0000000000401f0a <+85>:    retq
End of assembler dump.
```

9. Disassemble *procC* code:

```
(gdb) disassemble procC
Dump of assembler code for function _Z5procCv:
   0x0000000000401e8d <+0>:     push   %rbp
   0x0000000000401e8e <+1>:     mov    %rsp,%rbp
   0x0000000000401e91 <+4>:     mov    $0x4,%edi
   0x0000000000401e96 <+9>:     callq  0x403420 <__cxa_allocate_exception>
   0x0000000000401e9b <+14>:    movl   $0x0,(%rax)
   0x0000000000401ea1 <+20>:    mov    $0x0,%edx
   0x0000000000401ea6 <+25>:    lea    0xcde33(%rip),%rsi        # 0x4cfce0 <typeinfo for int>
   0x0000000000401ead <+32>:    mov    %rax,%rdi
   0x0000000000401eb0 <+35>:    callq  0x402200 <__cxa_throw>
End of assembler dump.
```

Note: We see that code throws an exception, so perhaps it was caught in the caller *procA,* and mutex unlock wasn't called, thus causing a deadlock.

10. Check if there was any exception processing:

```
(gdb) x/300a $rsp-2400
0x7fa8d106d368: 0x0     0x0
0x7fa8d106d378: 0x0     0x0
0x7fa8d106d388: 0x0     0x0
0x7fa8d106d398: 0x0     0x0
0x7fa8d106d3a8: 0x0     0x0
0x7fa8d106d3b8: 0x0     0x0
0x7fa8d106d3c8: 0x0     0x0
0x7fa8d106d3d8: 0x0     0x0
0x7fa8d106d3e8: 0x0     0x0
0x7fa8d106d3f8: 0x0     0x0
0x7fa8d106d408: 0x0     0x0
0x7fa8d106d418: 0x0     0x0
0x7fa8d106d428: 0x0     0x0
0x7fa8d106d438: 0x0     0x0
0x7fa8d106d448: 0x0     0x0
0x7fa8d106d458: 0x0     0x0
0x7fa8d106d468: 0x0     0x0
0x7fa8d106d478: 0x0     0x0
0x7fa8d106d488: 0x0     0x0
0x7fa8d106d498: 0x0     0x0
0x7fa8d106d4a8: 0x0     0x0
0x7fa8d106d4b8: 0x0     0x0
0x7fa8d106d4c8: 0x0     0x0
0x7fa8d106d4d8: 0x0     0x0
0x7fa8d106d4e8: 0x0     0x0
0x7fa8d106d4f8: 0x0     0x0
0x7fa8d106d508: 0x0     0x0
0x7fa8d106d518: 0x0     0x0
0x7fa8d106d528: 0x0     0x0
0x7fa8d106d538: 0x0     0x0
0x7fa8d106d548: 0x0     0x0
0x7fa8d106d558: 0x410d1d <fde_single_encoding_compare+93>       0x401850
<_GLOBAL__sub_I_eh_alloc.cc>
0x7fa8d106d568: 0x401840 <fini> 0x7fa8cc003370
0x7fa8d106d578: 0x1     0x7fa8cc003370
0x7fa8d106d588: 0x2     0x410cc0 <fde_single_encoding_compare>
0x7fa8d106d598: 0x410970 <frame_downheap+112>   0x7fa8cc001c98
0x7fa8d106d5a8: 0x4d3460 <object>       0x7fa8cc003380
0x7fa8d106d5b8: 0x9e    0xce0000027b
0x7fa8d106d5c8: 0x0     0x7fa8cc003370
0x7fa8d106d5d8: 0x4d3460 <object>       0x1dcb0
0x7fa8d106d5e8: 0x4cb   0x7fa8cc003360
0x7fa8d106d5f8: 0x410a51 <frame_heapsort+145>   0x27a8
0x7fa8d106d608: 0xc9abaa553021fb00      0x401834 <read_encoded_value_with_base.cold>
0x7fa8d106d618: 0x410cc0 <fde_single_encoding_compare>  0x4d3460 <object>
0x7fa8d106d628: 0x27    0x7fa8cc000bb0
0x7fa8d106d638: 0x41007d <_Unwind_RaiseException+61>    0x4c9768
0x7fa8d106d648: 0x411b56 <search_object+854>    0x0
0x7fa8d106d658: 0x7fa8d106d698  0x7fa8d106d6a0
0x7fa8d106d668: 0x7fa80000001b  0x0
0x7fa8d106d678: 0x7fa80000000b  0x7fa8d106dcf8
0x7fa8d106d688: 0x7fa8d106dd00  0x0
0x7fa8d106d698: 0x401eb5 <procA()>      0x56
0x7fa8d106d6a8: 0x0     0x7fa8d106dd48
0x7fa8d106d6b8: 0x4d3460 <object>       0x7fa8d106d7d0
0x7fa8d106d6c8: 0x7fa8d106db28  0x4160c0 <__pthread_key_create>
0x7fa8d106d6d8: 0x402838 <__gxx_personality_v0+184>     0x401ec9 <procA()+20>
```

```
0x7fa8d106d6e8: 0x7fa8cc000b80  0x601401eb5
--Type <RET> for more, q to quit, c to continue without paging--
0x7fa8d106d6f8: 0x0     0x4caa38
0x7fa8d106d708: 0x7fa8d106d701  0x7fa8d106dd60
0x7fa8d106d718: 0x401eb5 <procA()>      0x0
0x7fa8d106d728: 0x4bcdd1        0x4bcdd1
0x7fa8d106d738: 0xd106da80      0x0
0x7fa8d106d748: 0x1b    0x7fa8d106d7d0
0x7fa8d106d758: 0x40e940 <uw_frame_state_for+800>       0x0
0x7fa8d106d768: 0x9b00000000    0x4bcdb5
0x7fa8d106d778: 0x7fa8d106d788  0x7fa8d106d7d0
0x7fa8d106d788: 0x4caa38        0x7fa8d106d7d0
0x7fa8d106d798: 0x7fa8d106da80  0x4
0x7fa8d106d7a8: 0x7fa8cc000b80  0x7fa8d106d7d0
0x7fa8d106d7b8: 0x3     0x7fa8d106db70
0x7fa8d106d7c8: 0x40fa6b <_Unwind_RaiseException_Phase2+75>     0x0
0x7fa8d106d7d8: 0x0     0x0
0x7fa8d106d7e8: 0x0     0x0
0x7fa8d106d7f8: 0x0     0x0
0x7fa8d106d808: 0x0     0x0
0x7fa8d106d818: 0x0     0x0
0x7fa8d106d828: 0x0     0xfffffffffffffff0
0x7fa8d106d838: 0x1     0x0
0x7fa8d106d848: 0x0     0x0
0x7fa8d106d858: 0x0     0x0
0x7fa8d106d868: 0x0     0x0
0x7fa8d106d878: 0x0     0x0
0x7fa8d106d888: 0x0     0x0
0x7fa8d106d898: 0x0     0x0
0x7fa8d106d8a8: 0x0     0x0
0x7fa8d106d8b8: 0x0     0x0
0x7fa8d106d8c8: 0x0     0xfffffffffffffff8
0x7fa8d106d8d8: 0x1     0x0
0x7fa8d106d8e8: 0x0     0x0
0x7fa8d106d8f8: 0x10    0x6
0x7fa8d106d908: 0x0     0x1
0x7fa8d106d918: 0x401f0a <procA()+85>   0x402780 <__gxx_personality_v0>
0x7fa8d106d928: 0xfffffffffffffff8      0x1
0x7fa8d106d938: 0x10    0x11b1b
0x7fa8d106d948: 0x0     0x7fa8cc000b80
0x7fa8d106d958: 0x7fa8d106da80  0x7fa8d106dd30
0x7fa8d106d968: 0x7fa8d106db70  0x7fa8cc000b80
0x7fa8d106d978: 0x7fa8d106dd70  0x0
0x7fa8d106d988: 0x41031a <_Unwind_RaiseException+730>   0x7fa8d106dcf8
0x7fa8d106d998: 0x7fa8d106dd00  0x0
0x7fa8d106d9a8: 0x7fa8d106dd08  0x0
0x7fa8d106d9b8: 0x0     0x7fa8d106dd30
0x7fa8d106d9c8: 0x0     0x0
0x7fa8d106d9d8: 0x0     0x0
0x7fa8d106d9e8: 0x0     0x7fa8d106dd10
0x7fa8d106d9f8: 0x7fa8d106dd18  0x7fa8d106dd20
0x7fa8d106da08: 0x7fa8d106dd28  0x7fa8d106dd38
0x7fa8d106da18: 0x0     0x7fa8d106dd40
0x7fa8d106da28: 0x402237 <__cxa_throw+55>       0x0
0x7fa8d106da38: 0x0     0x0
0x7fa8d106da48: 0x410040 <_Unwind_RaiseException>       0x4000000000000000
0x7fa8d106da58: 0x0     0x0
0x7fa8d106da68: 0x0     0x0
0x7fa8d106da78: 0x0     0x7fa8d106dcf8
--Type <RET> for more, q to quit, c to continue without paging--
```

```
0x7fa8d106da88: 0x7fa8d106dd00  0x0
0x7fa8d106da98: 0x7fa8d106dd40  0x0
0x7fa8d106daa8: 0x0     0x7fa8d106dd60
0x7fa8d106dab8: 0x7fa8d106d978  0x0
0x7fa8d106dac8: 0x0     0x0
0x7fa8d106dad8: 0x0     0x7fa8d106dd50
0x7fa8d106dae8: 0x7fa8d106dd18  0x7fa8d106dd20
0x7fa8d106daf8: 0x7fa8d106dd28  0x7fa8d106dd68
0x7fa8d106db08: 0x0     0x7fa8d106dd70
0x7fa8d106db18: 0x401efa <procA()+69>   0x4caa38
0x7fa8d106db28: 0x0     0x0
0x7fa8d106db38: 0x401eb5 <procA()>      0x4000000000000000
0x7fa8d106db48: 0x0     0x0
0x7fa8d106db58: 0x0     0x0
0x7fa8d106db68: 0x0     0x3
0x7fa8d106db78: 0x0     0x0
0x7fa8d106db88: 0x0     0x0
0x7fa8d106db98: 0x0     0x0
0x7fa8d106dba8: 0x0     0x0
0x7fa8d106dbb8: 0x0     0x0
0x7fa8d106dbc8: 0x0     0xfffffffffffffff0
0x7fa8d106dbd8: 0x1     0x0
0x7fa8d106dbe8: 0x0     0x0
0x7fa8d106dbf8: 0x0     0x0
0x7fa8d106dc08: 0x0     0x0
0x7fa8d106dc18: 0x0     0x0
0x7fa8d106dc28: 0x0     0x0
0x7fa8d106dc38: 0x0     0x0
0x7fa8d106dc48: 0x0     0x0
0x7fa8d106dc58: 0x0     0x0
0x7fa8d106dc68: 0x0     0xfffffffffffffff8
0x7fa8d106dc78: 0x1     0x0
0x7fa8d106dc88: 0x0     0x0
0x7fa8d106dc98: 0x10    0x6
0x7fa8d106dca8: 0x0     0x1
0x7fa8d106dcb8: 0x401f0a <procA()+85>   0x402780 <__gxx_personality_v0>
```

Note: We see a reference **0x401efa <procA()+69>** from exception processing block in *procA* and also **0x402237 <__cxa_throw+55>**. We check whether the symbolic information we found is not coincidental:

```
(gdb) disassemble __cxa_throw
Dump of assembler code for function __cxa_throw:
   0x0000000000402200 <+0>:     push   %r12
   0x0000000000402202 <+2>:     mov    %rdx,%r12
   0x0000000000402205 <+5>:     push   %rbp
   0x0000000000402206 <+6>:     mov    %rsi,%rbp
   0x0000000000402209 <+9>:     push   %rbx
   0x000000000040220a <+10>:    mov    %rdi,%rbx
   0x000000000040220d <+13>:    nop
   0x000000000040220e <+14>:    callq  0x402d60 <__cxa_get_globals>
   0x0000000000402213 <+19>:    mov    %r12,%rdx
   0x0000000000402216 <+22>:    mov    %rbp,%rsi
   0x0000000000402219 <+25>:    mov    %rbx,%rdi
   0x000000000040221c <+28>:    addl   $0x1,0x8(%rax)
   0x0000000000402220 <+32>:    callq  0x4021b0 <__cxa_init_primary_exception>
   0x0000000000402225 <+37>:    movl   $0x1,(%rax)
   0x000000000040222b <+43>:    lea    0x60(%rax),%rbx
   0x000000000040222f <+47>:    mov    %rbx,%rdi
   0x0000000000402232 <+50>:    callq  0x410040 <_Unwind_RaiseException>
   0x0000000000402237 <+55>:    mov    %rbx,%rdi
```

```
   0x000000000040223a <+58>:    callq  0x402ec0 <__cxa_begin_catch>
   0x000000000040223f <+63>:    callq  0x4022f0 <std::terminate()>
End of assembler dump.
```

11. Since mutexes have owners, we can check their ownership instead of disassembly:

```
(gdb) print *(pthread_mutex_t *)&mutexA
$2 = {__data = {__lock = 2, __count = 0, __owner = 596, __nusers = 1, __kind = 0, __spins = 0,
__elision = 0, __list = {__prev = 0x0, __next = 0x0}},
  __size = "\002\000\000\000\000\000\000\000T\002\000\000\001", '\000' <repeats 26 times>,
__align = 2}

(gdb) print *(pthread_mutex_t *)&mutexB
$3 = {__data = {__lock = 2, __count = 0, __owner = 598, __nusers = 1, __kind = 0, __spins = 0,
__elision = 0, __list = {__prev = 0x0, __next = 0x0}},
  __size = "\002\000\000\000\000\000\000\000V\002\000\000\001", '\000' <repeats 26 times>,
__align = 2}
```

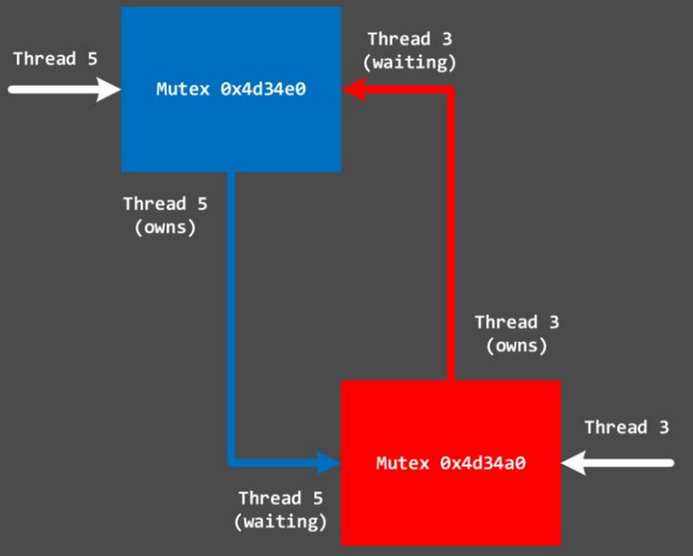
Note: We see their respective thread owners (LWP numbers).

```
(gdb) thread 5
[Switching to thread 5 (Thread 0x7fa8cbfff700 (LWP 598))]
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
103     in ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S

(gdb) bt
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
#1  0x0000000000415294 in __pthread_mutex_lock (mutex=0x4d34a0 <mutexA>) at
../nptl/pthread_mutex_lock.c:80
#2  0x0000000000401f27 in procB() () at pthread_create.c:688
#3  0x0000000000401fef in bar_four() () at pthread_create.c:688
#4  0x0000000000401ffb in foo_four() () at pthread_create.c:688
#5  0x000000000040200f in thread_four(void*) () at pthread_create.c:688
#6  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045512f in clone ()

(gdb) thread 3
[Switching to thread 3 (Thread 0x7fa8d106e700 (LWP 596))]
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
103     in ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S

(gdb) bt
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
#1  0x0000000000415294 in __pthread_mutex_lock (mutex=0x4d34e0 <mutex>) at
../nptl/pthread_mutex_lock.c:80
#2  0x0000000000401eec in procA() () at pthread_create.c:688
#3  0x0000000000401f8a in bar_two() () at pthread_create.c:688
#4  0x0000000000401f96 in foo_two() () at pthread_create.c:688
#5  0x0000000000401faa in thread_two(void*) () at pthread_create.c:688
#6  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045512f in clone ()
```

Note: We see the **mutex** name in the backtrace instead of **mutexB,** which can be discovered by disassembly after turning demangling off:

```
(gdb) disass 0x0000000000401eec
Dump of assembler code for function _Z5procAv:
   0x0000000000401eb5 <+0>:     push   %rbp
   0x0000000000401eb6 <+1>:     mov    %rsp,%rbp
   0x0000000000401eb9 <+4>:     lea    0xd15e0(%rip),%rdi        # 0x4d34a0 <mutexA>
   0x0000000000401ec0 <+11>:    callq  0x415240 <__pthread_mutex_lock>
   0x0000000000401ec5 <+16>:    callq  0x401e8d <_Z5procCv>
   0x0000000000401eca <+21>:    lea    0xd15cf(%rip),%rdi        # 0x4d34a0 <mutexA>
   0x0000000000401ed1 <+28>:    callq  0x4160b0 <__pthread_mutex_unlock>
   0x0000000000401ed6 <+33>:    mov    $0x14,%edi
   0x0000000000401edb <+38>:    callq  0x4528c0 <sleep>
   0x0000000000401ee0 <+43>:    lea    0xd15f9(%rip),%rdi        # 0x4d34e0 <mutexB>
   0x0000000000401ee7 <+50>:    callq  0x415240 <__pthread_mutex_lock>
   0x0000000000401eec <+55>:    lea    0xd15ed(%rip),%rdi        # 0x4d34e0 <mutexB>
   0x0000000000401ef3 <+62>:    callq  0x4160b0 <__pthread_mutex_unlock>
   0x0000000000401ef8 <+67>:    jmp    0x401f09 <_Z5procAv+84>
   0x0000000000401efa <+69>:    mov    %rax,%rdi
   0x0000000000401efd <+72>:    callq  0x402ec0 <__cxa_begin_catch>
   0x0000000000401f02 <+77>:    callq  0x402f30 <__cxa_end_catch>
   0x0000000000401f07 <+82>:    jmp    0x401ed6 <_Z5procAv+33>
   0x0000000000401f09 <+84>:    pop    %rbp
   0x0000000000401f0a <+85>:    retq
End of assembler dump.
```

Deadlock (x64, GDB)



Exercise A11 (A64, GDB)

Goal: Learn how to identify synchronization wait chains, deadlocks, hidden and handled exceptions.

Patterns: Wait Chain (Mutex Objects); Deadlock (Mutex Objects).

1. Load *App11.core.11410* dump file and *App11* executable from the A64/App11 directory:

```
~/ALCDA2/A64/App11$ gdb -c App11.core.11410 -se App11
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App11...
(No debugging symbols found in App11)

warning: Can't open file /home/opc/ALCDA2/App11/App11 during file-backed mapping note
processing
[New LWP 11411]
[New LWP 11412]
[New LWP 11421]
[New LWP 11422]
[New LWP 11423]
[New LWP 11410]
Core was generated by `./App11'.
#0  0x000000000041ae24 in nanosleep ()
[Current thread is 1 (LWP 11411)]
```

2. Set logging to a file in case of lengthy output from some commands:

```
(gdb) set logging file App11.log

(gdb) set logging enabled on
Copying output to App11.log.
Copying debug output to App11.log.

(gdb) set style enabled off
```

3. List all thread stack traces and identify possible wait chain and deadlock:

```
(gdb) thread apply all bt

Thread 6 (LWP 11410):
#0  0x000000000041ae24 in nanosleep ()
#1  0x00000000004338b4 in sleep ()
#2  0x00000000004034e0 in main ()
```

```
Thread 5 (LWP 11423):
#0  0x000000000041ae24 in nanosleep ()
#1  0x00000000004338b4 in sleep ()
#2  0x00000000004033f0 in bar_five() ()
#3  0x0000000000403404 in foo_five() ()
#4  0x000000000040341c in thread_five(void*) ()
#5  0x00000000004130a4 in start_thread ()
#6  0x0000000000438760 in thread_start ()

Thread 4 (LWP 11422):
#0  0x000000000041a110 in __lll_lock_wait ()
#1  0x0000000000414ea4 in pthread_mutex_lock ()
#2  0x00000000004032a0 in procB() ()
#3  0x00000000004033a8 in bar_four() ()
#4  0x00000000004033bc in foo_four() ()
#5  0x00000000004033d4 in thread_four(void*) ()
#6  0x00000000004130a4 in start_thread ()
#7  0x0000000000438760 in thread_start ()

Thread 3 (LWP 11421):
#0  0x000000000041ae24 in nanosleep ()
#1  0x00000000004338b4 in sleep ()
#2  0x0000000000403364 in bar_three() ()
#3  0x0000000000403378 in foo_three() ()
#4  0x0000000000403390 in thread_three(void*) ()
#5  0x00000000004130a4 in start_thread ()
#6  0x0000000000438760 in thread_start ()

Thread 2 (LWP 11412):
#0  0x000000000041a110 in __lll_lock_wait ()
#1  0x0000000000414ea4 in pthread_mutex_lock ()
#2  0x000000000040325c in procA() ()
#3  0x000000000040331c in bar_two() ()
#4  0x0000000000403330 in foo_two() ()
#5  0x0000000000403348 in thread_two(void*) ()
#6  0x00000000004130a4 in start_thread ()
#7  0x0000000000438760 in thread_start ()

Thread 1 (LWP 11411):
#0  0x000000000041ae24 in nanosleep ()
#1  0x00000000004338b4 in sleep ()
#2  0x00000000004032d8 in bar_one() ()
#3  0x00000000004032ec in foo_one() ()
--Type <RET> for more, q to quit, c to continue without paging--
#4  0x0000000000403304 in thread_one(void*) ()
#5  0x00000000004130a4 in start_thread ()
#6  0x0000000000438760 in thread_start ()
```

4. Check **thread #4** and its waiting code:

```
(gdb) thread 4
[Switching to thread 4 (LWP 11422)]
#0  0x000000000041a110 in __lll_lock_wait ()
```

```
(gdb) bt
#0  0x000000000041a110 in __lll_lock_wait ()
#1  0x0000000000414ea4 in pthread_mutex_lock ()
#2  0x00000000004032a0 in procB() ()
#3  0x00000000004033a8 in bar_four() ()
#4  0x00000000004033bc in foo_four() ()
#5  0x00000000004033d4 in thread_four(void*) ()
#6  0x00000000004130a4 in start_thread ()
#7  0x0000000000438760 in thread_start ()
```

```
(gdb) disassemble procB
Dump of assembler code for function _Z5procBv:
   0x0000000000403280 <+0>:     stp     x29, x30, [sp, #-16]!
   0x0000000000403284 <+4>:     mov     x29, sp
   0x0000000000403288 <+8>:     adrp    x0, 0x4d1000 <main_arena+2128>
   0x000000000040328c <+12>:    add     x0, x0, #0x608
   0x0000000000403290 <+16>:    bl      0x414dbc <pthread_mutex_lock>
   0x0000000000403294 <+20>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403298 <+24>:    add     x0, x0, #0x5d8
   0x000000000040329c <+28>:    bl      0x414dbc <pthread_mutex_lock>
   0x00000000004032a0 <+32>:    mov     w0, #0x1e                       // #30
   0x00000000004032a4 <+36>:    bl      0x4337a4 <sleep>
   0x00000000004032a8 <+40>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x00000000004032ac <+44>:    add     x0, x0, #0x5d8
   0x00000000004032b0 <+48>:    bl      0x416054 <pthread_mutex_unlock>
   0x00000000004032b4 <+52>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x00000000004032b8 <+56>:    add     x0, x0, #0x608
   0x00000000004032bc <+60>:    bl      0x416054 <pthread_mutex_unlock>
   0x00000000004032c0 <+64>:    ldp     x29, x30, [sp], #16
   0x00000000004032c4 <+68>:    ret
End of assembler dump.
```

Note: We see **thread #4** owns the mutex **0x004d1000 + 0x608** but is waiting for the mutex **0x004d1000 + 0x5D8**.

5. Check **thread #2** and its waiting code:

```
(gdb) thread 2
[Switching to thread 2 (LWP 11412)]
#0  0x000000000041a110 in __lll_lock_wait ()
```

```
(gdb) bt
#0  0x000000000041a110 in __lll_lock_wait ()
#1  0x0000000000414ea4 in pthread_mutex_lock ()
#2  0x000000000040325c in procA() ()
#3  0x000000000040331c in bar_two() ()
#4  0x0000000000403330 in foo_two() ()
#5  0x0000000000403348 in thread_two(void*) ()
#6  0x00000000004130a4 in start_thread ()
#7  0x0000000000438760 in thread_start ()
```

```
(gdb) disassemble procA
Dump of assembler code for function _Z5procAv:
   0x0000000000403224 <+0>:     stp     x29, x30, [sp, #-16]!
   0x0000000000403228 <+4>:     mov     x29, sp
   0x000000000040322c <+8>:     adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403230 <+12>:    add     x0, x0, #0x5d8
   0x0000000000403234 <+16>:    bl      0x414dbc <pthread_mutex_lock>
   0x0000000000403238 <+20>:    bl      0x403200 <_Z5procCv>
   0x000000000040323c <+24>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403240 <+28>:    add     x0, x0, #0x5d8
```

```
   0x0000000000403244 <+32>:    bl      0x416054 <pthread_mutex_unlock>
   0x0000000000403248 <+36>:    mov     w0, #0x14                       // #20
   0x000000000040324c <+40>:    bl      0x4337a4 <sleep>
   0x0000000000403250 <+44>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403254 <+48>:    add     x0, x0, #0x608
   0x0000000000403258 <+52>:    bl      0x414dbc <pthread_mutex_lock>
   0x000000000040325c <+56>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403260 <+60>:    add     x0, x0, #0x608
   0x0000000000403264 <+64>:    bl      0x416054 <pthread_mutex_unlock>
   0x0000000000403268 <+68>:    b       0x403278 <_Z5procAv+84>
   0x000000000040326c <+72>:    bl      0x4039b4 <__cxa_begin_catch>
   0x0000000000403270 <+76>:    bl      0x403a58 <__cxa_end_catch>
   0x0000000000403274 <+80>:    b       0x403248 <_Z5procAv+36>
   0x0000000000403278 <+84>:    ldp     x29, x30, [sp], #16
   0x000000000040327c <+88>:    ret
End of assembler dump.
```

Note: We see that **thread #2** is waiting for the **0x004d1000 + 0x608** mutex but shouldn't own the **0x004d1000 + 0x5D8** mutex because it should have unlocked it unless something happened in **procC**. We also notice *catch* exception processing which transfers execution for the block of code waiting for the mutex **0x004d1000 + 0x608.**

Note: We see C++ function names are mangled, so we can demangle them if necessary:

```
(gdb) set print asm-demangle on
```

```
(gdb) disassemble procA
Dump of assembler code for function procA():
   0x0000000000403224 <+0>:     stp     x29, x30, [sp, #-16]!
   0x0000000000403228 <+4>:     mov     x29, sp
   0x000000000040322c <+8>:     adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403230 <+12>:    add     x0, x0, #0x5d8
   0x0000000000403234 <+16>:    bl      0x414dbc <pthread_mutex_lock>
   0x0000000000403238 <+20>:    bl      0x403200 <procC()>
   0x000000000040323c <+24>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403240 <+28>:    add     x0, x0, #0x5d8
   0x0000000000403244 <+32>:    bl      0x416054 <pthread_mutex_unlock>
   0x0000000000403248 <+36>:    mov     w0, #0x14                       // #20
   0x000000000040324c <+40>:    bl      0x4337a4 <sleep>
   0x0000000000403250 <+44>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403254 <+48>:    add     x0, x0, #0x608
   0x0000000000403258 <+52>:    bl      0x414dbc <pthread_mutex_lock>
   0x000000000040325c <+56>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403260 <+60>:    add     x0, x0, #0x608
   0x0000000000403264 <+64>:    bl      0x416054 <pthread_mutex_unlock>
   0x0000000000403268 <+68>:    b       0x403278 <procA()+84>
   0x000000000040326c <+72>:    bl      0x4039b4 <__cxa_begin_catch>
   0x0000000000403270 <+76>:    bl      0x403a58 <__cxa_end_catch>
   0x0000000000403274 <+80>:    b       0x403248 <procA()+36>
   0x0000000000403278 <+84>:    ldp     x29, x30, [sp], #16
   0x000000000040327c <+88>:    ret
End of assembler dump.
```

6. Disassemble *procC* code:

```
(gdb) disassemble procC
Dump of assembler code for function procC():
   0x0000000000403200 <+0>:     stp     x29, x30, [sp, #-16]!
   0x0000000000403204 <+4>:     mov     x29, sp
   0x0000000000403208 <+8>:     mov     x0, #0x4                        // #4
   0x000000000040320c <+12>:    bl      0x403624 <__cxa_allocate_exception>
   0x0000000000403210 <+16>:    str     wzr, [x0]
   0x0000000000403214 <+20>:    adrp    x1, 0x4cf000 <vtable for std::exception+24>
   0x0000000000403218 <+24>:    add     x1, x1, #0x580
   0x000000000040321c <+28>:    mov     x2, #0x0                        // #0
   0x0000000000403220 <+32>:    bl      0x4047c0 <__cxa_throw>
End of assembler dump.
```

Note: We see that code throws an exception, so perhaps it was caught in the caller *procA,* and mutex *unlock* wasn't called, thus causing a deadlock.

7. Check if there was any exception processing:

```
(gdb) x/512a $sp-0x1000
0xfffe0c03d7d0: 0x0     0x0
0xfffe0c03d7e0: 0x0     0x0
0xfffe0c03d7f0: 0x0     0x0
0xfffe0c03d800: 0x0     0x0
0xfffe0c03d810: 0x0     0x0
0xfffe0c03d820: 0x0     0x0
0xfffe0c03d830: 0x0     0x0
0xfffe0c03d840: 0x0     0x0
0xfffe0c03d850: 0x0     0x0
0xfffe0c03d860: 0x0     0x0
0xfffe0c03d870: 0x0     0x0
0xfffe0c03d880: 0x0     0x0
0xfffe0c03d890: 0x0     0x10
0xfffe0c03d8a0: 0xfffe0c03d8f0  0x40f8b0 <_Unwind_RaiseException+344>
0xfffe0c03d8b0: 0xfffe0c03dd80  0xfffe0c03e140
0xfffe0c03d8c0: 0xfffe04000b80  0xfffe0c03d9c0
0xfffe0c03d8d0: 0x0     0x1e
0xfffe0c03d8e0: 0x11b1b 0xfffe0c03e800
0xfffe0c03d8f0: 0xfffe0c03e800  0x40326c <procA()+72>
0xfffe0c03d900: 0xfffe04000b80  0x1
0xfffe0c03d910: 0x0     0x1
0xfffe0c03d920: 0xfffe0c03f070  0x0
0xfffe0c03d930: 0x4d0000        0x403338 <thread_two(void*)>
0xfffe0c03d940: 0x0     0xfffe0c03f760
0xfffe0c03d950: 0x2ba06f0       0x4e9558 <__default_pthread_attr>
0xfffe0c03d960: 0x10000 0x810000
0xfffe0c03d970: 0x0     0x0
0xfffe0c03d980: 0x0     0x0
0xfffe0c03d990: 0x0     0x0
0xfffe0c03d9a0: 0x0     0x0
0xfffe0c03d9b0: 0x0     0x40
0xfffe0c03d9c0: 0xfffe0c03d900  0xfffe0c03d908
0xfffe0c03d9d0: 0xfffe0c03d910  0xfffe0c03d918
0xfffe0c03d9e0: 0x0     0x0
0xfffe0c03d9f0: 0x0     0x0
0xfffe0c03da00: 0x0     0x0
0xfffe0c03da10: 0x0     0x0
0xfffe0c03da20: 0x0     0x0
```

```
0xfffe0c03da30: 0x0     0x0
0xfffe0c03da40: 0x0     0x0
0xfffe0c03da50: 0x0     0xfffe0c03d920
0xfffe0c03da60: 0xfffe0c03d928  0xfffe0c03d930
0xfffe0c03da70: 0xfffe0c03d938  0xfffe0c03d940
0xfffe0c03da80: 0xfffe0c03d948  0xfffe0c03d950
0xfffe0c03da90: 0xfffe0c03d958  0xfffe0c03d960
0xfffe0c03daa0: 0xfffe0c03d968  0xfffe0c03d8f0
0xfffe0c03dab0: 0xfffe0c03d8f8  0x0
0xfffe0c03dac0: 0x0     0x0
0xfffe0c03dad0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe0c03dae0: 0x0     0x0
0xfffe0c03daf0: 0x0     0x0
0xfffe0c03db00: 0x0     0x0
0xfffe0c03db10: 0x0     0x0
0xfffe0c03db20: 0x0     0x0
0xfffe0c03db30: 0x0     0x0
0xfffe0c03db40: 0x0     0x0
0xfffe0c03db50: 0x0     0x0
0xfffe0c03db60: 0x0     0x0
0xfffe0c03db70: 0x0     0x0
0xfffe0c03db80: 0x0     0x0
0xfffe0c03db90: 0x0     0x0
0xfffe0c03dba0: 0x0     0x0
0xfffe0c03dbb0: 0x0     0x0
0xfffe0c03dbc0: 0x0     0x0
0xfffe0c03dbd0: 0x0     0x0
0xfffe0c03dbe0: 0x0     0x0
0xfffe0c03dbf0: 0x0     0x0
0xfffe0c03dc00: 0xfffe0c03d970  0xfffe0c03d978
0xfffe0c03dc10: 0xfffe0c03d980  0xfffe0c03d988
0xfffe0c03dc20: 0xfffe0c03d990  0xfffe0c03d998
0xfffe0c03dc30: 0xfffe0c03d9a0  0xfffe0c03d9a8
0xfffe0c03dc40: 0x0     0x0
0xfffe0c03dc50: 0x0     0x0
0xfffe0c03dc60: 0x0     0x0
0xfffe0c03dc70: 0x0     0x0
0xfffe0c03dc80: 0x0     0x0
0xfffe0c03dc90: 0x0     0x0
0xfffe0c03dca0: 0x0     0x0
0xfffe0c03dcb0: 0x0     0x0
0xfffe0c03dcc0: 0x0     0x0
0xfffe0c03dcd0: 0xfffe0c03e7c0  0x404850 <__cxa_throw+144>
0xfffe0c03dce0: 0x0     0x0
0xfffe0c03dcf0: 0x0     0x40f758 <_Unwind_RaiseException>
0xfffe0c03dd00: 0x4000000000000000      0x0
0xfffe0c03dd10: 0x0     0x0
0xfffe0c03dd20: 0x0     0x0
0xfffe0c03dd30: 0x0     0x0
0xfffe0c03dd40: 0x0     0x0
0xfffe0c03dd50: 0x0     0x0
0xfffe0c03dd60: 0x0     0x0
0xfffe0c03dd70: 0x0     0x0
0xfffe0c03dd80: 0xfffe0c03d900  0xfffe0c03d908
0xfffe0c03dd90: 0xfffe0c03d910  0xfffe0c03d918
0xfffe0c03dda0: 0x0     0x0
0xfffe0c03ddb0: 0x0     0x0
0xfffe0c03ddc0: 0x0     0x0
0xfffe0c03ddd0: 0x0     0x0
```

```
0xfffe0c03dde0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe0c03ddf0: 0x0     0x0
0xfffe0c03de00: 0x0     0x0
0xfffe0c03de10: 0x0     0xfffe0c03e7d0
0xfffe0c03de20: 0xfffe0c03e7d8  0xfffe0c03d930
0xfffe0c03de30: 0xfffe0c03d938  0xfffe0c03d940
0xfffe0c03de40: 0xfffe0c03d948  0xfffe0c03d950
0xfffe0c03de50: 0xfffe0c03d958  0xfffe0c03d960
0xfffe0c03de60: 0xfffe0c03d968  0xfffe0c03e7f0
0xfffe0c03de70: 0xfffe0c03e7f8  0xfffe0c03d8e8
0xfffe0c03de80: 0x0     0x0
0xfffe0c03de90: 0x0     0x0
0xfffe0c03dea0: 0x0     0x0
0xfffe0c03deb0: 0x0     0x0
0xfffe0c03dec0: 0x0     0x0
0xfffe0c03ded0: 0x0     0x0
0xfffe0c03dee0: 0x0     0x0
0xfffe0c03def0: 0x0     0x0
0xfffe0c03df00: 0x0     0x0
0xfffe0c03df10: 0x0     0x0
0xfffe0c03df20: 0x0     0x0
0xfffe0c03df30: 0x0     0x0
0xfffe0c03df40: 0x0     0x0
0xfffe0c03df50: 0x0     0x0
0xfffe0c03df60: 0x0     0x0
0xfffe0c03df70: 0x0     0x0
0xfffe0c03df80: 0x0     0x0
0xfffe0c03df90: 0x0     0x0
0xfffe0c03dfa0: 0x0     0x0
0xfffe0c03dfb0: 0x0     0x0
0xfffe0c03dfc0: 0xfffe0c03d970  0xfffe0c03d978
0xfffe0c03dfd0: 0xfffe0c03d980  0xfffe0c03d988
0xfffe0c03dfe0: 0xfffe0c03d990  0xfffe0c03d998
0xfffe0c03dff0: 0xfffe0c03d9a0  0xfffe0c03d9a8
0xfffe0c03e000: 0x0     0x0
0xfffe0c03e010: 0x0     0x0
0xfffe0c03e020: 0x0     0x0
0xfffe0c03e030: 0x0     0x0
0xfffe0c03e040: 0x0     0x0
0xfffe0c03e050: 0x0     0x0
0xfffe0c03e060: 0x0     0x0
0xfffe0c03e070: 0x0     0x0
0xfffe0c03e080: 0x0     0x0
0xfffe0c03e090: 0xfffe0c03e800  0x40326c <procA()+72>
0xfffe0c03e0a0: 0x4bd09c        0x0
0xfffe0c03e0b0: 0x0     0x403224 <procA()>
0xfffe0c03e0c0: 0x4000000000000000      0x0
0xfffe0c03e0d0: 0x0     0x0
0xfffe0c03e0e0: 0x0     0x0
0xfffe0c03e0f0: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe0c03e100: 0x0     0x0
0xfffe0c03e110: 0x0     0x0
0xfffe0c03e120: 0x0     0x0
0xfffe0c03e130: 0x0     0x0
0xfffe0c03e140: 0x0     0x0
0xfffe0c03e150: 0x0     0x0
0xfffe0c03e160: 0x0     0x0
0xfffe0c03e170: 0x0     0x0
```

```
0xfffe0c03e180: 0x0     0x0
0xfffe0c03e190: 0x0     0x0
0xfffe0c03e1a0: 0x0     0x0
0xfffe0c03e1b0: 0x0     0x0
0xfffe0c03e1c0: 0x0     0x0
0xfffe0c03e1d0: 0x0     0x0
0xfffe0c03e1e0: 0x0     0x0
0xfffe0c03e1f0: 0x0     0x0
0xfffe0c03e200: 0x0     0x0
0xfffe0c03e210: 0x0     0x0
0xfffe0c03e220: 0x0     0x0
0xfffe0c03e230: 0x0     0x0
0xfffe0c03e240: 0x0     0x0
0xfffe0c03e250: 0x0     0x0
0xfffe0c03e260: 0x0     0x0
0xfffe0c03e270: 0x0     0x0
0xfffe0c03e280: 0x0     0x0
0xfffe0c03e290: 0x0     0x0
0xfffe0c03e2a0: 0x0     0x0
0xfffe0c03e2b0: 0x0     0x0
0xfffe0c03e2c0: 0x0     0x0
0xfffe0c03e2d0: 0x0     0x0
0xfffe0c03e2e0: 0x0     0x0
0xfffe0c03e2f0: 0x0     0x0
0xfffe0c03e300: 0x0     0x0
0xfffe0c03e310: 0xfffffffffffffff0      0x1
0xfffe0c03e320: 0xfffffffffffffff8      0x1
0xfffe0c03e330: 0x0     0x0
0xfffe0c03e340: 0x0     0x0
0xfffe0c03e350: 0x0     0x0
0xfffe0c03e360: 0x0     0x0
0xfffe0c03e370: 0x0     0x0
0xfffe0c03e380: 0x0     0x0
0xfffe0c03e390: 0x0     0x0
0xfffe0c03e3a0: 0x0     0x0
0xfffe0c03e3b0: 0x0     0x0
0xfffe0c03e3c0: 0x0     0x0
0xfffe0c03e3d0: 0x0     0x0
0xfffe0c03e3e0: 0x0     0x0
0xfffe0c03e3f0: 0x0     0x0
0xfffe0c03e400: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe0c03e410: 0x0     0x0
0xfffe0c03e420: 0x0     0x0
0xfffe0c03e430: 0x0     0x0
0xfffe0c03e440: 0x0     0x0
0xfffe0c03e450: 0x0     0x0
0xfffe0c03e460: 0x0     0x0
0xfffe0c03e470: 0x0     0x0
0xfffe0c03e480: 0x0     0x0
0xfffe0c03e490: 0x0     0x0
0xfffe0c03e4a0: 0xfffe0c03e610  0x433888 <sleep+228>
0xfffe0c03e4b0: 0xffffffff      0x10000
0xfffe0c03e4c0: 0x0     0x0
0xfffe0c03e4d0: 0x0     0x0
0xfffe0c03e4e0: 0x0     0x0
0xfffe0c03e4f0: 0x0     0x0
0xfffe0c03e500: 0x0     0x0
0xfffe0c03e510: 0x0     0x0
0xfffe0c03e520: 0x0     0x0
```

```
0xfffe0c03e530: 0x0     0x0
0xfffe0c03e540: 0x0     0x0
0xfffe0c03e550: 0x0     0x0
0xfffe0c03e560: 0x0     0x0
0xfffe0c03e570: 0xfffe0c03e610  0x4338a8 <sleep+260>
0xfffe0c03e580: 0xffffffff      0x10000
0xfffe0c03e590: 0x0     0x0
0xfffe0c03e5a0: 0x0     0x0
0xfffe0c03e5b0: 0xfffe0c03e610  0x41ae18 <nanosleep+24>
0xfffe0c03e5c0: 0x2     0x0
0xfffe0c03e5d0: 0x4338b4 <sleep+272>    0x0
0xfffe0c03e5e0: 0xfffe0c03e650  0x0
0xfffe0c03e5f0: 0x0     0x0
0xfffe0c03e600: 0x0     0x0
0xfffe0c03e610: 0xfffe0c03e800  0x403250 <procA()+44>
0xfffe0c03e620: 0xfffe0c03f070  0x0
0xfffe0c03e630: 0x4d0000        0x403338 <thread_two(void*)>
0xfffe0c03e640: 0x0     0x0
0xfffe0c03e650: 0x14    0x0
0xfffe0c03e660: 0x0     0x0
0xfffe0c03e670: 0x0     0x0
0xfffe0c03e680: 0x0     0x0
0xfffe0c03e690: 0x0     0x0
0xfffe0c03e6a0: 0x0     0x0
0xfffe0c03e6b0: 0x0     0x0
0xfffe0c03e6c0: 0x0     0x0
0xfffe0c03e6d0: 0x0     0x0
0xfffe0c03e6e0: 0x10000 0x0
0xfffe0c03e6f0: 0x0     0x0
0xfffe0c03e700: 0x0     0x0
0xfffe0c03e710: 0x0     0x0
--Type <RET> for more, q to quit, c to continue without paging--
0xfffe0c03e720: 0x0     0x0
0xfffe0c03e730: 0x0     0x0
0xfffe0c03e740: 0x0     0x0
0xfffe0c03e750: 0x0     0x0
0xfffe0c03e760: 0x0     0x0
0xfffe0c03e770: 0x0     0x0
0xfffe0c03e780: 0x0     0x0
0xfffe0c03e790: 0x0     0x0
0xfffe0c03e7a0: 0x0     0x0
0xfffe0c03e7b0: 0x0     0x0
0xfffe0c03e7c0: 0x0     0x0
```

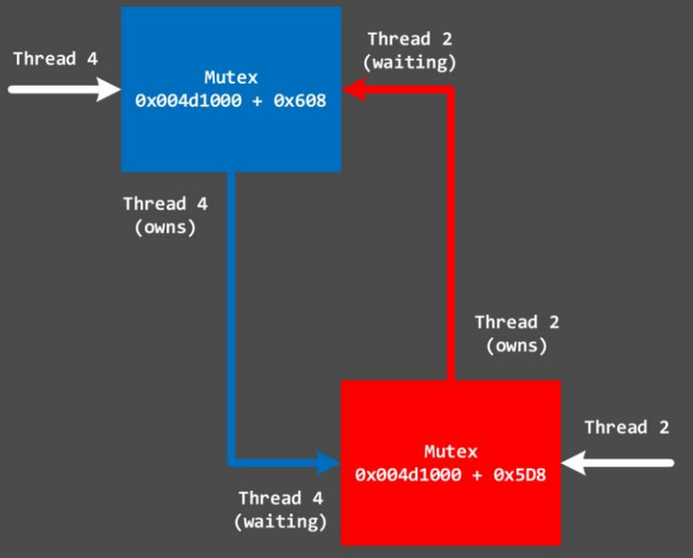
Note: We see a reference **0x40326c <procA()+72>** from the exception processing block in *procA* and also **0x404850 <__cxa_throw+144>**. We check whether the symbolic information we found is not coincidental:

```
(gdb) disassemble __cxa_throw
Dump of assembler code for function __cxa_throw:
   0x00000000004047c0 <+0>:     stp     x29, x30, [sp, #-48]!
   0x00000000004047c4 <+4>:     mov     x29, sp
   0x00000000004047c8 <+8>:     stp     x19, x20, [sp, #16]
   0x00000000004047cc <+12>:    mov     x19, x0
   0x00000000004047d0 <+16>:    nop
   0x00000000004047d4 <+20>:    str     x1, [x29, #40]
   0x00000000004047d8 <+24>:    str     x2, [x29, #32]
   0x00000000004047dc <+28>:    bl      0x403c34 <__cxa_get_globals>
   0x00000000004047e0 <+32>:    ldr     w3, [x0, #8]
   0x00000000004047e4 <+36>:    ldr     x1, [x29, #40]
   0x00000000004047e8 <+40>:    ldr     x2, [x29, #32]
```

```
   0x00000000004047ec <+44>:    add     w3, w3, #0x1
   0x00000000004047f0 <+48>:    str     w3, [x0, #8]
   0x00000000004047f4 <+52>:    mov     w0, #0x1                        // #1
   0x00000000004047f8 <+56>:    stur    w0, [x19, #-128]
   0x00000000004047fc <+60>:    stur    x1, [x19, #-112]
   0x0000000000404800 <+64>:    stur    x2, [x19, #-104]
   0x0000000000404804 <+68>:    adrp    x0, 0x4cf000 <vtable for std::exception+24>
   0x0000000000404808 <+72>:    ldr     x0, [x0, #4016]
   0x000000000040480c <+76>:    mov     x1, #0x2b00                     // #11008
   0x0000000000404810 <+80>:    ldr     x0, [x0]
   0x0000000000404814 <+84>:    movk    x1, #0x432b, lsl #16
   0x0000000000404818 <+88>:    stur    x0, [x19, #-96]
   0x000000000040481c <+92>:    adrp    x0, 0x4cf000 <vtable for std::exception+24>
   0x0000000000404820 <+96>:    ldr     x0, [x0, #3960]
   0x0000000000404824 <+100>:   movk    x1, #0x5543, lsl #32
   0x0000000000404828 <+104>:   ldr     x0, [x0]
   0x000000000040482c <+108>:   movk    x1, #0x474e, lsl #48
   0x0000000000404830 <+112>:   stur    x0, [x19, #-88]
   0x0000000000404834 <+116>:   adrp    x0, 0x404000 <base_of_encoded_value(unsigned char,
_Unwind_Context*)+72>
   0x0000000000404838 <+120>:   sub     x20, x19, #0x20
   0x000000000040483c <+124>:   add     x0, x0, #0x75c
   0x0000000000404840 <+128>:   stur    x1, [x19, #-32]
   0x0000000000404844 <+132>:   stur    x0, [x19, #-24]
   0x0000000000404848 <+136>:   mov     x0, x20
   0x000000000040484c <+140>:   bl      0x40f758 <_Unwind_RaiseException>
   0x0000000000404850 <+144>:   mov     x0, x20
   0x0000000000404854 <+148>:   bl      0x4039b4 <__cxa_begin_catch>
   0x0000000000404858 <+152>:   bl      0x4046ec <std::terminate()>
End of assembler dump.
```

Note: Full debug symbols make it possible to find mutex ownership faster. We will do that in the next exercise, A12.

Deadlock (A64, GDB)



Exercise A11 (A64, WinDbg Preview)

Goal: Learn how to identify synchronization wait chains, deadlocks, hidden and handled exceptions.

Patterns: Wait Chain (Mutex Objects); Deadlock (Mutex Objects); Disassembly Hole.

1. Launch WinDbg Preview.

2. Load *App11.core.11410* dump file from the A64\App11 folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App11\App11.core.11410]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
*** WARNING: Unable to verify timestamp for App11
App11+0x1ae24:
00000000`0041ae24 d4000001 svc         #0
```

3. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App11\App11.log
Opened log file 'C:\ALCDA2\A64\App11\App11.log'
```

4. Specify the dump folder as the symbol path and reload symbols:

```
0:000> .sympath+ C:\ALCDA2\A64\App11\
Symbol search path is: srv*;C:\ALCDA2\A64\App11\
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app11\

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App11\
*** WARNING: Unable to verify timestamp for App11
```

```
0:000> .reload
..
*** WARNING: Unable to verify timestamp for App11

************* Symbol Loading Error Summary **************
Module name            Error
App11                  The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

5. List all thread stack traces and identify possible wait chain and deadlock:

```
0:000> ~*k

Unable to get thread data for thread 0
.  0  Id: 2c92.2c93 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`0c84e5e0 00000000`004338b4     App11!_libc_nanosleep+0x24
01 0000fffe`0c84e620 00000000`004032d8     App11!sleep+0x110
02 0000fffe`0c84e810 00000000`004032ec     App11!bar_one+0x10
03 0000fffe`0c84e820 00000000`00403304     App11!foo_one+0xc
04 0000fffe`0c84e830 00000000`004130a4     App11!thread_one+0x10
05 0000fffe`0c84e850 00000000`00438760     App11!start_thread+0xb4
06 0000fffe`0c84e980 ffffffff`ffffffff     App11!thread_start+0x30
07 0000fffe`0c84e980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 1
   1  Id: 2c92.2c94 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`0c03e7d0 00000000`00414ea4     App11!_lll_lock_wait+0x3c
01 0000fffe`0c03e7d0 00000000`0040325c     App11!_pthread_mutex_lock+0xe8
02 0000fffe`0c03e800 00000000`0040331c     App11!procA+0x38
03 0000fffe`0c03e810 00000000`00403330     App11!bar_two+0xc
04 0000fffe`0c03e820 00000000`00403348     App11!foo_two+0xc
05 0000fffe`0c03e830 00000000`004130a4     App11!thread_two+0x10
06 0000fffe`0c03e850 00000000`00438760     App11!start_thread+0xb4
07 0000fffe`0c03e980 ffffffff`ffffffff     App11!thread_start+0x30
08 0000fffe`0c03e980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 2
   2  Id: 2c92.2c9d Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`0b82e5e0 00000000`004338b4     App11!_libc_nanosleep+0x24
01 0000fffe`0b82e620 00000000`00403364     App11!sleep+0x110
02 0000fffe`0b82e810 00000000`00403378     App11!bar_three+0x10
03 0000fffe`0b82e820 00000000`00403390     App11!foo_three+0xc
04 0000fffe`0b82e830 00000000`004130a4     App11!thread_three+0x10
05 0000fffe`0b82e850 00000000`00438760     App11!start_thread+0xb4
06 0000fffe`0b82e980 ffffffff`ffffffff     App11!thread_start+0x30
07 0000fffe`0b82e980 00000000`00000000     0xffffffff`ffffffff
```

```
Unable to get thread data for thread 3
   3  Id: 2c92.2c9e Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`0b01e7d0 00000000`00414ea4     App11!_lll_lock_wait+0x3c
01 0000fffe`0b01e7d0 00000000`004032a0     App11!_pthread_mutex_lock+0xe8
02 0000fffe`0b01e800 00000000`004033a8     App11!procB+0x20
03 0000fffe`0b01e810 00000000`004033bc     App11!bar_four+0xc
04 0000fffe`0b01e820 00000000`004033d4     App11!foo_four+0xc
05 0000fffe`0b01e830 00000000`004130a4     App11!thread_four+0x10
06 0000fffe`0b01e850 00000000`00438760     App11!start_thread+0xb4
07 0000fffe`0b01e980 ffffffff`ffffffff     App11!thread_start+0x30
08 0000fffe`0b01e980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 4
   4  Id: 2c92.2c9f Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffe`0a80e5e0 00000000`004338b4     App11!_libc_nanosleep+0x24
01 0000fffe`0a80e620 00000000`004033f0     App11!sleep+0x110
02 0000fffe`0a80e810 00000000`00403404     App11!bar_five+0x10
03 0000fffe`0a80e820 00000000`0040341c     App11!foo_five+0xc
04 0000fffe`0a80e830 00000000`004130a4     App11!thread_five+0x10
05 0000fffe`0a80e850 00000000`00438760     App11!start_thread+0xb4
06 0000fffe`0a80e980 ffffffff`ffffffff     App11!thread_start+0x30
07 0000fffe`0a80e980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 5
   5  Id: 2c92.2c92 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`d8b393c0 00000000`004338b4     App11!_libc_nanosleep+0x24
01 0000ffff`d8b39400 00000000`004034e0     App11!sleep+0x110
02 0000ffff`d8b395f0 00000000`0041d0bc     App11!main+0xb8
03 0000ffff`d8b39640 00000000`004030a8     App11!_libc_start_main+0x304
04 0000ffff`d8b397a0 00000000`00000000     App11!start+0x4c
```

6. Check **thread #3** and its waiting code:

```
0:000> ~3k
 # Child-SP          RetAddr               Call Site
00 0000fffe`0b01e7d0 00000000`00414ea4     App11!_lll_lock_wait+0x3c
01 0000fffe`0b01e7d0 00000000`004032a0     App11!_pthread_mutex_lock+0xe8
02 0000fffe`0b01e800 00000000`004033a8     App11!procB+0x20
03 0000fffe`0b01e810 00000000`004033bc     App11!bar_four+0xc
04 0000fffe`0b01e820 00000000`004033d4     App11!foo_four+0xc
05 0000fffe`0b01e830 00000000`004130a4     App11!thread_four+0x10
06 0000fffe`0b01e850 00000000`00438760     App11!start_thread+0xb4
07 0000fffe`0b01e980 ffffffff`ffffffff     App11!thread_start+0x30
08 0000fffe`0b01e980 00000000`00000000     0xffffffff`ffffffff

0:000> uf App11!procB
App11!procB:
00000000`00403280 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`00403284 910003fd mov         fp,sp
00000000`00403288 d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`0040328c 91182000 add         x0,x0,#0x608
00000000`00403290 940046cb bl          App11!_pthread_mutex_lock (00000000`00414dbc)
00000000`00403294 d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403298 91176000 add         x0,x0,#0x5D8
00000000`0040329c 940046c8 bl          App11!_pthread_mutex_lock (00000000`00414dbc)
00000000`004032a0 528003c0 mov         w0,#0x1E
00000000`004032a4 9400c140 bl          App11!sleep (00000000`004337a4)
```

```
00000000`004032a8 d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`004032ac 91176000 add         x0,x0,#0x5D8
00000000`004032b0 940046c3 bl          App11!_pthread_mutex_unlock (00000000`00416054)
00000000`004032b4 d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`004032b8 91182000 add         x0,x0,#0x608
00000000`004032bc 940046c0 bl          App11!_pthread_mutex_unlock (00000000`00416054)
00000000`004032c0 a8c17bfd ldp         fp,lr,[sp],#0x10
00000000`004032c4 d65f03c0 ret
```

Note: We see **thread #3** owns the mutex **00000000`004d1000 + 0x608** but is waiting for the mutex **00000000`004d1000 + 0x5D8**.

7. Check **thread #1** and its waiting code:

```
0:000> ~1k
 # Child-SP          RetAddr               Call Site
00 0000fffe`0c03e7d0 00000000`00414ea4     App11!_lll_lock_wait+0x3c
01 0000fffe`0c03e7d0 00000000`0040325c     App11!_pthread_mutex_lock+0xe8
02 0000fffe`0c03e800 00000000`0040331c     App11!procA+0x38
03 0000fffe`0c03e810 00000000`00403330     App11!bar_two+0xc
04 0000fffe`0c03e820 00000000`00403348     App11!foo_two+0xc
05 0000fffe`0c03e830 00000000`004130a4     App11!thread_two+0x10
06 0000fffe`0c03e850 00000000`00438760     App11!start_thread+0xb4
07 0000fffe`0c03e980 ffffffff`ffffffff     App11!thread_start+0x30
08 0000fffe`0c03e980 00000000`00000000     0xffffffff`ffffffff

0:000> uf App11!procA
App11!procA:
00000000`00403224 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`00403228 910003fd mov         fp,sp
00000000`0040322c d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403230 91176000 add         x0,x0,#0x5D8
00000000`00403234 940046e2 bl          App11!_pthread_mutex_lock (00000000`00414dbc)
00000000`00403238 97fffff2 bl          App11!procC (00000000`00403200)
00000000`0040323c d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403240 91176000 add         x0,x0,#0x5D8
00000000`00403244 94004b84 bl          App11!_pthread_mutex_unlock (00000000`00416054)
00000000`00403248 52800280 mov         w0,#0x14
00000000`0040324c 9400c156 bl          App11!sleep (00000000`004337a4)
00000000`00403250 d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403254 91182000 add         x0,x0,#0x608
00000000`00403258 940046d9 bl          App11!_pthread_mutex_lock (00000000`00414dbc)
00000000`0040325c d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403260 91182000 add         x0,x0,#0x608
00000000`00403264 94004b7c bl          App11!_pthread_mutex_unlock (00000000`00416054)
00000000`00403268 14000004 b           App11!Z5procAv+0x54 (00000000`00403278)  Branch

App11!procA+0x54:
00000000`00403278 a8c17bfd ldp         fp,lr,[sp],#0x10
00000000`0040327c d65f03c0 ret
```

Note: There's a hole in function disassembly. We can disassemble the missing part manually:

```
0:000> u 00000000`00403268
App11!procA+0x44:
00000000`00403268 14000004 b           App11!procA+0x54 (00000000`00403278)
00000000`0040326c 940001d2 bl          App11!_cxa_begin_catch (00000000`004039b4)
00000000`00403270 940001fa bl          App11!_cxa_end_catch (00000000`00403a58)
00000000`00403274 17fffff5 b           App11!procA+0x24 (00000000`00403248)
```

```
00000000`00403278 a8c17bfd ldp         fp,lr,[sp],#0x10
00000000`0040327c d65f03c0 ret
App11!procB:
00000000`00403280 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`00403284 910003fd mov         fp,sp
```

Note: We put all reconstructed disassembly pieces together:

```
App11!procA:
00000000`00403224 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`00403228 910003fd mov         fp,sp
00000000`0040322c d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403230 91176000 add         x0,x0,#0x5D8
00000000`00403234 940046e2 bl          App11!_pthread_mutex_lock (00000000`00414dbc)
00000000`00403238 97fffff2 bl          App11!procC (00000000`00403200)
00000000`0040323c d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403240 91176000 add         x0,x0,#0x5D8
00000000`00403244 94004b84 bl          App11!_pthread_mutex_unlock (00000000`00416054)
00000000`00403248 52800280 mov         w0,#0x14
00000000`0040324c 9400c156 bl          App11!sleep (00000000`004337a4)
00000000`00403250 d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403254 91182000 add         x0,x0,#0x608
00000000`00403258 940046d9 bl          App11!_pthread_mutex_lock (00000000`00414dbc)
00000000`0040325c d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403260 91182000 add         x0,x0,#0x608
00000000`00403264 94004b7c bl          App11!_pthread_mutex_unlock (00000000`00416054)
00000000`00403268 14000004 b           App11!procA+0x54 (00000000`00403278)  Branch
00000000`00403268 14000004 b           App11!procA+0x54 (00000000`00403278)
00000000`0040326c 940001d2 bl          App11!_cxa_begin_catch (00000000`004039b4)
00000000`00403270 940001fa bl          App11!_cxa_end_catch (00000000`00403a58)
00000000`00403274 17fffff5 b           App11!procA+0x24 (00000000`00403248)
App11!procA+0x54:
00000000`00403278 a8c17bfd ldp         fp,lr,[sp],#0x10
00000000`0040327c d65f03c0 ret
```

Note: We see that **thread #1** is waiting for the **00000000`004d1000 + 0x608** mutex but shouldn't own the **00000000`004d1000 + 0x5D8** mutex because it should have unlocked it unless something happened in **procC**. We also notice catch exception processing which transfers execution for the block of code waiting for the mutex **00000000`004d1000 + 0x608**.

8. Disassemble *procC* code:

```
0:000> uf App11!procC
App11!procC:
00000000`00403200 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`00403204 910003fd mov         fp,sp
00000000`00403208 d2800080 mov         x0,#4
00000000`0040320c 94000106 bl          App11!_cxa_allocate_exception (00000000`00403624)
00000000`00403210 b900001f str         wzr,[x0]
00000000`00403214 90000661 adrp        x1,App11!std::exception+0x18 (00000000`004cf000)
00000000`00403218 91160021 add         x1,x1,#0x580
00000000`0040321c d2800002 mov         x2,#0
00000000`00403220 94000568 bl          App11!_cxa_throw (00000000`004047c0)
00000000`00403224 a9bf7bfd stp         fp,lr,[sp,#-0x10]!
00000000`00403228 910003fd mov         fp,sp
00000000`0040322c d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403230 91176000 add         x0,x0,#0x5D8
00000000`00403234 940046e2 bl          App11!_pthread_mutex_lock (00000000`00414dbc)
00000000`00403238 97fffff2 bl          App11!procC (00000000`00403200)
```

```
00000000`0040323c d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403240 91176000 add         x0,x0,#0x5D8
00000000`00403244 94004b84 bl          App11!_pthread_mutex_unlock (00000000`00416054)
00000000`00403248 52800280 mov         w0,#0x14
00000000`0040324c 9400c156 bl          App11!sleep (00000000`004337a4)
00000000`00403250 d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403254 91182000 add         x0,x0,#0x608
00000000`00403258 940046d9 bl          App11!_pthread_mutex_lock (00000000`00414dbc)
00000000`0040325c d0000660 adrp        x0,App11!main_arena+0x850 (00000000`004d1000)
00000000`00403260 91182000 add         x0,x0,#0x608
00000000`00403264 94004b7c bl          App11!_pthread_mutex_unlock (00000000`00416054)
00000000`00403268 14000004 b           App11!procA+0x54 (00000000`00403278)  Branch

App11!procA+0x54:
00000000`00403278 a8c17bfd ldp         fp,lr,[sp],#0x10
00000000`0040327c d65f03c0 ret
```

Note: We see that code throws an exception, so perhaps it was caught in the caller *procA,* and mutex unlock wasn't called, thus causing a deadlock.

9. Check if there was any exception processing (we use the **dpS** command to omit values without associated symbolic information):

```
0:001> dpS sp-2000 sp
00000000`0040f150 App11!uw_update_context+0x18
 00000000`004e9558 App11!_default_pthread_attr
 00000000`004e3000 App11!ZL16emergency_buffer+0xfd90
 00000000`004e3000 App11!ZL16emergency_buffer+0xfd90
 00000000`00410ec0 App11!search_object+0x204
 00000000`004ad518 App11!$d+0x2c
 00000000`00410e48 App11!search_object+0x18c
 00000000`004117d0 App11!Unwind_Find_FDE+0x174
 00000000`004d15a8 App11!object.6205
 00000000`0040323b App11!procA+0x17
 00000000`004e3000 App11!ZL16emergency_buffer+0xfd90
 00000000`004cf000 App11!vtable for std::exception+0x18
 00000000`004e9558 App11!_default_pthread_attr
 00000000`0041178c App11!Unwind_Find_FDE+0x130
 00000000`004d15a8 App11!object.6205
 00000000`00403224 App11!procA
 00000000`00411778 App11!Unwind_Find_FDE+0x11c
 00000000`0040e374 App11!uw_frame_state_for+0x5cc
 00000000`00404228 App11!_gxx_personality_v0+0xf0
 00000000`0040f254 App11!Unwind_RaiseException_Phase2+0x70
 00000000`004e9558 App11!_default_pthread_attr
 00000000`0040f218 App11!Unwind_RaiseException_Phase2+0x34
 00000000`004bd09c App11!$d
 00000000`004e9558 App11!_default_pthread_attr
 00000000`0040f274 App11!Unwind_RaiseException_Phase2+0x90
 00000000`0040f89c App11!Unwind_RaiseException+0x144
 00000000`0040f8b0 App11!Unwind_RaiseException+0x158
 00000000`0040326c App11!procA+0x48
 00000000`004d0000 App11!+0x18
 00000000`00403338 App11!thread_two
 00000000`004e9558 App11!_default_pthread_attr
 00000000`00404850 App11!_cxa_throw+0x90
 00000000`0040f758 App11!Unwind_RaiseException
 00000000`0040326c App11!procA+0x48
 00000000`004bd09c App11!$d
```

```
00000000`00403224 App11!procA
00000000`00433888 App11!sleep+0xe4
00000000`004338a8 App11!sleep+0x104
00000000`0041ae18 App11!_libc_nanosleep+0x18
00000000`004338b4 App11!sleep+0x110
00000000`00403250 App11!procA+0x2c
00000000`004d0000 App11!+0x18
00000000`00403338 App11!thread_two
```

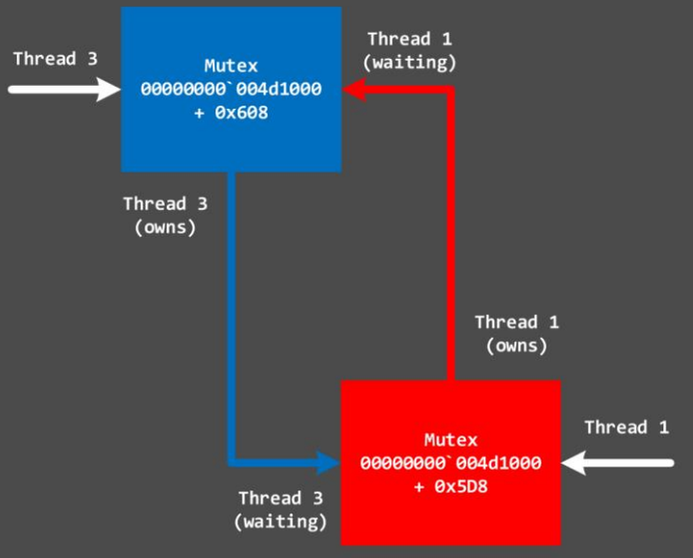
Note: We see a reference **00000000`0040326c App11!procA+0x48** from the exception processing block in *procA* and also **00000000`00404850 App11!_cxa_throw+0x90**. We check whether the symbolic information we found is not coincidental:

```
0:001> ub 00000000`00404850
App11!_cxa_throw+0x70:
00000000`00404830 f81a8260 stur        x0,[x19,#-0x58]
00000000`00404834 90000000 adrp
x0,App11!ZL21base_of_encoded_valuehP15_Unwind_Context+0x48 (00000000`00404000)
00000000`00404838 d1008274 sub         x20,x19,#0x20
00000000`0040483c 911d7000 add         x0,x0,#0x75C
00000000`00404840 f81e0261 stur        x1,[x19,#-0x20]
00000000`00404844 f81e8260 stur        x0,[x19,#-0x18]
00000000`00404848 aa1403e0 mov         x0,x20
00000000`0040484c 94002bc3 bl          App11!Unwind_RaiseException (00000000`0040f758)
```

13. We close logging before exiting WinDbg Preview:

```
0:001> .logclose
Closing open log file 'C:\ALCDA2\A64\App11\App11.log
```

Deadlock (A64, WinDbg)



Exercise A12

- **Goal:** Learn how to dump memory for post-processing, get the list of functions and module variables, load symbols, inspect arguments and local variables
- **Patterns:** Module Variable
- \ALCDA-Dumps\Exercise-A12-x64-GDB.pdf
- \ALCDA-Dumps\Exercise-A12-A64-GDB.pdf
- \ALCDA-Dumps\Exercise-A12-A64-WinDbg.pdf

Exercise A12 (x64, GDB)

Goal: Learn how to dump memory for post-processing, get the list of functions and module variables, load symbols, inspect arguments and local variables.

Patterns: Module Variable.

1. Load *App12.core.698* dump file and *App12* executable from the x64/App12 directory:

```
~/ALCDA2/x64/App12$ gdb -c App12.core.698 -se App12
GNU gdb (Debian 8.2.1-2+b3) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from App12...(no debugging symbols found)...done.
[New LWP 698]
[New LWP 699]
[New LWP 700]
[New LWP 701]
[New LWP 702]
[New LWP 703]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Core was generated by `./App12'.
#0  0x0000000000452970 in nanosleep ()
[Current thread is 1 (Thread 0x1438880 (LWP 698))]
```

2. List all thread stack traces:

```
(gdb) thread apply all bt

Thread 6 (Thread 0x7fbcedc7f700 (LWP 703)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000402024 in bar_five() ()
#3  0x0000000000402030 in foo_five() ()
#4  0x0000000000402044 in thread_five(void*) ()
#5  0x00000000004137f3 in start_thread ()
#6  0x000000000045512f in clone ()

Thread 5 (Thread 0x7fbcee480700 (LWP 702)):
#0  0x000000000041665c in __lll_lock_wait ()
#1  0x0000000000415294 in pthread_mutex_lock ()
#2  0x0000000000401f27 in procB() ()
#3  0x0000000000401fef in bar_four() ()
#4  0x0000000000401ffb in foo_four() ()
```

```
#5  0x000000000040200f in thread_four(void*) ()
#6  0x00000000004137f3 in start_thread ()
#7  0x000000000045512f in clone ()

Thread 4 (Thread 0x7fbceec81700 (LWP 701)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000401fbf in bar_three() ()
#3  0x0000000000401fcb in foo_three() ()
#4  0x0000000000401fdf in thread_three(void*) ()
#5  0x00000000004137f3 in start_thread ()
#6  0x000000000045512f in clone ()

Thread 3 (Thread 0x7fbcef482700 (LWP 700)):
#0  0x000000000041665c in __lll_lock_wait ()
#1  0x0000000000415294 in pthread_mutex_lock ()
#2  0x0000000000401eec in procA() ()
#3  0x0000000000401f8a in bar_two() ()
#4  0x0000000000401f96 in foo_two() ()
#5  0x0000000000401faa in thread_two(void*) ()
#6  0x00000000004137f3 in start_thread ()
#7  0x000000000045512f in clone ()

Thread 2 (Thread 0x7fbcefc83700 (LWP 699)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000401f5a in bar_one() ()
#3  0x0000000000401f66 in foo_one() ()
#4  0x0000000000401f7a in thread_one(void*) ()
#5  0x00000000004137f3 in start_thread ()
#6  0x000000000045512f in clone ()

Thread 1 (Thread 0x1438880 (LWP 698)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000402121 in main ()
```

3. *App12* is an executable with stripped-off debug symbols. Change the symbol file to *App12.debug,* which is the same executable as *App12* but with debug symbols included:

```
(gdb) symbol-file App12.debug
Reading symbols from App12.debug...done.
```

4. List all thread stack traces again:

```
(gdb) thread apply all bt

Thread 6 (Thread 0x7fbcedc7f700 (LWP 703)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000402024 in bar_five () at main.cpp:75
#3  0x0000000000402030 in foo_five () at main.cpp:75
#4  0x0000000000402044 in thread_five (arg=0x0) at main.cpp:75
#5  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000045512f in clone ()

Thread 5 (Thread 0x7fbcee480700 (LWP 702)):
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
#1  0x0000000000415294 in __pthread_mutex_lock (mutex=0x4d34a0 <mutexA>) at
../nptl/pthread_mutex_lock.c:80
```

```
#2  0x0000000000401f27 in procB () at main.cpp:48
#3  0x0000000000401fef in bar_four () at main.cpp:74
#4  0x0000000000401ffb in foo_four () at main.cpp:74
#5  0x000000000040200f in thread_four (arg=0x0) at main.cpp:74
#6  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045512f in clone ()

Thread 4 (Thread 0x7fbceec81700 (LWP 701)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000401fbf in bar_three () at main.cpp:73
#3  0x0000000000401fcb in foo_three () at main.cpp:73
#4  0x0000000000401fdf in thread_three (arg=0x0) at main.cpp:73
#5  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000045512f in clone ()

Thread 3 (Thread 0x7fbcef482700 (LWP 700)):
#0  __lll_lock_wait () at ../sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:103
#1  0x0000000000415294 in __pthread_mutex_lock (mutex=0x4d34e0 <mutex>) at
../nptl/pthread_mutex_lock.c:80
#2  0x0000000000401eec in procA () at main.cpp:41
#3  0x0000000000401f8a in bar_two () at main.cpp:72
#4  0x0000000000401f96 in foo_two () at main.cpp:72
#5  0x0000000000401faa in thread_two (arg=0x0) at main.cpp:72
#6  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#7  0x000000000045512f in clone ()

Thread 2 (Thread 0x7fbcefc83700 (LWP 699)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000401f5a in bar_one () at main.cpp:71
#3  0x0000000000401f66 in foo_one () at main.cpp:71
#4  0x0000000000401f7a in thread_one (arg=0x0) at main.cpp:71
#5  0x00000000004137f3 in start_thread (arg=<optimized out>) at pthread_create.c:486
#6  0x000000000045512f in clone ()

Thread 1 (Thread 0x1438880 (LWP 698)):
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000402121 in main (argc=1, argv=0x7ffee378a7d8) at main.cpp:91
```

5. Switch to thread #1 and its frame #2, and list arguments and locals:

```
(gdb) thread 1
[Switching to thread 1 (Thread 0x1438880 (LWP 698))]
#0  0x0000000000452970 in nanosleep ()

(gdb) bt
#0  0x0000000000452970 in nanosleep ()
#1  0x00000000004528fa in sleep ()
#2  0x0000000000402121 in main (argc=1, argv=0x7ffee378a7d8) at main.cpp:91

(gdb) frame 2
#2  0x0000000000402121 in main (argc=1, argv=0x7ffee378a7d8) at main.cpp:91
91          sleep(-1);

(gdb) info args
argc = 1
argv = 0x7ffee378a7d8
```

```
(gdb) info locals
No locals.
```

6. Examine the *argv* array:

```
(gdb) print argv[0]
$1 = 0x7ffee378b74e "./App12"

(gdb) print *argv@10
$2 = {0x7ffee378b74e "./App12", 0x0, 0x7ffee378b756 "SHELL=/bin/bash", 0x7ffee378b766
"HISTCONTROL=ignoreboth", 0x7ffee378b77d "WSL_DISTRO_NAME=Debian", 0x7ffee378b794
"NAME=DESKTOP-IS6V2L0",
  0x7ffee378b7a9 "PWD=/home/coredump/ALCDA2/x64/App12", 0x7ffee378b7cd "LOGNAME=coredump",
0x7ffee378b7de "MC_TMPDIR=/tmp/mc-coredump", 0x7ffee378b7f9 "MC_SID=14"}
```

7. Dump the region 0x4b0000 - 0x4bc000 to a binary file:

```
(gdb) dump memory mem.raw 0x4b0000 0x4bc000
```

8. List all functions:

```
(gdb) info functions
All defined functions:

File ../nptl/pthread_mutex_lock.c:
63:     int __pthread_mutex_lock(pthread_mutex_t *);
170:    static int __pthread_mutex_lock_full(pthread_mutex_t *);

File ../nptl/pthread_mutex_trylock.c:
34:     int __pthread_mutex_trylock(pthread_mutex_t *);

File ../nptl/sigaction.c:
22:     int __sigaction(int, const sigaction *, sigaction *);

File ../sysdeps/unix/sysv/linux/createthread.c:
49:     static int create_thread(pthread *, const pthread_attr *, _Bool *, void *, _Bool *);

File ../sysdeps/unix/sysv/linux/sigaction.c:
42:     int __libc_sigaction(int, const sigaction *, sigaction *);

File ../sysdeps/unix/sysv/linux/write.c:
24:     ssize_t __libc_write(int, const void *, size_t);

File ../sysdeps/unix/sysv/linux/x86/elision-conf.c:
75:     void _dl_tunable_set_elision_enable(tunable_val_t *);
97:     void _dl_tunable_set_elision_retry_try_xbegin(tunable_val_t *);
95:     void _dl_tunable_set_elision_skip_lock_busy(tunable_val_t *);
96:     void _dl_tunable_set_elision_skip_lock_internal_abort(tunable_val_t *);
98:     void _dl_tunable_set_elision_skip_trylock_internal_abort(tunable_val_t *);
104:    static void elision_init(int, char **, char **);

File ../sysdeps/unix/sysv/linux/x86/elision-lock.c:
45:     int __lll_lock_elision(int *, short *, int);

File ../sysdeps/unix/sysv/linux/x86/elision-trylock.c:
31:     int __lll_trylock_elision(int *, short *);

File ../sysdeps/unix/sysv/linux/x86/elision-unlock.c:
24:     int __lll_unlock_elision(int *, int);
```

```
File allocatestack.c:
787:    void __deallocate_stack(pthread *);
970:    pthread *__find_thread_by_id(pid_t);
810:    int __make_stacks_executable(void **);
1119:   int __nptl_setxid(xid_command *);
1098:   void __nptl_setxid_error(xid_command *, int);
293:    void __nptl_stacks_freeres(void);
1245:   void __pthread_init_static_tls(link_map *);
861:    void __reclaim_stacks(void);
1264:   void __wait_lookup_done(void);
316:    static int change_stack_perm(pthread *);
255:    static void free_stacks(size_t);
1015:   static void setxid_mark_thread(pthread *, xid_command *);
1072:   static int setxid_signal_thread(xid_command *, pthread *);
1052:   static void setxid_unmark_thread(pthread *, xid_command *);

File cleanup_compat.c:
39:     void _pthread_cleanup_pop(_pthread_cleanup_buffer *, int);
24:     void _pthread_cleanup_push(_pthread_cleanup_buffer *, void (*)(void *), void *);
--Type <RET> for more, q to quit, c to continue without paging--

File events.c:
24:     void __nptl_create_event(void);

File main.cpp:
75:     void bar_five();
74:     void bar_four();
71:     void bar_one();
73:     void bar_three();
72:     void bar_two();
75:     void foo_five();
74:     void foo_four();
71:     void foo_one();
73:     void foo_three();
72:     void foo_two();
79:     int main(int, char const**);
27:     void procA();
45:     void procB();
22:     void procC();
75:     void *thread_five(void*);
74:     void *thread_four(void*);
71:     void *thread_one(void*);
73:     void *thread_three(void*);
72:     void *thread_two(void*);

File nptl-init.c:
152:    void __nptl_set_robust(pthread *);
440:    size_t __pthread_get_minstack(const pthread_attr_t *);
269:    void __pthread_initialize_minimal_internal(void);
165:    static void sigcancel_handler(int, siginfo_t *, void *);
218:    static void sighandler_setxid(int, siginfo_t *, void *);

File pthread_cancel.c:
28:     int __pthread_cancel(pthread_t);

File pthread_create.c:
209:    pthread *__find_in_stack_list(pthread *);
344:    void __free_tcb(pthread *);
250:    void __nptl_deallocate_tsd(void);
```

```
632:    int __pthread_create_2_1(pthread_t *, const pthread_attr_t *, void *(*)(void *), void
*);
378:    static int start_thread(void *);

File pthread_getspecific.c:
24:     void *__pthread_getspecific(pthread_key_t);

File pthread_key_create.c:
25:     int __pthread_key_create(pthread_key_t *, void (*)(void *));

File pthread_key_delete.c:
25:     int __pthread_key_delete(pthread_key_t);

File pthread_mutex_init.c:
56:     int __pthread_mutex_init(pthread_mutex_t *, const pthread_mutexattr_t *);

File pthread_mutex_unlock.c:
354:    int __pthread_mutex_unlock(pthread_mutex_t *);
36:     int __pthread_mutex_unlock_usercnt(pthread_mutex_t *, int);
--Type <RET> for more, q to quit, c to continue without paging--
96:     static int __pthread_mutex_unlock_full(pthread_mutex_t *, int);

File pthread_once.c:
135:    int __pthread_once(pthread_once_t *, void (*)(void));
67:     static int __pthread_once_slow(pthread_once_t *, void (*)(void));
28:     static void clear_once_control(void *);

File pthread_setspecific.c:
25:     int __pthread_setspecific(pthread_key_t, const void *);

File tpp.c:
43:     void __init_sched_fifo_prio(void);
160:    int __pthread_current_priority(void);
52:     int __pthread_tpp_change_priority(int, int);

File unwind.c:
111:    void __pthread_unwind(__pthread_unwind_buf_t *);
132:    void __pthread_unwind_next(__pthread_unwind_buf_t *);
101:    static void unwind_cleanup(_Unwind_Reason_Code, _Unwind_Exception *);
39:     static _Unwind_Reason_Code unwind_stop(int, _Unwind_Action, _Unwind_Exception_Class,
_Unwind_Exception *, _Unwind_Context *, void *);

Non-debugging symbols:
0x00007ffee3797600  __vdso_gettimeofday
0x00007ffee3797600  gettimeofday
0x00007ffee3797730  __vdso_time
0x00007ffee3797730  time
0x00007ffee3797740  __vdso_clock_gettime
0x00007ffee3797740  clock_gettime
0x00007ffee37978a0  __vdso_clock_getres
0x00007ffee37978a0  clock_getres
0x00007ffee37978f0  __vdso_getcpu
0x00007ffee37978f0  getcpu
0x0000000000401000  _init
0x00000000004010f0  __cxxabiv1::__terminate(void (*)()) [clone .cold.0]
0x00000000004010fd  read_encoded_value_with_base(unsigned char, unsigned long, unsigned char
const*, unsigned long*) [clone .cold.4]
0x0000000000401102  __gxx_personality_v0.cold.5
0x000000000040110f  __cxa_call_unexpected.cold.6
```

```
0x00000000004011bc  (anonymous namespace)::pool::free(void*) [clone .constprop.2] [clone
.cold.5]
0x00000000004011cc  (anonymous namespace)::pool::allocate(unsigned long) [clone .constprop.3]
[clone .cold.6]
0x00000000004011dc  __gnu_cxx::__verbose_terminate_handler() [clone .cold.1]
0x000000000040125e  d_type.cold
0x0000000000401303  read_encoded_value_with_base.cold
0x0000000000401308  execute_cfa_program.cold
0x000000000040130d  execute_stack_op.cold
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
```

9. We can also list all available types or specific types:

```
(gdb) info types pthread_mutex_t
All types matching regular expression "pthread_mutex_t":

File ../sysdeps/nptl/bits/pthreadtypes.h:
72:     typedef union {
    __pthread_mutex_s __data;
    char __size[40];
    long __align;
} pthread_mutex_t;

File /usr/include/x86_64-linux-gnu/bits/pthreadtypes.h:
68:     pthread_mutex_t;
72:     typedef pthread_mutex_t pthread_mutex_t;
```

10. List all variables:

```
(gdb) info variables
All defined variables:

File ../nptl_db/db_info.c:
108:    const uint32_t _thread_db_const_thread_area;

File ../nptl_db/structs.def:
80:     const uint32_t _thread_db___nptl_initial_report_events[3];
78:     const uint32_t _thread_db___nptl_nthreads[3];
82:     const uint32_t _thread_db___pthread_keys[3];
96:     const uint32_t _thread_db_dtv_dtv[3];
113:    const uint32_t _thread_db_dtv_slotinfo_list_slotinfo[3];
93:     const uint32_t _thread_db_link_map_l_tls_modid[3];
94:     const uint32_t _thread_db_link_map_l_tls_offset[3];
62:     const uint32_t _thread_db_list_t_next[3];
63:     const uint32_t _thread_db_list_t_prev[3];
52:     const uint32_t _thread_db_pthread_cancelhandling[3];
56:     const uint32_t _thread_db_pthread_eventbuf[3];
57:     const uint32_t _thread_db_pthread_eventbuf_eventmask[3];
58:     const uint32_t _thread_db_pthread_eventbuf_eventmask_event_bits[3];
91:     const uint32_t _thread_db_pthread_key_data_level2_data[3];
48:     const uint32_t _thread_db_pthread_list[3];
59:     const uint32_t _thread_db_pthread_nextevent[3];
49:     const uint32_t _thread_db_pthread_report_events[3];
54:     const uint32_t _thread_db_pthread_schedparam_sched_priority[3];
53:     const uint32_t _thread_db_pthread_schedpolicy[3];
55:     const uint32_t _thread_db_pthread_specific[3];
51:     const uint32_t _thread_db_pthread_start_routine[3];
50:     const uint32_t _thread_db_pthread_tid[3];
61:     const uint32_t _thread_db_sizeof_list_t;
```

```
47:     const uint32_t _thread_db_sizeof_pthread;
90:     const uint32_t _thread_db_sizeof_pthread_key_data_level2;
68:     const uint32_t _thread_db_sizeof_td_eventbuf_t;
65:     const uint32_t _thread_db_sizeof_td_thr_events_t;
70:     const uint32_t _thread_db_td_eventbuf_t_eventdata[3];
69:     const uint32_t _thread_db_td_eventbuf_t_eventnum[3];
66:     const uint32_t _thread_db_td_thr_events_t_event_bits[3];

File ../sysdeps/unix/sysv/linux/x86/elision-conf.c:
33:     elision_config __elision_aconf;
56:     int __pthread_force_elision;
134:    void (* const__pthread_init_array[1])(int, char **, char **);

File allocatestack.c:
125:    list_t __stack_user;
121:    static uintptr_t in_flight_stack;
113:    static list_t stack_cache;
107:    static size_t stack_cache_actsize;
110:    static int stack_cache_lock;
106:    static size_t stack_cache_maxsize;
116:    static list_t stack_used;

File main.cpp:
20:     pthread_mutex_t mutexA;
20:     pthread_mutex_t mutexB;

File nptl-init.c:
44:     int *__libc_multiple_threads_ptr;
--Type <RET> for more, q to quit, c to continue without paging--
49:     size_t __static_tls_align_m1;
48:     size_t __static_tls_size;
212:    xid_command *__xidcmd;
266:    static _Bool __nptl_initial_report_events;
70:     static const char nptl_version[5];

File pthread_create.c:
53:     unsigned int __nptl_nthreads;
44:     int __pthread_debug;
50:     static pthread *__nptl_last_event;
47:     static td_thr_events_t __nptl_threads_events;

File pthread_mutex_init.c:
30:     static const pthread_mutexattr default_mutexattr;

File pthread_once.c:
24:     unsigned long __fork_generation;

File tpp.c:
30:     int __sched_fifo_max_prio;
29:     int __sched_fifo_min_prio;

File vars.c:
25:     pthread_attr __default_pthread_attr;
28:     int __default_pthread_attr_lock;
31:     int __is_smp;
41:     pthread_key_struct __pthread_keys[1024];
37:     int __pthread_multiple_threads;

Non-debugging symbols:
0x0000000000000000  __libc_resp
```

```
0x0000000000000008  _nl_current_LC_CTYPE
0x0000000000000010  __libc_tsd_LOCALE
0x0000000000000018  _nl_current_LC_MONETARY
0x0000000000000020  _nl_current_LC_NUMERIC
0x0000000000000028  (anonymous namespace)::get_global()::global
0x0000000000000038  __libc_errno
0x0000000000000040  __libc_tsd_CTYPE_TOLOWER
0x0000000000000048  __libc_tsd_CTYPE_TOUPPER
0x0000000000000050  __libc_tsd_CTYPE_B
0x0000000000000058  tcache
0x0000000000000060  tcache_shutting_down
0x0000000000000068  thread_arena
0x0000000000000070  current
0x0000000000000078  catch_hook
0x0000000000400000  __ehdr_start
0x0000000000400248  __rela_iplt_start
0x00000000004004d0  __rela_iplt_end
0x00000000004a1000  _IO_stdin_used
0x00000000004a1020  typeinfo name for __cxxabiv1::__fundamental_type_info
0x00000000004a1048  typeinfo name for void
0x00000000004a104a  typeinfo name for void*
0x00000000004a104d  typeinfo name for void const*
0x00000000004a1051  typeinfo name for bool
0x00000000004a1053  typeinfo name for bool*
0x00000000004a1056  typeinfo name for bool const*
0x00000000004a105a  typeinfo name for wchar_t
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
```

11. List segment info (also **info files**):

```
(gdb) info target
Symbols from "/home/coredump/ALCDA2/x64/App12/App12.debug".
Local core dump file:
        `/home/coredump/ALCDA2/x64/App12/App12.core.698', file type elf64-x86-64.
        0x0000000000401000 - 0x00000000004a1000 is load1
        0x00000000004cc000 - 0x00000000004d4000 is load2
        0x00000000004d4000 - 0x00000000004da000 is load3
        0x0000000001438000 - 0x000000000145b000 is load4
        0x00007fbce8000000 - 0x00007fbce8021000 is load5
        0x00007fbce8021000 - 0x00007fbcec000000 is load6
        0x00007fbced47f000 - 0x00007fbced480000 is load7
        0x00007fbced480000 - 0x00007fbcedc80000 is load8
        0x00007fbcedc80000 - 0x00007fbcedc81000 is load9
        0x00007fbcedc81000 - 0x00007fbcee481000 is load10
        0x00007fbcee481000 - 0x00007fbcee482000 is load11
        0x00007fbcee482000 - 0x00007fbceec82000 is load12
        0x00007fbceec82000 - 0x00007fbceec83000 is load13
        0x00007fbceec83000 - 0x00007fbcef483000 is load14
        0x00007fbcef483000 - 0x00007fbcef484000 is load15
        0x00007fbcef484000 - 0x00007fbcefc84000 is load16
        0x00007ffee376b000 - 0x00007ffee378c000 is load17
        0x00007ffee3797000 - 0x00007ffee3798000 is load18
Local exec file:
        `/home/coredump/ALCDA2/x64/App12/App12', file type elf64-x86-64.
        Entry point: 0x401d70
        0x0000000000400200 - 0x0000000000400220 is .note.ABI-tag
        0x0000000000400220 - 0x0000000000400244 is .note.gnu.build-id
        0x0000000000400248 - 0x00000000004004d0 is .rela.plt
        0x0000000000401000 - 0x0000000000401017 is .init
```

```
        0x0000000000401018 - 0x00000000004010f0 is .plt
        0x00000000004010f0 - 0x000000000049f55b is .text
        0x000000000049f560 - 0x00000000004a0107 is __libc_freeres_fn
        0x00000000004a0108 - 0x00000000004a0111 is .fini
        0x00000000004a1000 - 0x00000000004bcd20 is .rodata
        0x00000000004bcd20 - 0x00000000004bcd21 is .stapsdt.base
        0x00000000004bcd28 - 0x00000000004caa38 is .eh_frame
        0x00000000004caa38 - 0x00000000004cac13 is .gcc_except_table
        0x00000000004cc848 - 0x00000000004cc870 is .tdata
        0x00000000004cc870 - 0x00000000004cc8c8 is .tbss
        0x00000000004cc870 - 0x00000000004cc878 is .preinit_array
        0x00000000004cc878 - 0x00000000004cc890 is .init_array
        0x00000000004cc890 - 0x00000000004cc8a0 is .fini_array
        0x00000000004cc8a0 - 0x00000000004d0ef4 is .data.rel.ro
        0x00000000004d0ef8 - 0x00000000004d1000 is .got
        0x00000000004d1000 - 0x00000000004d10f0 is .got.plt
        0x00000000004d1100 - 0x00000000004d2c48 is .data
        0x00000000004d2c48 - 0x00000000004d2ca8 is __libc_subfreeres
        0x00000000004d2cc0 - 0x00000000004d3428 is __libc_IO_vtables
        0x00000000004d3428 - 0x00000000004d3430 is __libc_atexit
        0x00000000004d3440 - 0x00000000004d9628 is .bss
        0x00000000004d9628 - 0x00000000004d9658 is __libc_freeres_ptrs
        0x00007ffee3797120 - 0x00007ffee3797164 is .hash in system-supplied DSO at
0x7ffee3797000
        0x00007ffee3797168 - 0x00007ffee37971b8 is .gnu.hash in system-supplied DSO at
0x7ffee3797000
        0x00007ffee37971b8 - 0x00007ffee37972d8 is .dynsym in system-supplied DSO at
0x7ffee3797000
        0x00007ffee37972d8 - 0x00007ffee379734a is .dynstr in system-supplied DSO at
0x7ffee3797000
        0x00007ffee379734a - 0x00007ffee3797362 is .gnu.version in system-supplied DSO at
0x7ffee3797000
        0x00007ffee3797368 - 0x00007ffee37973a0 is .gnu.version_d in system-supplied DSO at
0x7ffee3797000
        0x00007ffee37973a0 - 0x00007ffee37974b0 is .dynamic in system-supplied DSO at
0x7ffee3797000
        0x00007ffee37974b0 - 0x00007ffee3797504 is .note in system-supplied DSO at
0x7ffee3797000
        0x00007ffee3797504 - 0x00007ffee3797538 is .eh_frame_hdr in system-supplied DSO at
0x7ffee3797000
        0x00007ffee3797538 - 0x00007ffee37975fc is .eh_frame in system-supplied DSO at
0x7ffee3797000
--Type <RET> for more, q to quit, c to continue without paging--
        0x00007ffee3797600 - 0x00007ffee3797915 is .text in system-supplied DSO at
0x7ffee3797000
        0x00007ffee3797915 - 0x00007ffee379798a is .altinstructions in system-supplied DSO at
0x7ffee3797000
        0x00007ffee379798a - 0x00007ffee37979ae is .altinstr_replacement in system-supplied DSO
at 0x7ffee3797000
```

Exercise A12 (A64, GDB)

Goal: Learn how to dump memory for post-processing, get the list of functions and module variables, load symbols, inspect arguments and local variables.

Patterns: Module Variable.

1. Load *App12.core.17894* dump file and *App12* executable from the A64/App12 directory:

```
~/ALCDA2/A64/App12$ gdb -c App12.core.17894 -se App12
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "aarch64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
App12: No such file or directory.

warning: Can't open file /home/opc/ALCDA2/App12/App12 during file-backed mapping note
processing
[New LWP 17895]
[New LWP 17896]
[New LWP 17905]
[New LWP 17906]
[New LWP 17907]
[New LWP 17894]
Core was generated by `./App12'.
#0  0x000000000041ae24 in ?? ()
[Current thread is 1 (LWP 17895)]
```

2. Set logging to a file in case of lengthy output from some commands:

```
(gdb) set logging file App12.log

(gdb) set logging enabled on
Copying output to App12.log.
Copying debug output to App12.log.

(gdb) set style enabled off
```

3. List all thread stack traces:

```
(gdb) thread apply all bt

Thread 6 (LWP 17894):
#0  0x000000000041ae24 in ?? ()
#1  0x00000000004d0020 in ?? ()
Backtrace stopped: previous frame inner to this frame (corrupt stack?)
```

```
Thread 5 (LWP 17907):
#0  0x000000000041ae24 in ?? ()
Backtrace stopped: previous frame identical to this frame (corrupt stack?)

Thread 4 (LWP 17906):
#0  0x000000000041a110 in ?? ()
Backtrace stopped: previous frame identical to this frame (corrupt stack?)

Thread 3 (LWP 17905):
#0  0x000000000041ae24 in ?? ()
Backtrace stopped: previous frame identical to this frame (corrupt stack?)

Thread 2 (LWP 17896):
#0  0x000000000041a110 in ?? ()
Backtrace stopped: previous frame identical to this frame (corrupt stack?)

Thread 1 (LWP 17895):
#0  0x000000000041ae24 in ?? ()
Backtrace stopped: previous frame identical to this frame (corrupt stack?)
```

4. *App12* is an executable with stripped-off debug symbols. Change the symbol file to *Symbols/App12,* which is the same executable as *App12* but with debug symbols included:

```
(gdb) symbol-file Symbols/App12
Reading symbols from Symbols/App12...
```

5. List all thread stack traces again:

```
(gdb) thread apply all bt

Thread 6 (LWP 17894):
#0  0x000000000041ae24 in nanosleep ()
#1  0x00000000004338b4 in sleep ()
#2  0x00000000004034e0 in main (argc=1, argv=0xfffff841c0a8) at main.cpp:92

Thread 5 (LWP 17907):
#0  0x000000000041ae24 in nanosleep ()
#1  0x00000000004338b4 in sleep ()
#2  0x00000000004033f0 in bar_five () at main.cpp:76
#3  0x0000000000403404 in foo_five () at main.cpp:76
#4  0x000000000040341c in thread_five (arg=0x0) at main.cpp:76
#5  0x00000000004130a4 in start_thread ()
#6  0x0000000000438760 in thread_start ()

Thread 4 (LWP 17906):
#0  0x000000000041a110 in __lll_lock_wait ()
#1  0x0000000000414ea4 in pthread_mutex_lock ()
#2  0x00000000004032a0 in procB () at main.cpp:49
#3  0x00000000004033a8 in bar_four () at main.cpp:75
#4  0x00000000004033bc in foo_four () at main.cpp:75
#5  0x00000000004033d4 in thread_four (arg=0x0) at main.cpp:75
#6  0x00000000004130a4 in start_thread ()
#7  0x0000000000438760 in thread_start ()

Thread 3 (LWP 17905):
#0  0x000000000041ae24 in nanosleep ()
#1  0x00000000004338b4 in sleep ()
#2  0x0000000000403364 in bar_three () at main.cpp:74
#3  0x0000000000403378 in foo_three () at main.cpp:74
```

```
#4  0x0000000000403390 in thread_three (arg=0x0) at main.cpp:74
#5  0x00000000004130a4 in start_thread ()
#6  0x0000000000438760 in thread_start ()

Thread 2 (LWP 17896):
#0  0x000000000041a110 in __lll_lock_wait ()
#1  0x0000000000414ea4 in pthread_mutex_lock ()
#2  0x000000000040325c in procA () at main.cpp:42
#3  0x000000000040331c in bar_two () at main.cpp:73
#4  0x0000000000403330 in foo_two () at main.cpp:73
#5  0x0000000000403348 in thread_two (arg=0x0) at main.cpp:73
#6  0x00000000004130a4 in start_thread ()
#7  0x0000000000438760 in thread_start ()

Thread 1 (LWP 17895):
#0  0x000000000041ae24 in nanosleep ()
#1  0x00000000004338b4 in sleep ()
#2  0x00000000004032d8 in bar_one () at main.cpp:72
#3  0x00000000004032ec in foo_one () at main.cpp:72
--Type <RET> for more, q to quit, c to continue without paging--
#4  0x0000000000403304 in thread_one (arg=0x0) at main.cpp:72
#5  0x00000000004130a4 in start_thread ()
#6  0x0000000000438760 in thread_start ()
```

6. Having full symbols allows us to check the ownership of the mutex faster (this is a new process with the same source code as in the previous A11 exercise):

```
(gdb) disassemble procB
Dump of assembler code for function _Z5procBv:
   0x0000000000403280 <+0>:     stp     x29, x30, [sp, #-16]!
   0x0000000000403284 <+4>:     mov     x29, sp
   0x0000000000403288 <+8>:     adrp    x0, 0x4d1000 <main_arena+2128>
   0x000000000040328c <+12>:    add     x0, x0, #0x608
   0x0000000000403290 <+16>:    bl      0x414dbc <pthread_mutex_lock>
   0x0000000000403294 <+20>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403298 <+24>:    add     x0, x0, #0x5d8
   0x000000000040329c <+28>:    bl      0x414dbc <pthread_mutex_lock>
   0x00000000004032a0 <+32>:    mov     w0, #0x1e                       // #30
   0x00000000004032a4 <+36>:    bl      0x4337a4 <sleep>
   0x00000000004032a8 <+40>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x00000000004032ac <+44>:    add     x0, x0, #0x5d8
   0x00000000004032b0 <+48>:    bl      0x414dbc <pthread_mutex_lock>
   0x00000000004032b4 <+52>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x00000000004032b8 <+56>:    add     x0, x0, #0x608
   0x00000000004032bc <+60>:    bl      0x414dbc <pthread_mutex_lock>
   0x00000000004032c0 <+64>:    ldp     x29, x30, [sp], #16
   0x00000000004032c4 <+68>:    ret
End of assembler dump.

(gdb) print (pthread_mutex_t *)0x4d15d8
$1 = (pthread_mutex_t *) 0x4d15d8 <mutexA>

(gdb) print *(pthread_mutex_t *)0x4d15d8
$2 = {__data = {__lock = 2, __count = 0, __owner = 17896, __nusers = 1, __kind = 0, __spins =
0, __list = {__prev = 0x0,
      __next = 0x0}}, __size = "\002\000\000\000\000\000\000\000\350E\000\000\001", '\000'
<repeats 34 times>, __align = 2}
```

```
(gdb) print mutexA
$2 = {__data = {__lock = 2, __count = 0, __owner = 17896, __nusers = 1, __kind = 0, __spins =
0, __list = {__prev = 0x0,
      __next = 0x0}}, __size = "\002\000\000\000\000\000\000\000\350E\000\000\001", '\000'
<repeats 34 times>, __align = 2}

(gdb) print mutexB
$3 = {__data = {__lock = 2, __count = 0, __owner = 17906, __nusers = 1, __kind = 0, __spins =
0, __list = {__prev = 0x0,
      __next = 0x0}}, __size = "\002\000\000\000\000\000\000\000\362E\000\000\001", '\000'
<repeats 34 times>, __align = 2}
```

7. Switch to thread #6 and its frame #2, and list arguments and locals:

```
(gdb) thread 6
[Switching to thread 6 (LWP 17894)]
#0  0x000000000041ae24 in nanosleep ()

(gdb) bt
#0  0x000000000041ae24 in nanosleep ()
#1  0x00000000004338b4 in sleep ()
#2  0x00000000004034e0 in main (argc=1, argv=0xfffff841c0a8) at main.cpp:92

(gdb) frame 2
#2  0x00000000004034e0 in main (argc=1, argv=0xfffff841c0a8) at main.cpp:92
92      main.cpp: No such file or directory.

(gdb) info args
argc = 1
argv = 0xfffff841c0a8

(gdb) info locals
No locals.
```

8. Examine the *argv* array:

```
(gdb) print argv[0]
$3 = 0xfffff841f6d7 "./App12"

(gdb) print *argv@10
$4 = {0xfffff841f6d7 "./App12", 0x0, 0xfffff841f6df "XDG_SESSION_ID=8954",
  0xfffff841f6f3 "HOSTNAME=instance-20211109-2004", 0xfffff841f713 "SELINUX_ROLE_REQUESTED=",
  0xfffff841f72b "TERM=xterm-256color", 0xfffff841f73f "SHELL=/bin/bash", 0xfffff841f74f
"HISTSIZE=1000",
  0xfffff841f75d "SSH_CLIENT=37.228.238.120 61014 22", 0xfffff841f780
"SELINUX_USE_CURRENT_RANGE="}
```

9. Dump the region 0x00400000 – 0x004e0000 to a binary file:

```
(gdb) dump memory mem.raw 0x00400000 0x004e0000
```

10. List all functions:

```
(gdb) info functions
All defined functions:

File main.cpp:
76:     void bar_five();
75:     void bar_four();
72:     void bar_one();
74:     void bar_three();
73:     void bar_two();
76:     void foo_five();
75:     void foo_four();
72:     void foo_one();
74:     void foo_three();
73:     void foo_two();
80:     int main(int, char const**);
28:     void procA();
46:     void procB();
23:     void procC();
76:     void *thread_five(void*);
75:     void *thread_four(void*);
72:     void *thread_one(void*);
74:     void *thread_three(void*);
73:     void *thread_two(void*);

Non-debugging symbols:
0x0000fffbfad70420  __kernel_clock_gettime
0x0000fffbfad704f8  __kernel_gettimeofday
0x0000fffbfad705a0  __kernel_clock_getres
0x0000fffbfad705b8  __kernel_rt_sigreturn
0x0000000000400250  _init
0x00000000004002c0  __pthread_mutex_lock_full
0x000000000040085c  __pthread_mutex_unlock_full
0x0000000000400cb8  __pthread_once_slow
0x0000000000400d9c  __pthread_mutex_cond_lock_full
0x0000000000401320  check_one_fd.part
0x00000000004013e0  new_heap
0x0000000000401568  arena_get2
0x0000000000401b28  arena_get_retry
0x0000000000401c14  sysmalloc
0x00000000004021d8  tcache_init.part
0x00000000004022d4  cancel_handler.part
0x00000000004022f8  put_locked_global.isra.3.part
0x000000000040231c  strip
0x00000000004023c0  read_int
0x000000000040242c  group_number
0x000000000040257c  _i18n_number_rewrite
0x00000000004027d0  _i18n_number_rewrite
0x0000000000402a24  search_list_add__
0x0000000000402a9c  nameserver_list_emplace__
0x0000000000402b48  is_trusted_path_normalize
--Type <RET> for more, q to quit, c to continue without paging-- q
Quit
```

11. We can also list all available types or specific types:

```
(gdb) info types pthread_mutex_t
All types matching regular expression "pthread_mutex_t":

File /usr/include/bits/pthreadtypes.h:
61:     pthread_mutex_t;
62:     pthread_mutex_t::__pthread_mutex_s;
75:     typedef pthread_mutex_t pthread_mutex_t;
```

12. List all variables:

```
(gdb) info variables
All defined variables:

File main.cpp:
21:     pthread_mutex_t mutexA;
21:     pthread_mutex_t mutexB;

Non-debugging symbols:
0x0000000000000000  _TLS_MODULE_BASE_
0x0000000000000000  __libc_resp
0x0000000000000000  __resp
0x0000000000000008  _nl_current_LC_CTYPE
0x0000000000000010  __libc_tsd_LOCALE
0x0000000000000018  _nl_current_LC_MONETARY
0x0000000000000020  _nl_current_LC_NUMERIC
0x0000000000000028  (anonymous namespace)::get_global()::global
0x0000000000000038  __libc_errno
0x0000000000000038  errno
0x0000000000000040  __libc_tsd_CTYPE_B
0x0000000000000048  __libc_tsd_CTYPE_TOUPPER
0x0000000000000050  __libc_tsd_CTYPE_TOLOWER
0x0000000000000058  thread_arena
0x0000000000000060  tcache
0x0000000000000068  tcache_shutting_down
0x0000000000000070  current
0x0000000000000078  __h_errno
0x0000000000000078  __libc_h_errno
0x0000000000000080  data
0x00000000004001d8  __rela_iplt_start
0x0000000000400250  __rela_iplt_end
0x00000000004933e0  _IO_stdin_used
0x00000000004933e8  __dso_handle
0x00000000004933f0  typeinfo name for __gnu_cxx::__concurrence_lock_error
0x0000000000493418  typeinfo name for __gnu_cxx::__concurrence_unlock_error
0x0000000000493498  typeinfo name for std::exception
0x00000000004934a8  typeinfo name for std::bad_exception
0x00000000004934c0  typeinfo name for __cxxabiv1::__forced_unwind
0x00000000004934e0  typeinfo name for __cxxabiv1::__foreign_exception
0x0000000000493548  typeinfo name for __cxxabiv1::__fundamental_type_info
0x0000000000493570  typeinfo name for void
0x0000000000493578  typeinfo name for void*
0x0000000000493580  typeinfo name for void const*
0x0000000000493588  typeinfo name for bool
0x0000000000493590  typeinfo name for bool*
0x0000000000493598  typeinfo name for bool const*
0x00000000004935a0  typeinfo name for wchar_t
0x00000000004935a8  typeinfo name for wchar_t*
0x00000000004935b0  typeinfo name for wchar_t const*
```

```
0x00000000004935b8  typeinfo name for char16_t
0x00000000004935c0  typeinfo name for char16_t*
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
```

13. List segment info (also **info files**):

```
(gdb) info target
Symbols from "/home/ubuntu/ALCDA2/A64/App12/Symbols/App12".
Local core dump file:
        `/home/ubuntu/ALCDA2/A64/App12/App12.core.17894', file type elf64-littleaarch64.
        0x0000000000400000 - 0x00000000004c0000 is load1
        0x00000000004c0000 - 0x00000000004e0000 is load2
        0x00000000004e0000 - 0x00000000004f0000 is load3
        0x000000003a6e0000 - 0x000000003a720000 is load4
        0x0000fffbf4000000 - 0x0000fffbf4030000 is load5
        0x0000fffbf4030000 - 0x0000fffbf8000000 is load6
        0x0000fffbf8510000 - 0x0000fffbf8520000 is load7
        0x0000fffbf8520000 - 0x0000fffbf8d20000 is load8
        0x0000fffbf8d20000 - 0x0000fffbf8d30000 is load9
        0x0000fffbf8d30000 - 0x0000fffbf9530000 is load10
        0x0000fffbf9530000 - 0x0000fffbf9540000 is load11
        0x0000fffbf9540000 - 0x0000fffbf9d40000 is load12
        0x0000fffbf9d40000 - 0x0000fffbf9d50000 is load13
        0x0000fffbf9d50000 - 0x0000fffbfa550000 is load14
        0x0000fffbfa550000 - 0x0000fffbfa560000 is load15
        0x0000fffbfa560000 - 0x0000fffbfad60000 is load16
        0x0000fffbfad70000 - 0x0000fffbfad80000 is load17
        0x0000fffff83f0000 - 0x0000fffff8420000 is load18
```

14. If we disassemble the *procA* function with the source code option, we don't get source code fragments because GDB can't find the location of *main.cpp*:

```
(gdb) disassemble /s procA
Dump of assembler code for function _Z5procAv:
main.cpp:
29      main.cpp: No such file or directory.
   0x0000000000403224 <+0>:     stp     x29, x30, [sp, #-16]!
   0x0000000000403228 <+4>:     mov     x29, sp

30      in main.cpp
31      in main.cpp
32      in main.cpp
   0x000000000040322c <+8>:     adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403230 <+12>:    add     x0, x0, #0x5d8
   0x0000000000403234 <+16>:    bl      0x414dbc <pthread_mutex_lock>

33      in main.cpp
   0x0000000000403238 <+20>:    bl      0x403200 <_Z5procCv>

34      in main.cpp
   0x000000000040323c <+24>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403240 <+28>:    add     x0, x0, #0x5d8
   0x0000000000403244 <+32>:    bl      0x416054 <pthread_mutex_unlock>

37      in main.cpp
38      in main.cpp
39      in main.cpp
40      in main.cpp
```

```
41      in main.cpp
   0x0000000000403248 <+36>:    mov     w0, #0x14                       // #20
   0x000000000040324c <+40>:    bl      0x4337a4 <sleep>

42      in main.cpp
   0x0000000000403250 <+44>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403254 <+48>:    add     x0, x0, #0x608
   0x0000000000403258 <+52>:    bl      0x414dbc <pthread_mutex_lock>

--Type <RET> for more, q to quit, c to continue without paging--
43      in main.cpp
   0x000000000040325c <+56>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403260 <+60>:    add     x0, x0, #0x608
   0x0000000000403264 <+64>:    bl      0x416054 <pthread_mutex_unlock>
   0x0000000000403268 <+68>:    b       0x403278 <_Z5procAv+84>

36      in main.cpp
   0x000000000040326c <+72>:    bl      0x4039b4 <__cxa_begin_catch>
   0x0000000000403270 <+76>:    bl      0x403a58 <__cxa_end_catch>
   0x0000000000403274 <+80>:    b       0x403248 <_Z5procAv+36>

44      in main.cpp
   0x0000000000403278 <+84>:    ldp     x29, x30, [sp], #16
   0x000000000040327c <+88>:    ret
End of assembler dump.
```

Note: We can specify the source code directory:

```
(gdb) directory Source/
Source directories searched: /home/ubuntu/ALCDA2/A64/App12/Source: $cdir:$cwd

(gdb) disassemble /s procA
Dump of assembler code for function _Z5procAv:
main.cpp:
29      {
   0x0000000000403224 <+0>:     stp     x29, x30, [sp, #-16]!
   0x0000000000403228 <+4>:     mov     x29, sp

30          try
31          {
32              pthread_mutex_lock(&mutexA);
   0x000000000040322c <+8>:     adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403230 <+12>:    add     x0, x0, #0x5d8
   0x0000000000403234 <+16>:    bl      0x414dbc <pthread_mutex_lock>

33              procC();
   0x0000000000403238 <+20>:    bl      0x403200 <_Z5procCv>

34              pthread_mutex_unlock(&mutexA);
   0x000000000040323c <+24>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403240 <+28>:    add     x0, x0, #0x5d8
   0x0000000000403244 <+32>:    bl      0x416054 <pthread_mutex_unlock>

37          {
38
39          }
40
41          sleep(20);
   0x0000000000403248 <+36>:    mov     w0, #0x14                       // #20
   0x000000000040324c <+40>:    bl      0x4337a4 <sleep>
```

```
42          pthread_mutex_lock(&mutexB);
   0x0000000000403250 <+44>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403254 <+48>:    add     x0, x0, #0x608
   0x0000000000403258 <+52>:    bl      0x414dbc <pthread_mutex_lock>
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
(gdb) disassemble /s procA
Dump of assembler code for function _Z5procAv:
main.cpp:
29      {
   0x0000000000403224 <+0>:     stp     x29, x30, [sp, #-16]!
   0x0000000000403228 <+4>:     mov     x29, sp

30          try
31          {
32              pthread_mutex_lock(&mutexA);
   0x000000000040322c <+8>:     adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403230 <+12>:    add     x0, x0, #0x5d8
   0x0000000000403234 <+16>:    bl      0x414dbc <pthread_mutex_lock>

33              procC();
   0x0000000000403238 <+20>:    bl      0x403200 <_Z5procCv>

34              pthread_mutex_unlock(&mutexA);
   0x000000000040323c <+24>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403240 <+28>:    add     x0, x0, #0x5d8
   0x0000000000403244 <+32>:    bl      0x416054 <pthread_mutex_unlock>

37          {
38
39          }
40
41          sleep(20);
   0x0000000000403248 <+36>:    mov     w0, #0x14                       // #20
   0x000000000040324c <+40>:    bl      0x4337a4 <sleep>

42          pthread_mutex_lock(&mutexB);
   0x0000000000403250 <+44>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403254 <+48>:    add     x0, x0, #0x608
   0x0000000000403258 <+52>:    bl      0x414dbc <pthread_mutex_lock>
--Type <RET> for more, q to quit, c to continue without paging--

43          pthread_mutex_unlock(&mutexB);
   0x000000000040325c <+56>:    adrp    x0, 0x4d1000 <main_arena+2128>
   0x0000000000403260 <+60>:    add     x0, x0, #0x608
   0x0000000000403264 <+64>:    bl      0x416054 <pthread_mutex_unlock>
   0x0000000000403268 <+68>:    b       0x403278 <_Z5procAv+84>

36          catch(...)
   0x000000000040326c <+72>:    bl      0x4039b4 <__cxa_begin_catch>
   0x0000000000403270 <+76>:    bl      0x403a58 <__cxa_end_catch>
   0x0000000000403274 <+80>:    b       0x403248 <_Z5procAv+36>

44      }
   0x0000000000403278 <+84>:    ldp     x29, x30, [sp], #16
   0x000000000040327c <+88>:    ret
End of assembler dump.
```

Exercise A12 (A64, WinDbg Preview)

Goal: Learn how to dump memory for post-processing, get the list of functions and module variables, load symbols, inspect arguments and local variables.

Patterns: Module Variable.

1. We have a core dump of the *App12* executable that was stripped of debugging information to run in production, and we also have an original executable in the App12\Symbols folder.

2. Launch WinDbg Preview.

3. Load *App12.core.17894* dump file from the A64\App12 folder:

```
Microsoft (R) Windows Debugger Version 10.0.25111.1000 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.


Loading Dump File [C:\ALCDA2\A64\App12\App12.core.17894]
64-bit machine not using 64-bit API

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
Symbol search path is: srv*
Executable search path is:
Generic Unix Version 0 UP Free ARM 64-bit (AArch64)
Machine Name:
System Uptime: not available
Process Uptime: not available
..
*** WARNING: Unable to verify timestamp for App12
App12+0x1ae24:
00000000`0041ae24 d4000001 svc         #0
```

4. Set logging to a file in case of lengthy output from some commands:

```
0:000> .logopen C:\ALCDA2\A64\App12\App12.log
Opened log file 'C:\ALCDA2\A64\App12\App12.log'
```

5. Specify the folder for the executable and symbol paths, and reload symbols:

```
0:000> .exepath+ C:\ALCDA2\A64\App12\Symbols
Executable image search path is: C:\ALCDA2\A64\App12\Symbols
Expanded Executable image search path is: c:\alcda2\a64\app12\symbols

************* Path validation summary **************
Response                         Time (ms)     Location
OK                                             C:\ALCDA2\A64\App12\Symbols
```

```
0:000> .sympath+ C:\ALCDA2\A64\App12\Symbols
Symbol search path is: srv*;C:\ALCDA2\A64\App12\Symbols
Expanded Symbol search path is:
cache*;SRV*https://msdl.microsoft.com/download/symbols;c:\alcda2\a64\app12\symbols

************* Path validation summary **************
Response                         Time (ms)     Location
Deferred                                       srv*
OK                                             C:\ALCDA2\A64\App12\Symbols
*** WARNING: Unable to verify timestamp for App12
```

```
0:000> .reload
..
*** WARNING: Unable to verify timestamp for App12

************* Symbol Loading Error Summary **************
Module name            Error
App12                  The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.
```

Note: We ignore warnings and errors as they are not relevant for now.

6. List all thread stack traces (we see that source code references are now included):

```
0:000> ~*k

Unable to get thread data for thread 0
.  0  Id: 45e6.45e7 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffb`fad5e5e0 00000000`004338b4     App12!_libc_nanosleep+0x24
01 0000fffb`fad5e620 00000000`004032d8     App12!sleep+0x110
02 0000fffb`fad5e810 00000000`004032ec     App12!bar_one+0x10 [/home/opc/ALCDA2/App12\main.cpp @ 72]
03 0000fffb`fad5e820 00000000`00403304     App12!foo_one+0xc [/home/opc/ALCDA2/App12\main.cpp @ 72]
04 0000fffb`fad5e830 00000000`004130a4     App12!thread_one+0x10 [/home/opc/ALCDA2/App12\main.cpp @ 72]
05 0000fffb`fad5e850 00000000`00438760     App12!start_thread+0xb4
06 0000fffb`fad5e980 ffffffff`ffffffff     App12!thread_start+0x30
07 0000fffb`fad5e980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 1
   1  Id: 45e6.45e8 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffb`fa54e7d0 00000000`00414ea4     App12!_lll_lock_wait+0x3c
01 0000fffb`fa54e7d0 00000000`0040325c     App12!_pthread_mutex_lock+0xe8
02 0000fffb`fa54e800 00000000`0040331c     App12!procA+0x38 [/home/opc/ALCDA2/App12\main.cpp @ 43]
03 0000fffb`fa54e810 00000000`00403330     App12!bar_two+0xc [/home/opc/ALCDA2/App12\main.cpp @ 73]
04 0000fffb`fa54e820 00000000`00403348     App12!foo_two+0xc [/home/opc/ALCDA2/App12\main.cpp @ 73]
05 0000fffb`fa54e830 00000000`004130a4     App12!thread_two+0x10 [/home/opc/ALCDA2/App12\main.cpp @ 73]
06 0000fffb`fa54e850 00000000`00438760     App12!start_thread+0xb4
07 0000fffb`fa54e980 ffffffff`ffffffff     App12!thread_start+0x30
08 0000fffb`fa54e980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 2
   2  Id: 45e6.45f1 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffb`f9d3e5e0 00000000`004338b4     App12!_libc_nanosleep+0x24
01 0000fffb`f9d3e620 00000000`00403364     App12!sleep+0x110
02 0000fffb`f9d3e810 00000000`00403378     App12!bar_three+0x10 [/home/opc/ALCDA2/App12\main.cpp @ 74]
03 0000fffb`f9d3e820 00000000`00403390     App12!foo_three+0xc [/home/opc/ALCDA2/App12\main.cpp @ 74]
04 0000fffb`f9d3e830 00000000`004130a4     App12!thread_three+0x10 [/home/opc/ALCDA2/App12\main.cpp @ 74]
05 0000fffb`f9d3e850 00000000`00438760     App12!start_thread+0xb4
```

```
06 0000fffb`f9d3e980 ffffffff`ffffffff     App12!thread_start+0x30
07 0000fffb`f9d3e980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 3
   3  Id: 45e6.45f2 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffb`f952e7d0 00000000`00414ea4     App12!_lll_lock_wait+0x3c
01 0000fffb`f952e7d0 00000000`004032a0     App12!_pthread_mutex_lock+0xe8
02 0000fffb`f952e800 00000000`004033a8     App12!procB+0x20 [/home/opc/ALCDA2/App12\main.cpp @ 50]
03 0000fffb`f952e810 00000000`004033bc     App12!bar_four+0xc [/home/opc/ALCDA2/App12\main.cpp @ 75]
04 0000fffb`f952e820 00000000`004033d4     App12!foo_four+0xc [/home/opc/ALCDA2/App12\main.cpp @ 75]
05 0000fffb`f952e830 00000000`004130a4     App12!thread_four+0x10 [/home/opc/ALCDA2/App12\main.cpp @ 75]
06 0000fffb`f952e850 00000000`00438760     App12!start_thread+0xb4
07 0000fffb`f952e980 ffffffff`ffffffff     App12!thread_start+0x30
08 0000fffb`f952e980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 4
   4  Id: 45e6.45f3 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000fffb`f8d1e5e0 00000000`004338b4     App12!_libc_nanosleep+0x24
01 0000fffb`f8d1e620 00000000`004033f0     App12!sleep+0x110
02 0000fffb`f8d1e810 00000000`00403404     App12!bar_five+0x10 [/home/opc/ALCDA2/App12\main.cpp @ 76]
03 0000fffb`f8d1e820 00000000`0040341c     App12!foo_five+0xc [/home/opc/ALCDA2/App12\main.cpp @ 76]
04 0000fffb`f8d1e830 00000000`004130a4     App12!thread_five+0x10 [/home/opc/ALCDA2/App12\main.cpp @ 76]
05 0000fffb`f8d1e850 00000000`00438760     App12!start_thread+0xb4
06 0000fffb`f8d1e980 ffffffff`ffffffff     App12!thread_start+0x30
07 0000fffb`f8d1e980 00000000`00000000     0xffffffff`ffffffff

Unable to get thread data for thread 5
   5  Id: 45e6.45e6 Suspend: 0 Teb: 00000000`00000000 Unfrozen
 # Child-SP          RetAddr               Call Site
00 0000ffff`f841bcc0 00000000`004338b4     App12!_libc_nanosleep+0x24
01 0000ffff`f841bd00 00000000`004034e0     App12!sleep+0x110
02 0000ffff`f841bef0 00000000`0041d0bc     App12!main+0xb8 [/home/opc/ALCDA2/App12\main.cpp @ 93]
03 0000ffff`f841bf40 00000000`004030a8     App12!_libc_start_main+0x304
04 0000ffff`f841c0a0 00000000`00000000     App12!start+0x4c
```

7. If we want to omit them we use this command variant:

```
0:000> kL
 # Child-SP          RetAddr               Call Site
00 0000fffb`fad5e5e0 00000000`004338b4     App12!_libc_nanosleep+0x24
01 0000fffb`fad5e620 00000000`004032d8     App12!sleep+0x110
02 0000fffb`fad5e810 00000000`004032ec     App12!bar_one+0x10
03 0000fffb`fad5e820 00000000`00403304     App12!foo_one+0xc
04 0000fffb`fad5e830 00000000`004130a4     App12!thread_one+0x10
05 0000fffb`fad5e850 00000000`00438760     App12!start_thread+0xb4
06 0000fffb`fad5e980 ffffffff`ffffffff     App12!thread_start+0x30
07 0000fffb`fad5e980 00000000`00000000     0xffffffff`ffffffff
```

8. If we want to include parameters in the stack trace, we use another command variant:

```
0:000> ~5kPL
 # Child-SP          RetAddr               Call Site
00 0000ffff`f841bcc0 00000000`004338b4     App12!_libc_nanosleep+0x24
01 0000ffff`f841bd00 00000000`004034e0     App12!sleep+0x110
02 0000ffff`f841bef0 00000000`0041d0bc     App12!main(
                    int argc = 0n1,
                    char ** argv = 0x0000ffff`f841c0a8)+0xb8
03 0000ffff`f841bf40 00000000`004030a8     App12!_libc_start_main+0x304
04 0000ffff`f841c0a0 00000000`00000000     App12!start+0x4c
```

9. Switch to thread #5 and its frame #2, and list arguments and locals:

```
0:000> ~5s
App12!_libc_nanosleep+0x24:
00000000`0041ae24 d4000001 svc         #0

0:005> .frame /c /r 2
02 0000ffff`f841bef0 00000000`0041d0bc     App12!main+0xb8 [/home/opc/ALCDA2/App12\main.cpp @
93]
 x0=0000ffff00000000   x1=0000fffff841bd40   x2=000000003a6e0000   x3=000000003a6e0108
 x4=0000fffff841bcb0   x5=000000003a6e06f0   x6=00000000ffffffbb   x7=0000000000000000
 x8=0000000000000065   x9=00000000004d07b0  x10=000000003a6e2670  x11=000000000003d990
x12=0000000000000002  x13=0000fffff841bd70  x14=0000000000000008  x15=0000000000000000
x16=0000000000000000  x17=0000000000431dd0  x18=0000000000000110  x19=0000000000400250
x20=00000000004d0020  x21=0000000000400250  x22=0000000000000018  x23=00000000004e8000
x24=00000000004e8000  x25=0000000000000000  x26=000000000041d4f8  x27=000000000041d5b0
x28=0000000000000000   fp=0000fffff841bef0   lr=00000000004034e0   sp=0000fffff841bef0
 pc=00000000004034e0  psr=80001000 N--- EL0
App12!main+0xb8:
00000000`004034e0 52800000 mov         w0,#0

0:005> dv /i /V
prv param  0000ffff`f841bf0c @fp+0x001c                  argc = 0n1
prv param  0000ffff`f841bf00 @fp+0x0010                  argv = 0x0000ffff`f841c0a8
```

10. Examine the *argv* array:

```
0:005> dpa 0x0000ffff`f841c0a8 L10
0000ffff`f841c0a8  0000ffff`f841f6d7 "./App12"
0000ffff`f841c0b0  00000000`00000000
0000ffff`f841c0b8  0000ffff`f841f6df "XDG_SESSION_ID=8954"
0000ffff`f841c0c0  0000ffff`f841f6f3 "HOSTNAME=instance-20211109-2004"
0000ffff`f841c0c8  0000ffff`f841f713 "SELINUX_ROLE_REQUESTED="
0000ffff`f841c0d0  0000ffff`f841f72b "TERM=xterm-256color"
0000ffff`f841c0d8  0000ffff`f841f73f "SHELL=/bin/bash"
0000ffff`f841c0e0  0000ffff`f841f74f "HISTSIZE=1000"
0000ffff`f841c0e8  0000ffff`f841f75d "SSH_CLIENT=37.228.238.120 61014 22"
0000ffff`f841c0f0  0000ffff`f841f780 "SELINUX_USE_CURRENT_RANGE="
0000ffff`f841c0f8  0000ffff`f841f79b "SSH_TTY=/dev/pts/0"
0000ffff`f841c100  0000ffff`f841f7ae "USER=opc"
0000ffff`f841c108  0000ffff`f841f7b7 "LS_COLORS=rs=0:di=38;5;27:ln=38;5;51:mh=44;38;5;15:pi=4"
0000ffff`f841c110  0000ffff`f841fe6f "MAIL=/var/spool/mail/opc"
0000ffff`f841c118  0000ffff`f841fe88 "PATH=/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:"
0000ffff`f841c120  0000ffff`f841fee2 "PWD=/home/opc/ALCDA2/App12"
```

11. We can also specify the path to the source code (source code view should appear after resetting the context via the **.cxr** command):

```
0:005> .srcpath+ C:\ALCDA2\A64\App12\Source
Source search path is: C:\ALCDA2\A64\App12\Source

************* Path validation summary **************
Response                         Time (ms)     Location
OK                                             C:\ALCDA2\A64\App12\Source

0:005> .cxr
```

main.cpp

```
67 foo_##num();\
68 \
69 return 0;\
70 }
71
72 THREAD_DECLARE(one,sleep(-1))
73 THREAD_DECLARE(two,procA())
74 THREAD_DECLARE(three,sleep(-1))
75 THREAD_DECLARE(four,procB())
76 THREAD_DECLARE(five,sleep(-1))
77
78 #define THREAD_CREATE(num) {pthread_t threadID_##num; pthrea
79
80 int main(int argc, const char * argv[])
81 {
82     pthread_mutex_init(&mutexA, NULL);
83     pthread_mutex_init(&mutexB, NULL);
84
85     THREAD_CREATE(one)
86     THREAD_CREATE(two)
87     sleep(10);
88     THREAD_CREATE(three)
89     THREAD_CREATE(four)
90     THREAD_CREATE(five)
91
92     sleep(-1);
93     return 0;
94 }
95
```

12. Dump the contents of the loaded *App12* module to a binary file:

```
0:005> lm
start             end                 module name
00000000`00400000 00000000`004e0000   App12    T (service symbols: DWARF Private Symbols)
c:\alcda2\a64\app12\symbols\App12

0:005> .writemem C:\ALCDA2\A64\App12\mem.raw 00000000`00400000 00000000`004e0000
Writing e0001
bytes.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................
```

13. List all available symbols:

```
0:005> x App12!*
00000000`004d15d8 App12!mutexA = pthread_mutex_t
00000000`004d1608 App12!mutexB = pthread_mutex_t
00000000`00000000 App12!__prev = <Memory access error>
00000000`00403200 App12!procC (int, char **)
00000000`00403224 App12!procA (int, char **)
00000000`00403280 App12!procB (int, char **)
00000000`004032c8 App12!bar_one (int, char **)
00000000`004032e0 App12!foo_one (int, char **)
```

```
00000000`004032f4 App12!thread_one (int, char **)
00000000`00403310 App12!bar_two (int, char **)
00000000`00403324 App12!foo_two (int, char **)
00000000`00403338 App12!thread_two (int, char **)
00000000`00403354 App12!bar_three (int, char **)
00000000`0040336c App12!foo_three (int, char **)
00000000`00403380 App12!thread_three (int, char **)
00000000`0040339c App12!bar_four (int, char **)
00000000`004033b0 App12!foo_four (int, char **)
00000000`004033c4 App12!thread_four (int, char **)
00000000`004033e0 App12!bar_five (int, char **)
00000000`004033f8 App12!foo_five (int, char **)
00000000`0040340c App12!thread_five (int, char **)
00000000`00403428 App12!main (int, char **)
00000000`00000800 App12!/usr/lib/gcc/aarch64-redhat-linux/4.8.5/../../../../lib64/crt1.o = <no
type information>
00000000`00400190 App12!$d = <no type information>
00000000`0040305c App12!$x = <no type information>
00000000`004933e0 App12!$d = <no type information>
00000000`00000800 App12!/usr/lib/gcc/aarch64-redhat-linux/4.8.5/../../../../lib64/crti.o = <no
type information>
00000000`004030ac App12!$x = <no type information>
00000000`004030ac App12!call_weak_fn = <no type information>
00000000`00400250 App12!$x = <no type information>
00000000`004933c8 App12!$x = <no type information>
00000000`00000800 App12!/usr/lib/gcc/aarch64-redhat-linux/4.8.5/../../../../lib64/crtn.o = <no
type information>
00000000`0040025c App12!$x = <no type information>
00000000`004933d0 App12!$x = <no type information>
00000000`00000800 App12!libpthread.o = <no type information>
00000000`00411860 App12!$x = <no type information>
00000000`00411860 App12!sighandler_setxid = <no type information>
00000000`0041195c App12!sigcancel_handler = <no type information>
00000000`00494d00 App12!$d = <no type information>
00000000`00494d18 App12!nptl_version = <no type information>
00000000`004e34c8 App12!_nptl_initial_report_events = <no type information>
00000000`004e34c8 App12!$d = <no type information>
00000000`004afab4 App12!$d = <no type information>
00000000`004e34d0 App12!$d = <no type information>
00000000`00411d40 App12!$x = <no type information>
00000000`004afb74 App12!$d = <no type information>
00000000`00411d48 App12!$x = <no type information>
[...]
00000000`0042bef8 App12!free = <no type information>
00000000`00468fd0 App12!_towctrans = <no type information>
00000000`004051b0 App12!ZN9__gnu_cxx27__verbose_terminate_handlerEv = <no type information>
00000000`004671d0 App12!nl_get_era_entry = <no type information>
00000000`00404de8 App12!ZN10__cxxabiv117__class_type_infoD0Ev = <no type information>
00000000`00412f80 App12!_free_tcb = <no type information>
00000000`00438d74 App12!_res_nclose = <no type information>
00000000`0041d820 App12!sigprocmask = <no type information>
00000000`00425dc0 App12!IO_old_init = <no type information>
00000000`004ace60 App12!IO_file_jumps_mmap = <no type information>
00000000`0047fc38 App12!_libc_register_dlfcn_hook = <no type information>
00000000`0048da10 App12!getsockname = <no type information>
00000000`0048e060 App12!dl_map_object_deps = <no type information>
00000000`004a65d8 App12!nl_C_LC_IDENTIFICATION = <no type information>
00000000`004e9af8 App12!dl_ns = <no type information>
00000000`004464ec App12!nl_load_locale_from_archive = <no type information>
00000000`00468f34 App12!wctrans = <no type information>
```

```
00000000`0041f0c0 App12!fopen64 = <no type information>
```

14. List all available types and functions/variables:

```
0:005> dt App12!*
          App12!pthread_t
          App12!__pthread_internal_list
          App12!__pthread_list_t
          App12!pthread_mutex_t
          App12!__pthread_mutex_s
00000000004d15d8  App12!mutexA
00000000004d1608  App12!mutexB
          App12!__prev
0000000000403200  App12!procC
0000000000403224  App12!procA
0000000000403280  App12!procB
00000000004032c8  App12!bar_one
00000000004032e0  App12!foo_one
00000000004032f4  App12!thread_one
0000000000403310  App12!bar_two
0000000000403324  App12!foo_two
0000000000403338  App12!thread_two
0000000000403354  App12!bar_three
000000000040336c  App12!foo_three
0000000000403380  App12!thread_three
000000000040339c  App12!bar_four
00000000004033b0  App12!foo_four
00000000004033c4  App12!thread_four
00000000004033e0  App12!bar_five
00000000004033f8  App12!foo_five
000000000040340c  App12!thread_five
0000000000403428  App12!main
```

15. We notice mutex waits on backtraces, and since mutexes have owners (thread IDs), we can check their ownership instead of disassembly to detect a deadlock as we did in the previous exercise A11 (we can use either **dx** or **dt** commands):

```
0:005> x App12!mutex*
00000000`004d15d8 App12!mutexA = pthread_mutex_t
00000000`004d1608 App12!mutexB = pthread_mutex_t

0:005> dx mutexA
mutexA                 [Type: pthread_mutex_t]
    [+0x000] __data           [Type: __pthread_mutex_s]
    [+0x000] __size           : "???" [Type: char [48]]
    [+0x000] __align          : 2 [Type: long int]

0:005> dx mutexA.__data
mutexA.__data                 [Type: __pthread_mutex_s]
    [+0x000] __lock           : 2 [Type: int]
    [+0x004] __count          : 0x0 [Type: unsigned int]
    [+0x008] __owner          : 17896 [Type: int]
    [+0x00c] __nusers         : 0x1 [Type: unsigned int]
    [+0x010] __kind           : 0 [Type: int]
    [+0x014] __spins          : 0 [Type: int]
    [+0x018] __list           [Type: __pthread_list_t]

0:005> ? 0n17896
Evaluate expression: 17896 = 00000000`000045e8
```

```
0:005> ~~[45e8]k
 # Child-SP          RetAddr               Call Site
00 0000fffb`fa54e7d0 00000000`00414ea4     App12!_lll_lock_wait+0x3c
01 0000fffb`fa54e7d0 00000000`0040325c     App12!_pthread_mutex_lock+0xe8
02 0000fffb`fa54e800 00000000`0040331c     App12!procA+0x38
03 0000fffb`fa54e810 00000000`00403330     App12!bar_two+0xc
04 0000fffb`fa54e820 00000000`00403348     App12!foo_two+0xc
05 0000fffb`fa54e830 00000000`004130a4     App12!thread_two+0x10
06 0000fffb`fa54e850 00000000`00438760     App12!start_thread+0xb4
07 0000fffb`fa54e980 ffffffff`ffffffff     App12!thread_start+0x30
08 0000fffb`fa54e980 00000000`00000000     0xffffffff`ffffffff

0:005> dt pthread_mutex_t 00000000`004d15d8
App12!pthread_mutex_t
   +0x000 __data           : __pthread_mutex_s
   +0x000 __size           : [48]  "???"
   +0x000 __align          : 0n2

0:005> dt -r pthread_mutex_t 00000000`004d15d8
App12!pthread_mutex_t
   +0x000 __data           : __pthread_mutex_s
      +0x000 __lock           : 0n2
      +0x004 __count          : 0
      +0x008 __owner          : 0n17896
      +0x00c __nusers         : 1
      +0x010 __kind           : 0n0
      +0x014 __spins          : 0n0
      +0x018 __list           : __pthread_internal_list
         +0x000 __prev           : (null)
         +0x008 __next           : (null)
   +0x000 __size           : [48]  "???"
   +0x000 __align          : 0n2

0:005> ~~[0n17896]kL
 # Child-SP          RetAddr               Call Site
00 0000fffb`fa54e7d0 00000000`00414ea4     App12!_lll_lock_wait+0x3c
01 0000fffb`fa54e7d0 00000000`0040325c     App12!_pthread_mutex_lock+0xe8
02 0000fffb`fa54e800 00000000`0040331c     App12!procA+0x38
03 0000fffb`fa54e810 00000000`00403330     App12!bar_two+0xc
04 0000fffb`fa54e820 00000000`00403348     App12!foo_two+0xc
05 0000fffb`fa54e830 00000000`004130a4     App12!thread_two+0x10
06 0000fffb`fa54e850 00000000`00438760     App12!start_thread+0xb4
07 0000fffb`fa54e980 ffffffff`ffffffff     App12!thread_start+0x30
08 0000fffb`fa54e980 00000000`00000000     0xffffffff`ffffffff
```

16. We close logging before exiting WinDbg Preview:

```
0:005> .logclose
Closing open log file 'C:\ALCDA2\A64\App12\App12.log
```

Kernel Core Dumps

Exercises K1 – K5

Exercise K1

- **Goal:** Learn how to navigate a normal kernel dump
- **Patterns:** Manual Dump (Kernel); Stack Trace Collection
- \ALCDA-Dumps\Exercise-K1-x64-GDB.pdf

Exercise K1 (x64, GDB)

Goal: Learn how to navigate a normal kernel dump.

Patterns: Manual Dump (Kernel); Stack Trace Collection.

1. Install crash analysis tool:

```
~/ALCDA2/x64$ sudo apt install crash
```

2. Load a core dump *dump.202112280237* from the x64/K1 directory and the matching *vmlinux-5.10.0-10-amd64* file from the x64/KSym directory:

```
~/ALCDA2/x64/K1$ crash dump.202112280237 ../KSym/vmlinux-5.10.0-10-amd64

crash 8.0.0++
Copyright (C) 2002-2021  Red Hat, Inc.
Copyright (C) 2004, 2005, 2006, 2010  IBM Corporation
Copyright (C) 1999-2006  Hewlett-Packard Co
Copyright (C) 2005, 2006, 2011, 2012  Fujitsu Limited
Copyright (C) 2006, 2007  VA Linux Systems Japan K.K.
Copyright (C) 2005, 2011, 2020-2021  NEC Corporation
Copyright (C) 1999, 2002, 2007  Silicon Graphics, Inc.
Copyright (C) 1999, 2000, 2001, 2002  Mission Critical Linux, Inc.
Copyright (C) 2015, 2021  VMware, Inc.
This program is free software, covered by the GNU General Public License,
and you are welcome to change it and/or distribute copies of it under
certain conditions.  Enter "help copying" to see the conditions.
This program has absolutely no warranty.  Enter "help warranty" for details.

GNU gdb (GDB) 10.2
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-pc-linux-gnu".
Type "show configuration" for configuration details.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...

      KERNEL: ../KSym/vmlinux-5.10.0-10-amd64  [TAINTED]
    DUMPFILE: dump.202112280237  [PARTIAL DUMP]
        CPUS: 4
        DATE: Tue Dec 28 02:36:54 GMT 2021
      UPTIME: 00:04:36
LOAD AVERAGE: 0.04, 0.17, 0.09
       TASKS: 473
    NODENAME: coredump
     RELEASE: 5.10.0-10-amd64
     VERSION: #1 SMP Debian 5.10.84-1 (2021-12-08)
     MACHINE: x86_64  (1991 Mhz)
      MEMORY: 4 GB
```

```
       PANIC: "Kernel panic - not syncing: sysrq triggered crash"
         PID: 2135
     COMMAND: "tee"
        TASK: ffff9a2c45920000  [THREAD_INFO: ffff9a2c45920000]
         CPU: 3
       STATE: TASK_RUNNING (PANIC)

crash>
```

Note: If the *crash* tool fails to launch, it means that it is not up to date with the *vmlinux* kernel, and the latest version of the tool is required. You need to install it from the source as we did for WSL2 Debian:

```
$ git clone https://github.com/crash-utility/crash.git
$ sudo apt install bison
$ cd crash
$ make
$ sudo make install
```

3. We can see the current thread from the process ID that led to the crash:

```
crash> bt
PID: 2135   TASK: ffff9a2c45920000  CPU: 3   COMMAND: "tee"
 #0 [ffffa77fc2837cd0] machine_kexec at ffffffff8fc6436b
 #1 [ffffa77fc2837d28] __crash_kexec at ffffffff8fd3aaad
 #2 [ffffa77fc2837df0] panic at ffffffff9047f24d
 #3 [ffffa77fc2837e70] sysrq_handle_crash at ffffffff901ca426
 #4 [ffffa77fc2837e78] __handle_sysrq.cold at ffffffff904a44c3
 #5 [ffffa77fc2837ea8] write_sysrq_trigger at ffffffff901cad34
 #6 [ffffa77fc2837eb8] proc_reg_write at ffffffff8ff64501
 #7 [ffffa77fc2837ed0] vfs_write at ffffffff8fec1f40
 #8 [ffffa77fc2837f08] ksys_write at ffffffff8fec23cf
 #9 [ffffa77fc2837f40] do_syscall_64 at ffffffff904b3883
#10 [ffffa77fc2837f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f1ddc1f0f33  RSP: 00007ffea91896f8  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 0000000000000002  RCX: 00007f1ddc1f0f33
    RDX: 0000000000000002  RSI: 00007ffea9189810  RDI: 0000000000000003
    RBP: 00007ffea9189810   R8: 0000000000000000   R9: 0000000000000001
    R10: fffffffffffff286  R11: 0000000000000246  R12: 0000000000000002
    R13: 000055be3051d4a0  R14: 0000000000000002  R15: 00007f1ddc2c18a0
    ORIG_RAX: 0000000000000001  CS: 0033  SS: 002b
```

Note: User space addresses are not available in the kernel dump.

```
crash> sym 00007f1ddc1f0f33
sym: invalid address: 00007f1ddc1f0f33

crash> sym ffffffff9047f24d
ffffffff9047f24d (T) panic+273 debian/build/build_amd64_none_amd64/arch/x86/include/asm/smp.h:
62
```

4. The tool has built-in help:

```
crash> help

*               extend          log             rd              task
alias           files           mach            repeat          timer
ascii           foreach         mod             runq            tree
bpf             fuser           mount           search          union
bt              gdb             net             set             vm
btop            help            p               sig             vtop
dev             ipcs            ps              struct          waitq
dis             irq             pte             swap            whatis
eval            kmem            ptob            sym             wr
exit            list            ptov            sys             q

crash version: 8.0.0++   gdb version: 10.2
For help on any command above, enter "help <command>".
For help on input options, enter "help input".
For help on output options, enter "help output".
```

5. Print kernel message buffer before the crash (**dmesg** or **log**) with human-readable timestamps:

```
crash> dmesg -T
[Tue Dec 28 02:32:18 GMT 2021] Linux version 5.10.0-10-amd64 (debian-kernel@lists.debian.org) (gcc-10 (Debian 10.2.1-
6) 10.2.1 20210110, GNU ld (GNU Binutils for Debian) 2.35.2) #1 SMP Debian 5.10.84-1 (2021-12-08)
[Tue Dec 28 02:32:18 GMT 2021] Command line: BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64 root=UUID=9cc5ee1e-5533-4a0b-
a88f-903bf52d812d ro quiet crashkernel=384M-:128M
[Tue Dec 28 02:32:18 GMT 2021] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[Tue Dec 28 02:32:18 GMT 2021] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[Tue Dec 28 02:32:18 GMT 2021] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[Tue Dec 28 02:32:18 GMT 2021] x86/fpu: xstate_offset[2]:  576, xstate_sizes[2]:  256
[Tue Dec 28 02:32:18 GMT 2021] x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using 'standard'
format.
[Tue Dec 28 02:32:18 GMT 2021] BIOS-provided physical RAM map:
[Tue Dec 28 02:32:18 GMT 2021] BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
[Tue Dec 28 02:32:18 GMT 2021] BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved
[Tue Dec 28 02:32:18 GMT 2021] BIOS-e820: [mem 0x00000000000f0000-0x00000000000fffff] reserved
[Tue Dec 28 02:32:18 GMT 2021] BIOS-e820: [mem 0x0000000000100000-0x00000000dffeffff] usable
[Tue Dec 28 02:32:18 GMT 2021] BIOS-e820: [mem 0x00000000dfff0000-0x00000000dfffffff] ACPI data
[Tue Dec 28 02:32:18 GMT 2021] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec00fff] reserved
[Tue Dec 28 02:32:18 GMT 2021] BIOS-e820: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
[Tue Dec 28 02:32:18 GMT 2021] BIOS-e820: [mem 0x00000000fffc0000-0x00000000ffffffff] reserved
[Tue Dec 28 02:32:18 GMT 2021] BIOS-e820: [mem 0x0000000100000000-0x000000011fffffff] usable
[Tue Dec 28 02:32:18 GMT 2021] NX (Execute Disable) protection: active
[Tue Dec 28 02:32:18 GMT 2021] SMBIOS 2.5 present.
[Tue Dec 28 02:32:18 GMT 2021] DMI: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Tue Dec 28 02:32:18 GMT 2021] Hypervisor detected: KVM
[Tue Dec 28 02:32:18 GMT 2021] kvm-clock: Using msrs 4b564d01 and 4b564d00
[Tue Dec 28 02:32:18 GMT 2021] kvm-clock: cpu 0, msr 968b7001, primary cpu clock
[Tue Dec 28 02:32:18 GMT 2021] kvm-clock: using sched offset of 6116840976 cycles
[Tue Dec 28 02:32:18 GMT 2021] clocksource: kvm-clock: mask: 0xffffffffffffffff max_cycles: 0x1cd42e4dffb,
max_idle_ns: 881590591483 ns
[Tue Dec 28 02:32:18 GMT 2021] tsc: Detected 1991.998 MHz processor
[Tue Dec 28 02:32:18 GMT 2021] e820: update [mem 0x00000000-0x00000fff] usable ==> reserved
[Tue Dec 28 02:32:18 GMT 2021] e820: remove [mem 0x000a0000-0x000fffff] usable
[Tue Dec 28 02:32:18 GMT 2021] last_pfn = 0x120000 max_arch_pfn = 0x400000000
[Tue Dec 28 02:32:18 GMT 2021] MTRR default type: uncachable
[Tue Dec 28 02:32:18 GMT 2021] MTRR variable ranges disabled:
[Tue Dec 28 02:32:18 GMT 2021] Disabled
[Tue Dec 28 02:32:18 GMT 2021] x86/PAT: MTRRs disabled, skipping PAT initialization too.
[Tue Dec 28 02:32:18 GMT 2021] CPU MTRRs all blank - virtualized system.
[Tue Dec 28 02:32:18 GMT 2021] x86/PAT: Configuration [0-7]: WB  WT  UC- UC  WB  WT  UC- UC
[Tue Dec 28 02:32:18 GMT 2021] last_pfn = 0xdfff0 max_arch_pfn = 0x400000000
[Tue Dec 28 02:32:18 GMT 2021] found SMP MP-table at [mem 0x0009fff0-0x0009ffff]
[Tue Dec 28 02:32:18 GMT 2021] kexec: Reserving the low 1M of memory for crashkernel
[Tue Dec 28 02:32:18 GMT 2021] RAMDISK: [mem 0x32ec7000-0x3575afff]
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Early table checksum verification disabled
[Tue Dec 28 02:32:18 GMT 2021] ACPI: RSDP 0x00000000000E0000 000024 (v02 VBOX  )
```

```
[Tue Dec 28 02:32:18 GMT 2021] ACPI: XSDT 0x00000000DFFF0030 00003C (v01 VBOX   VBOXXSDT 00000001 ASL  00000061)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: FACP 0x00000000DFFF00F0 0000F4 (v04 VBOX   VBOXFACP 00000001 ASL  00000061)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: DSDT 0x00000000DFFF0480 002325 (v02 VBOX   VBOXBIOS 00000002 INTL 20190509)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: FACS 0x00000000DFFF0200 000040
[Tue Dec 28 02:32:18 GMT 2021] ACPI: FACS 0x00000000DFFF0200 000040
[Tue Dec 28 02:32:18 GMT 2021] ACPI: APIC 0x00000000DFFF0240 00006C (v02 VBOX   VBOXAPIC 00000001 ASL  00000061)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: SSDT 0x00000000DFFF02B0 0001CC (v01 VBOX   VBOXCPUT 00000002 INTL 20190509)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Reserving FACP table memory at [mem 0xdfff00f0-0xdfff01e3]
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Reserving DSDT table memory at [mem 0xdfff0480-0xdfff27a4]
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Reserving FACS table memory at [mem 0xdfff0200-0xdfff023f]
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Reserving FACS table memory at [mem 0xdfff0200-0xdfff023f]
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Reserving APIC table memory at [mem 0xdfff0240-0xdfff02ab]
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Reserving SSDT table memory at [mem 0xdfff02b0-0xdfff047b]
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Local APIC address 0xfee00000
[Tue Dec 28 02:32:18 GMT 2021] No NUMA configuration found
[Tue Dec 28 02:32:18 GMT 2021] Faking a node at [mem 0x0000000000000000-0x000000011fffffff]
[Tue Dec 28 02:32:18 GMT 2021] NODE_DATA(0) allocated [mem 0x11ffd2000-0x11fffbfff]
[Tue Dec 28 02:32:18 GMT 2021] Reserving 128MB of memory at 3440MB for crashkernel (System RAM: 4095MB)
[Tue Dec 28 02:32:18 GMT 2021] Zone ranges:
[Tue Dec 28 02:32:18 GMT 2021]   DMA      [mem 0x0000000000001000-0x0000000000ffffff]
[Tue Dec 28 02:32:18 GMT 2021]   DMA32    [mem 0x0000000001000000-0x00000000ffffffff]
[Tue Dec 28 02:32:18 GMT 2021]   Normal   [mem 0x0000000100000000-0x000000011fffffff]
[Tue Dec 28 02:32:18 GMT 2021]   Device   empty
[Tue Dec 28 02:32:18 GMT 2021] Movable zone start for each node
[Tue Dec 28 02:32:18 GMT 2021] Early memory node ranges
[Tue Dec 28 02:32:18 GMT 2021]   node   0: [mem 0x0000000000001000-0x000000000009efff]
[Tue Dec 28 02:32:18 GMT 2021]   node   0: [mem 0x0000000000100000-0x00000000dffeffff]
[Tue Dec 28 02:32:18 GMT 2021]   node   0: [mem 0x0000000100000000-0x000000011fffffff]
[Tue Dec 28 02:32:18 GMT 2021] Initmem setup node 0 [mem 0x0000000000001000-0x000000011fffffff]
[Tue Dec 28 02:32:18 GMT 2021] On node 0 totalpages: 1048462
[Tue Dec 28 02:32:18 GMT 2021]   DMA zone: 64 pages used for memmap
[Tue Dec 28 02:32:18 GMT 2021]   DMA zone: 158 pages reserved
[Tue Dec 28 02:32:18 GMT 2021]   DMA zone: 3998 pages, LIFO batch:0
[Tue Dec 28 02:32:18 GMT 2021]   DMA32 zone: 14272 pages used for memmap
[Tue Dec 28 02:32:18 GMT 2021]   DMA32 zone: 913392 pages, LIFO batch:63
[Tue Dec 28 02:32:18 GMT 2021]   Normal zone: 2048 pages used for memmap
[Tue Dec 28 02:32:18 GMT 2021]   Normal zone: 131072 pages, LIFO batch:31
[Tue Dec 28 02:32:18 GMT 2021] On node 0, zone DMA: 1 pages in unavailable ranges
[Tue Dec 28 02:32:18 GMT 2021] On node 0, zone DMA: 97 pages in unavailable ranges
[Tue Dec 28 02:32:18 GMT 2021] On node 0, zone Normal: 16 pages in unavailable ranges
[Tue Dec 28 02:32:18 GMT 2021] ACPI: PM-Timer IO Port: 0x4008
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Local APIC address 0xfee00000
[Tue Dec 28 02:32:18 GMT 2021] IOAPIC[0]: apic_id 4, version 32, address 0xfec00000, GSI 0-23
[Tue Dec 28 02:32:18 GMT 2021] ACPI: INT_SRC_OVR (bus 0 bus_irq 0 global_irq 2 dfl dfl)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: INT_SRC_OVR (bus 0 bus_irq 9 global_irq 9 low level)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: IRQ0 used by override.
[Tue Dec 28 02:32:18 GMT 2021] ACPI: IRQ9 used by override.
[Tue Dec 28 02:32:18 GMT 2021] Using ACPI (MADT) for SMP configuration information
[Tue Dec 28 02:32:18 GMT 2021] smpboot: Allowing 4 CPUs, 0 hotplug CPUs
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0x00000000-0x00000fff]
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0x0009f000-0x0009ffff]
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0x000a0000-0x000effff]
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0x000f0000-0x000fffff]
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0xdfff0000-0xdfffffff]
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0xe0000000-0xfebfffff]
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0xfec00000-0xfec00fff]
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0xfec01000-0xfedfffff]
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0xfee00000-0xfee00fff]
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0xfee01000-0xfffbffff]
[Tue Dec 28 02:32:18 GMT 2021] PM: hibernation: Registered nosave memory: [mem 0xfffc0000-0xffffffff]
[Tue Dec 28 02:32:18 GMT 2021] [mem 0xe0000000-0xfebfffff] available for PCI devices
[Tue Dec 28 02:32:18 GMT 2021] Booting paravirtualized kernel on KVM
[Tue Dec 28 02:32:18 GMT 2021] clocksource: refined-jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns:
7645519600211568 ns
[Tue Dec 28 02:32:18 GMT 2021] setup_percpu: NR_CPUS:8192 nr_cpumask_bits:4 nr_cpu_ids:4 nr_node_ids:1
[Tue Dec 28 02:32:18 GMT 2021] percpu: Embedded 58 pages/cpu s200536 r8192 d28840 u524288
[Tue Dec 28 02:32:18 GMT 2021] pcpu-alloc: s200536 r8192 d28840 u524288 alloc=1*2097152
[Tue Dec 28 02:32:18 GMT 2021] pcpu-alloc: [0] 0 1 2 3
[Tue Dec 28 02:32:18 GMT 2021] kvm-guest: PV spinlocks enabled
[Tue Dec 28 02:32:18 GMT 2021] PV qspinlock hash table entries: 256 (order: 0, 4096 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] Built 1 zonelists, mobility grouping on.  Total pages: 1031920
[Tue Dec 28 02:32:18 GMT 2021] Policy zone: Normal
[Tue Dec 28 02:32:18 GMT 2021] Kernel command line: BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64 root=UUID=9cc5ee1e-5533-
4a0b-a88f-903bf52d812d ro quiet crashkernel=384M-:128M
[Tue Dec 28 02:32:18 GMT 2021] Dentry cache hash table entries: 524288 (order: 10, 4194304 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] Inode-cache hash table entries: 262144 (order: 9, 2097152 bytes, linear)
```

```
[Tue Dec 28 02:32:18 GMT 2021] mem auto-init: stack:off, heap alloc:on, heap free:off
[Tue Dec 28 02:32:18 GMT 2021] Memory: 3526712K/4193848K available (12295K kernel code, 2545K rwdata, 7564K rodata,
2408K init, 3684K bss, 346912K reserved, 0K cma-reserved)
[Tue Dec 28 02:32:18 GMT 2021] random: get_random_u64 called from __kmem_cache_create+0x2a/0x4d0 with crng_init=0
[Tue Dec 28 02:32:18 GMT 2021] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=4, Nodes=1
[Tue Dec 28 02:32:18 GMT 2021] Kernel/User page tables isolation: enabled
[Tue Dec 28 02:32:18 GMT 2021] ftrace: allocating 36444 entries in 143 pages
[Tue Dec 28 02:32:18 GMT 2021] ftrace: allocated 143 pages with 5 groups
[Tue Dec 28 02:32:18 GMT 2021] rcu: Hierarchical RCU implementation.
[Tue Dec 28 02:32:18 GMT 2021] rcu:     RCU restricting CPUs from NR_CPUS=8192 to nr_cpu_ids=4.
[Tue Dec 28 02:32:18 GMT 2021]  Rude variant of Tasks RCU enabled.
[Tue Dec 28 02:32:18 GMT 2021]  Tracing variant of Tasks RCU enabled.
[Tue Dec 28 02:32:18 GMT 2021] rcu: RCU calculated value of scheduler-enlistment delay is 25 jiffies.
[Tue Dec 28 02:32:18 GMT 2021] rcu: Adjusting geometry for rcu_fanout_leaf=16, nr_cpu_ids=4
[Tue Dec 28 02:32:18 GMT 2021] NR_IRQS: 524544, nr_irqs: 456, preallocated irqs: 16
[Tue Dec 28 02:32:18 GMT 2021] random: crng done (trusting CPU's manufacturer)
[Tue Dec 28 02:32:18 GMT 2021] Console: colour VGA+ 80x25
[Tue Dec 28 02:32:18 GMT 2021] printk: console [tty0] enabled
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Core revision 20200925
[Tue Dec 28 02:32:18 GMT 2021] APIC: Switch to symmetric I/O mode setup
[Tue Dec 28 02:32:18 GMT 2021] x2apic enabled
[Tue Dec 28 02:32:18 GMT 2021] Switched APIC routing to physical x2apic.
[Tue Dec 28 02:32:18 GMT 2021] ..TIMER: vector=0x30 apic1=0 pin1=2 apic2=-1 pin2=-1
[Tue Dec 28 02:32:18 GMT 2021] clocksource: tsc-early: mask: 0xffffffffffffffff max_cycles: 0x396d4e5fc9d,
max_idle_ns: 881590756024 ns
[Tue Dec 28 02:32:18 GMT 2021] Calibrating delay loop (skipped) preset value.. 3983.99 BogoMIPS (lpj=7967992)
[Tue Dec 28 02:32:18 GMT 2021] pid_max: default: 32768 minimum: 301
[Tue Dec 28 02:32:18 GMT 2021] LSM: Security Framework initializing
[Tue Dec 28 02:32:18 GMT 2021] Yama: disabled by default; enable with sysctl kernel.yama.*
[Tue Dec 28 02:32:18 GMT 2021] AppArmor: AppArmor initialized
[Tue Dec 28 02:32:18 GMT 2021] TOMOYO Linux initialized
[Tue Dec 28 02:32:18 GMT 2021] Mount-cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] Mountpoint-cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] Last level iTLB entries: 4KB 64, 2MB 8, 4MB 8
[Tue Dec 28 02:32:18 GMT 2021] Last level dTLB entries: 4KB 64, 2MB 0, 4MB 0, 1GB 4
[Tue Dec 28 02:32:18 GMT 2021] Spectre V1 : Mitigation: usercopy/swapgs barriers and __user pointer sanitization
[Tue Dec 28 02:32:18 GMT 2021] Spectre V2 : Mitigation: Full generic retpoline
[Tue Dec 28 02:32:18 GMT 2021] Spectre V2 : Spectre v2 / SpectreRSB mitigation: Filling RSB on context switch
[Tue Dec 28 02:32:18 GMT 2021] Speculative Store Bypass: Vulnerable
[Tue Dec 28 02:32:18 GMT 2021] SRBDS: Unknown: Dependent on hypervisor status
[Tue Dec 28 02:32:18 GMT 2021] MDS: Mitigation: Clear CPU buffers
[Tue Dec 28 02:32:18 GMT 2021] Freeing SMP alternatives memory: 32K
[Tue Dec 28 02:32:18 GMT 2021] APIC calibration not consistent with PM-Timer: 97ms instead of 100ms
[Tue Dec 28 02:32:18 GMT 2021] APIC delta adjusted to PM-Timer: 6250278 (6107953)
[Tue Dec 28 02:32:18 GMT 2021] smpboot: CPU0: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz (family: 0x6, model: 0x8e,
stepping: 0xa)
[Tue Dec 28 02:32:18 GMT 2021] Performance Events: unsupported p6 CPU model 142 no PMU driver, software events only.
[Tue Dec 28 02:32:18 GMT 2021] rcu: Hierarchical SRCU implementation.
[Tue Dec 28 02:32:18 GMT 2021] NMI watchdog: Perf NMI watchdog permanently disabled
[Tue Dec 28 02:32:18 GMT 2021] smp: Bringing up secondary CPUs ...
[Tue Dec 28 02:32:18 GMT 2021] x86: Booting SMP configuration:
[Tue Dec 28 02:32:18 GMT 2021] .... node  #0, CPUs:      #1
[Tue Dec 28 02:32:18 GMT 2021] kvm-clock: cpu 1, msr 968b7041, secondary cpu clock
[Tue Dec 28 02:32:18 GMT 2021]  #2
[Tue Dec 28 02:32:18 GMT 2021] kvm-clock: cpu 2, msr 968b7081, secondary cpu clock
[Tue Dec 28 02:32:18 GMT 2021]  #3
[Tue Dec 28 02:32:18 GMT 2021] kvm-clock: cpu 3, msr 968b70c1, secondary cpu clock
[Tue Dec 28 02:32:18 GMT 2021] smp: Brought up 1 node, 4 CPUs
[Tue Dec 28 02:32:18 GMT 2021] smpboot: Max logical packages: 1
[Tue Dec 28 02:32:18 GMT 2021] smpboot: Total of 4 processors activated (15935.98 BogoMIPS)
[Tue Dec 28 02:32:18 GMT 2021] node 0 deferred pages initialised in 0ms
[Tue Dec 28 02:32:18 GMT 2021] devtmpfs: initialized
[Tue Dec 28 02:32:18 GMT 2021] x86/mm: Memory block size: 128MB
[Tue Dec 28 02:32:18 GMT 2021] clocksource: jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns:
7645041785100000 ns
[Tue Dec 28 02:32:18 GMT 2021] futex hash table entries: 1024 (order: 4, 65536 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] pinctrl core: initialized pinctrl subsystem
[Tue Dec 28 02:32:18 GMT 2021] NET: Registered protocol family 16
[Tue Dec 28 02:32:18 GMT 2021] audit: initializing netlink subsys (disabled)
[Tue Dec 28 02:32:18 GMT 2021] audit: type=2000 audit(1640658745.987:1): state=initialized audit_enabled=0 res=1
[Tue Dec 28 02:32:18 GMT 2021] thermal_sys: Registered thermal governor 'fair_share'
[Tue Dec 28 02:32:18 GMT 2021] thermal_sys: Registered thermal governor 'bang_bang'
[Tue Dec 28 02:32:18 GMT 2021] thermal_sys: Registered thermal governor 'step_wise'
[Tue Dec 28 02:32:18 GMT 2021] thermal_sys: Registered thermal governor 'user_space'
[Tue Dec 28 02:32:18 GMT 2021] thermal_sys: Registered thermal governor 'power_allocator'
[Tue Dec 28 02:32:18 GMT 2021] cpuidle: using governor ladder
```

```
[Tue Dec 28 02:32:18 GMT 2021] cpuidle: using governor menu
[Tue Dec 28 02:32:18 GMT 2021] ACPI: bus type PCI registered
[Tue Dec 28 02:32:18 GMT 2021] acpiphp: ACPI Hot Plug PCI Controller Driver version: 0.5
[Tue Dec 28 02:32:18 GMT 2021] PCI: Using configuration type 1 for base access
[Tue Dec 28 02:32:18 GMT 2021] Kprobes globally optimized
[Tue Dec 28 02:32:18 GMT 2021] HugeTLB registered 2.00 MiB page size, pre-allocated 0 pages
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Added _OSI(Module Device)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Added _OSI(Processor Device)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Added _OSI(3.0 _SCP Extensions)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Added _OSI(Processor Aggregator Device)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Added _OSI(Linux-Dell-Video)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Added _OSI(Linux-Lenovo-NV-HDMI-Audio)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Added _OSI(Linux-HPI-Hybrid-Graphics)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: 2 ACPI AML tables successfully acquired and loaded
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Interpreter enabled
[Tue Dec 28 02:32:18 GMT 2021] ACPI: (supports S0 S5)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Using IOAPIC for interrupt routing
[Tue Dec 28 02:32:18 GMT 2021] PCI: Using host bridge windows from ACPI; if necessary, use "pci=nocrs" and report a
bug
[Tue Dec 28 02:32:18 GMT 2021] ACPI: Enabled 2 GPEs in block 00 to 07
[Tue Dec 28 02:32:18 GMT 2021] ACPI: PCI Root Bridge [PCI0] (domain 0000 [bus 00-ff])
[Tue Dec 28 02:32:18 GMT 2021] acpi PNP0A03:00: _OSC: OS supports [ASPM ClockPM Segments MSI HPX-Type3]
[Tue Dec 28 02:32:18 GMT 2021] acpi PNP0A03:00: _OSC: not requesting OS control; OS requires [ExtendedConfig ASPM
ClockPM MSI]
[Tue Dec 28 02:32:18 GMT 2021] acpi PNP0A03:00: fail to add MMCONFIG information, can't access extended PCI
configuration space under this bridge.
[Tue Dec 28 02:32:18 GMT 2021] PCI host bridge to bus 0000:00
[Tue Dec 28 02:32:18 GMT 2021] pci_bus 0000:00: root bus resource [io  0x0000-0x0cf7 window]
[Tue Dec 28 02:32:18 GMT 2021] pci_bus 0000:00: root bus resource [io  0x0d00-0xffff window]
[Tue Dec 28 02:32:18 GMT 2021] pci_bus 0000:00: root bus resource [mem 0x000a0000-0x000bffff window]
[Tue Dec 28 02:32:18 GMT 2021] pci_bus 0000:00: root bus resource [mem 0xe0000000-0xfdffffff window]
[Tue Dec 28 02:32:18 GMT 2021] pci_bus 0000:00: root bus resource [bus 00-ff]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:00.0: [8086:1237] type 00 class 0x060000
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:01.0: [8086:7000] type 00 class 0x060100
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:01.1: [8086:7111] type 00 class 0x01018a
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:01.1: reg 0x20: [io  0xd000-0xd00f]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:01.1: legacy IDE quirk: reg 0x10: [io  0x01f0-0x01f7]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:01.1: legacy IDE quirk: reg 0x14: [io  0x03f6]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:01.1: legacy IDE quirk: reg 0x18: [io  0x0170-0x0177]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:01.1: legacy IDE quirk: reg 0x1c: [io  0x0376]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:02.0: [15ad:0405] type 00 class 0x030000
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:02.0: reg 0x10: [io  0xd010-0xd01f]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:02.0: reg 0x14: [mem 0xe0000000-0xe7ffffff pref]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:02.0: reg 0x18: [mem 0xf0000000-0xf01fffff]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:03.0: [8086:100e] type 00 class 0x020000
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:03.0: reg 0x10: [mem 0xf0200000-0xf021ffff]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:03.0: reg 0x18: [io  0xd020-0xd027]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:04.0: [80ee:cafe] type 00 class 0x088000
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:04.0: reg 0x10: [io  0xd040-0xd05f]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:04.0: reg 0x14: [mem 0xf0400000-0xf07fffff]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:04.0: reg 0x18: [mem 0xf0800000-0xf0803fff pref]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:05.0: [8086:2415] type 00 class 0x040100
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:05.0: reg 0x10: [io  0xd100-0xd1ff]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:05.0: reg 0x14: [io  0xd200-0xd23f]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:06.0: [106b:003f] type 00 class 0x0c0310
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:06.0: reg 0x10: [mem 0xf0804000-0xf0804fff]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:07.0: [8086:7113] type 00 class 0x068000
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:07.0: quirk: [io  0x4000-0x403f] claimed by PIIX4 ACPI
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:07.0: quirk: [io  0x4100-0x410f] claimed by PIIX4 SMB
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:0d.0: [8086:2829] type 00 class 0x010601
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:0d.0: reg 0x10: [io  0xd240-0xd247]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:0d.0: reg 0x14: [io  0xd248-0xd24b]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:0d.0: reg 0x18: [io  0xd250-0xd257]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:0d.0: reg 0x1c: [io  0xd258-0xd25b]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:0d.0: reg 0x20: [io  0xd260-0xd26f]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:0d.0: reg 0x24: [mem 0xf0806000-0xf0807fff]
[Tue Dec 28 02:32:18 GMT 2021] ACPI: PCI Interrupt Link [LNKA] (IRQs 5 9 10 *11)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: PCI Interrupt Link [LNKB] (IRQs 5 9 *10 11)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: PCI Interrupt Link [LNKC] (IRQs 5 *9 10 11)
[Tue Dec 28 02:32:18 GMT 2021] ACPI: PCI Interrupt Link [LNKD] (IRQs 5 9 10 *11)
[Tue Dec 28 02:32:18 GMT 2021] iommu: Default domain type: Translated
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:02.0: vgaarb: setting as boot VGA device
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:02.0: vgaarb: VGA device added: decodes=io+mem,owns=io+mem,locks=none
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:02.0: vgaarb: bridge control possible
[Tue Dec 28 02:32:18 GMT 2021] vgaarb: loaded
[Tue Dec 28 02:32:18 GMT 2021] EDAC MC: Ver: 3.0.0
```

```
[Tue Dec 28 02:32:18 GMT 2021] NetLabel: Initializing
[Tue Dec 28 02:32:18 GMT 2021] NetLabel:  domain hash size = 128
[Tue Dec 28 02:32:18 GMT 2021] NetLabel:  protocols = UNLABELED CIPSOv4 CALIPSO
[Tue Dec 28 02:32:18 GMT 2021] NetLabel:  unlabeled traffic allowed by default
[Tue Dec 28 02:32:18 GMT 2021] PCI: Using ACPI for IRQ routing
[Tue Dec 28 02:32:18 GMT 2021] PCI: pci_cache_line_size set to 64 bytes
[Tue Dec 28 02:32:18 GMT 2021] e820: reserve RAM buffer [mem 0x0009fc00-0x0009ffff]
[Tue Dec 28 02:32:18 GMT 2021] e820: reserve RAM buffer [mem 0xdfff0000-0xdfffffff]
[Tue Dec 28 02:32:18 GMT 2021] clocksource: Switched to clocksource kvm-clock
[Tue Dec 28 02:32:18 GMT 2021] VFS: Disk quotas dquot_6.6.0
[Tue Dec 28 02:32:18 GMT 2021] VFS: Dquot-cache hash table entries: 512 (order 0, 4096 bytes)
[Tue Dec 28 02:32:18 GMT 2021] AppArmor: AppArmor Filesystem Enabled
[Tue Dec 28 02:32:18 GMT 2021] pnp: PnP ACPI init
[Tue Dec 28 02:32:18 GMT 2021] pnp 00:00: Plug and Play ACPI device, IDs PNP0303 (active)
[Tue Dec 28 02:32:18 GMT 2021] pnp 00:01: Plug and Play ACPI device, IDs PNP0f03 (active)
[Tue Dec 28 02:32:18 GMT 2021] pnp: PnP ACPI: found 2 devices
[Tue Dec 28 02:32:18 GMT 2021] clocksource: acpi_pm: mask: 0xffffff max_cycles: 0xffffff, max_idle_ns: 2085701024 ns
[Tue Dec 28 02:32:18 GMT 2021] NET: Registered protocol family 2
[Tue Dec 28 02:32:18 GMT 2021] IP idents hash table entries: 65536 (order: 7, 524288 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] tcp_listen_portaddr_hash hash table entries: 2048 (order: 3, 32768 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] TCP established hash table entries: 32768 (order: 6, 262144 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] TCP bind hash table entries: 32768 (order: 7, 524288 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] TCP: Hash tables configured (established 32768 bind 32768)
[Tue Dec 28 02:32:18 GMT 2021] UDP hash table entries: 2048 (order: 4, 65536 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] UDP-Lite hash table entries: 2048 (order: 4, 65536 bytes, linear)
[Tue Dec 28 02:32:18 GMT 2021] NET: Registered protocol family 1
[Tue Dec 28 02:32:18 GMT 2021] NET: Registered protocol family 44
[Tue Dec 28 02:32:18 GMT 2021] pci_bus 0000:00: resource 4 [io  0x0000-0x0cf7 window]
[Tue Dec 28 02:32:18 GMT 2021] pci_bus 0000:00: resource 5 [io  0x0d00-0xffff window]
[Tue Dec 28 02:32:18 GMT 2021] pci_bus 0000:00: resource 6 [mem 0x000a0000-0x000bffff window]
[Tue Dec 28 02:32:18 GMT 2021] pci_bus 0000:00: resource 7 [mem 0xe0000000-0xfdffffff window]
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:00.0: Limiting direct PCI/PCI transfers
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:01.0: Activating ISA DMA hang workarounds
[Tue Dec 28 02:32:18 GMT 2021] pci 0000:00:02.0: Video device with shadowed ROM at [mem 0x000c0000-0x000dffff]
[Tue Dec 28 02:32:18 GMT 2021] PCI: CLS 0 bytes, default 64
[Tue Dec 28 02:32:18 GMT 2021] Trying to unpack rootfs image as initramfs...
[Tue Dec 28 02:32:19 GMT 2021] Freeing initrd memory: 41552K
[Tue Dec 28 02:32:19 GMT 2021] PCI-DMA: Using software bounce buffering for IO (SWIOTLB)
[Tue Dec 28 02:32:19 GMT 2021] software IO TLB: mapped [mem 0x00000000d3000000-0x00000000d7000000] (64MB)
[Tue Dec 28 02:32:19 GMT 2021] clocksource: tsc: mask: 0xffffffffffffffff max_cycles: 0x396d4e5fc9d, max_idle_ns:
881590756024 ns
[Tue Dec 28 02:32:19 GMT 2021] clocksource: Switched to clocksource tsc
[Tue Dec 28 02:32:19 GMT 2021] platform rtc_cmos: registered platform RTC device (no PNP device found)
[Tue Dec 28 02:32:19 GMT 2021] Initialise system trusted keyrings
[Tue Dec 28 02:32:19 GMT 2021] Key type blacklist registered
[Tue Dec 28 02:32:19 GMT 2021] workingset: timestamp_bits=36 max_order=20 bucket_order=0
[Tue Dec 28 02:32:19 GMT 2021] zbud: loaded
[Tue Dec 28 02:32:19 GMT 2021] integrity: Platform Keyring initialized
[Tue Dec 28 02:32:19 GMT 2021] Key type asymmetric registered
[Tue Dec 28 02:32:19 GMT 2021] Asymmetric key parser 'x509' registered
[Tue Dec 28 02:32:19 GMT 2021] Block layer SCSI generic (bsg) driver version 0.4 loaded (major 251)
[Tue Dec 28 02:32:19 GMT 2021] io scheduler mq-deadline registered
[Tue Dec 28 02:32:19 GMT 2021] shpchp: Standard Hot Plug PCI Controller Driver version: 0.4
[Tue Dec 28 02:32:19 GMT 2021] intel_idle: Please enable MWAIT in BIOS SETUP
[Tue Dec 28 02:32:19 GMT 2021] Serial: 8250/16550 driver, 4 ports, IRQ sharing enabled
[Tue Dec 28 02:32:19 GMT 2021] Linux agpgart interface v0.103
[Tue Dec 28 02:32:19 GMT 2021] AMD-Vi: AMD IOMMUv2 functionality not available on this system - This is not a bug.
[Tue Dec 28 02:32:19 GMT 2021] i8042: PNP: PS/2 Controller [PNP0303:PS2K,PNP0f03:PS2M] at 0x60,0x64 irq 1,12
[Tue Dec 28 02:32:19 GMT 2021] serio: i8042 KBD port at 0x60,0x64 irq 1
[Tue Dec 28 02:32:19 GMT 2021] serio: i8042 AUX port at 0x60,0x64 irq 12
[Tue Dec 28 02:32:19 GMT 2021] mousedev: PS/2 mouse device common for all mice
[Tue Dec 28 02:32:19 GMT 2021] input: AT Translated Set 2 keyboard as /devices/platform/i8042/serio0/input/input0
[Tue Dec 28 02:32:19 GMT 2021] rtc_cmos rtc_cmos: registered as rtc0
[Tue Dec 28 02:32:19 GMT 2021] rtc_cmos rtc_cmos: setting system clock to 2021-12-28T02:32:19 UTC (1640658739)
[Tue Dec 28 02:32:19 GMT 2021] rtc_cmos rtc_cmos: alarms up to one day, 114 bytes nvram
[Tue Dec 28 02:32:19 GMT 2021] intel_pstate: CPU model not supported
[Tue Dec 28 02:32:19 GMT 2021] ledtrig-cpu: registered to indicate activity on CPUs
[Tue Dec 28 02:32:19 GMT 2021] NET: Registered protocol family 10
[Tue Dec 28 02:32:19 GMT 2021] Segment Routing with IPv6
[Tue Dec 28 02:32:19 GMT 2021] mip6: Mobile IPv6
[Tue Dec 28 02:32:19 GMT 2021] NET: Registered protocol family 17
[Tue Dec 28 02:32:19 GMT 2021] mpls_gso: MPLS GSO support
[Tue Dec 28 02:32:19 GMT 2021] IPI shorthand broadcast: enabled
[Tue Dec 28 02:32:19 GMT 2021] sched_clock: Marking stable (1506031014, 14314125)->(1523591256, -3246117)
[Tue Dec 28 02:32:19 GMT 2021] registered taskstats version 1
[Tue Dec 28 02:32:19 GMT 2021] Loading compiled-in X.509 certificates
```

```
[Tue Dec 28 02:32:19 GMT 2021] Loaded X.509 cert 'Debian Secure Boot CA: 6ccece7e4c6c0d1f6149f3dd27dfcc5cbb419ea1'
[Tue Dec 28 02:32:19 GMT 2021] Loaded X.509 cert 'Debian Secure Boot Signer 2021 - linux:
4b6ef5abca669825178e052c84667ccbc0531f8c'
[Tue Dec 28 02:32:19 GMT 2021] zswap: loaded using pool lzo/zbud
[Tue Dec 28 02:32:19 GMT 2021] Key type ._fscrypt registered
[Tue Dec 28 02:32:19 GMT 2021] Key type .fscrypt registered
[Tue Dec 28 02:32:19 GMT 2021] Key type fscrypt-provisioning registered
[Tue Dec 28 02:32:19 GMT 2021] AppArmor: AppArmor sha1 policy hashing enabled
[Tue Dec 28 02:32:19 GMT 2021] Freeing unused kernel image (initmem) memory: 2408K
[Tue Dec 28 02:32:19 GMT 2021] Write protecting the kernel read-only data: 22528k
[Tue Dec 28 02:32:19 GMT 2021] Freeing unused kernel image (text/rodata gap) memory: 2040K
[Tue Dec 28 02:32:19 GMT 2021] Freeing unused kernel image (rodata/data gap) memory: 628K
[Tue Dec 28 02:32:19 GMT 2021] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[Tue Dec 28 02:32:19 GMT 2021] x86/mm: Checking user space page tables
[Tue Dec 28 02:32:19 GMT 2021] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[Tue Dec 28 02:32:19 GMT 2021] Run /init as init process
[Tue Dec 28 02:32:19 GMT 2021]   with arguments:
[Tue Dec 28 02:32:19 GMT 2021]     /init
[Tue Dec 28 02:32:19 GMT 2021]   with environment:
[Tue Dec 28 02:32:19 GMT 2021]     HOME=/
[Tue Dec 28 02:32:19 GMT 2021]     TERM=linux
[Tue Dec 28 02:32:19 GMT 2021]     BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64
[Tue Dec 28 02:32:19 GMT 2021]     crashkernel=384M-:128M
[Tue Dec 28 02:32:19 GMT 2021] input: Power Button as /devices/LNXSYSTM:00/LNXPWRBN:00/input/input2
[Tue Dec 28 02:32:19 GMT 2021] ACPI: Video Device [GFX0] (multi-head: yes  rom: no  post: no)
[Tue Dec 28 02:32:19 GMT 2021] battery: ACPI: Battery Slot [BAT0] (battery present)
[Tue Dec 28 02:32:19 GMT 2021] input: Video Bus as
/devices/LNXSYSTM:00/LNXSYBUS:00/PNP0A03:00/LNXVIDEO:00/input/input3
[Tue Dec 28 02:32:19 GMT 2021] SCSI subsystem initialized
[Tue Dec 28 02:32:19 GMT 2021] ACPI: Power Button [PWRF]
[Tue Dec 28 02:32:19 GMT 2021] input: Sleep Button as /devices/LNXSYSTM:00/LNXSLPBN:00/input/input4
[Tue Dec 28 02:32:19 GMT 2021] ACPI: Sleep Button [SLPF]
[Tue Dec 28 02:32:19 GMT 2021] e1000: Intel(R) PRO/1000 Network Driver
[Tue Dec 28 02:32:19 GMT 2021] e1000: Copyright (c) 1999-2006 Intel Corporation.
[Tue Dec 28 02:32:19 GMT 2021] libata version 3.00 loaded.
[Tue Dec 28 02:32:19 GMT 2021] piix4_smbus 0000:00:07.0: SMBus Host Controller at 0x4100, revision 0
[Tue Dec 28 02:32:19 GMT 2021] ata_piix 0000:00:01.1: version 2.13
[Tue Dec 28 02:32:19 GMT 2021] ahci 0000:00:0d.0: version 3.0
[Tue Dec 28 02:32:19 GMT 2021] ACPI: bus type USB registered
[Tue Dec 28 02:32:19 GMT 2021] usbcore: registered new interface driver usbfs
[Tue Dec 28 02:32:19 GMT 2021] usbcore: registered new interface driver hub
[Tue Dec 28 02:32:19 GMT 2021] usbcore: registered new device driver usb
[Tue Dec 28 02:32:19 GMT 2021] ahci 0000:00:0d.0: SSS flag set, parallel bus scan disabled
[Tue Dec 28 02:32:19 GMT 2021] ahci 0000:00:0d.0: AHCI 0001.0100 32 slots 1 ports 3 Gbps 0x1 impl SATA mode
[Tue Dec 28 02:32:19 GMT 2021] ahci 0000:00:0d.0: flags: 64bit ncq stag only ccc
[Tue Dec 28 02:32:19 GMT 2021] scsi host0: ata_piix
[Tue Dec 28 02:32:19 GMT 2021] ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
[Tue Dec 28 02:32:19 GMT 2021] ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
[Tue Dec 28 02:32:19 GMT 2021] ehci-pci: EHCI PCI platform driver
[Tue Dec 28 02:32:19 GMT 2021] ohci-pci: OHCI PCI platform driver
[Tue Dec 28 02:32:19 GMT 2021] ohci-pci 0000:00:06.0: OHCI PCI host controller
[Tue Dec 28 02:32:19 GMT 2021] ohci-pci 0000:00:06.0: new USB bus registered, assigned bus number 1
[Tue Dec 28 02:32:19 GMT 2021] ohci-pci 0000:00:06.0: irq 22, io mem 0xf0804000
[Tue Dec 28 02:32:19 GMT 2021] scsi host2: ata_piix
[Tue Dec 28 02:32:19 GMT 2021] ata1: PATA max UDMA/33 cmd 0x1f0 ctl 0x3f6 bmdma 0xd000 irq 14
[Tue Dec 28 02:32:19 GMT 2021] ata2: PATA max UDMA/33 cmd 0x170 ctl 0x376 bmdma 0xd008 irq 15
[Tue Dec 28 02:32:19 GMT 2021] scsi host1: ahci
[Tue Dec 28 02:32:19 GMT 2021] ata3: SATA max UDMA/133 abar m8192@0xf0806000 port 0xf0806100 irq 21
[Tue Dec 28 02:32:19 GMT 2021] [drm] DMA map mode: Caching DMA mappings.
[Tue Dec 28 02:32:19 GMT 2021] [drm] Capabilities:
[Tue Dec 28 02:32:19 GMT 2021] [drm]   Cursor.
[Tue Dec 28 02:32:19 GMT 2021] [drm]   Cursor bypass 2.
[Tue Dec 28 02:32:19 GMT 2021] [drm]   Alpha cursor.
[Tue Dec 28 02:32:19 GMT 2021] [drm]   3D.
[Tue Dec 28 02:32:19 GMT 2021] [drm]   Extended Fifo.
[Tue Dec 28 02:32:19 GMT 2021] [drm]   Pitchlock.
[Tue Dec 28 02:32:19 GMT 2021] [drm]   Irq mask.
[Tue Dec 28 02:32:19 GMT 2021] [drm]   GMR.
[Tue Dec 28 02:32:19 GMT 2021] [drm]   Traces.
[Tue Dec 28 02:32:19 GMT 2021] [drm]   GMR2.
[Tue Dec 28 02:32:19 GMT 2021] [drm]   Screen Object 2.
[Tue Dec 28 02:32:19 GMT 2021] [drm] Max GMR ids is 8192
[Tue Dec 28 02:32:19 GMT 2021] [drm] Max number of GMR pages is 1048576
[Tue Dec 28 02:32:19 GMT 2021] [drm] Max dedicated hypervisor surface memory is 393216 kiB
[Tue Dec 28 02:32:19 GMT 2021] [drm] Maximum display memory size is 131072 kiB
[Tue Dec 28 02:32:19 GMT 2021] [drm] VRAM at 0xe0000000 size is 131072 kiB
```

```
[Tue Dec 28 02:32:19 GMT 2021] [drm] MMIO at 0xf0000000 size is 2048 kiB
[Tue Dec 28 02:32:19 GMT 2021] [TTM] Zone  kernel: Available graphics memory: 1946798 KiB
[Tue Dec 28 02:32:19 GMT 2021] [TTM] Initializing pool allocator
[Tue Dec 28 02:32:19 GMT 2021] [TTM] Initializing DMA pool allocator
[Tue Dec 28 02:32:19 GMT 2021] [drm] Screen Objects Display Unit initialized
[Tue Dec 28 02:32:19 GMT 2021] [drm] width 720
[Tue Dec 28 02:32:19 GMT 2021] [drm] height 400
[Tue Dec 28 02:32:19 GMT 2021] [drm] bpp 32
[Tue Dec 28 02:32:19 GMT 2021] [drm] Fifo max 0x00200000 min 0x00001000 cap 0x00000355
[Tue Dec 28 02:32:19 GMT 2021] [drm] Atomic: yes.
[Tue Dec 28 02:32:19 GMT 2021] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.
[Tue Dec 28 02:32:19 GMT 2021] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.
[Tue Dec 28 02:32:19 GMT 2021] fbcon: svgadrmfb (fb0) is primary device
[Tue Dec 28 02:32:19 GMT 2021] Console: switching to colour frame buffer device 100x37
[Tue Dec 28 02:32:19 GMT 2021] [drm] Initialized vmwgfx 2.18.0 20200114 for 0000:00:02.0 on minor 0
[Tue Dec 28 02:32:19 GMT 2021] usb usb1: New USB device found, idVendor=1d6b, idProduct=0001, bcdDevice= 5.10
[Tue Dec 28 02:32:19 GMT 2021] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[Tue Dec 28 02:32:19 GMT 2021] usb usb1: Product: OHCI PCI host controller
[Tue Dec 28 02:32:19 GMT 2021] usb usb1: Manufacturer: Linux 5.10.0-10-amd64 ohci_hcd
[Tue Dec 28 02:32:19 GMT 2021] usb usb1: SerialNumber: 0000:00:06.0
[Tue Dec 28 02:32:19 GMT 2021] hub 1-0:1.0: USB hub found
[Tue Dec 28 02:32:19 GMT 2021] hub 1-0:1.0: 12 ports detected
[Tue Dec 28 02:32:20 GMT 2021] ata2.00: ATAPI: VBOX CD-ROM, 1.0, max UDMA/133
[Tue Dec 28 02:32:20 GMT 2021] scsi 2:0:0:0: CD-ROM            VBOX     CD-ROM           1.0  PQ: 0 ANSI: 5
[Tue Dec 28 02:32:20 GMT 2021] input: ImExPS/2 Generic Explorer Mouse as /devices/platform/i8042/serio1/input/input5
[Tue Dec 28 02:32:20 GMT 2021] ata3: SATA link up 3.0 Gbps (SStatus 123 SControl 300)
[Tue Dec 28 02:32:20 GMT 2021] ata3.00: ATA-6: VBOX HARDDISK, 1.0, max UDMA/133
[Tue Dec 28 02:32:20 GMT 2021] ata3.00: 209715200 sectors, multi 128: LBA48 NCQ (depth 32)
[Tue Dec 28 02:32:20 GMT 2021] ata3.00: configured for UDMA/133
[Tue Dec 28 02:32:20 GMT 2021] scsi 1:0:0:0: Direct-Access     ATA      VBOX HARDDISK    1.0  PQ: 0 ANSI: 5
[Tue Dec 28 02:32:20 GMT 2021] sr 2:0:0:0: [sr0] scsi3-mmc drive: 32x/32x xa/form2 tray
[Tue Dec 28 02:32:20 GMT 2021] cdrom: Uniform CD-ROM driver Revision: 3.20
[Tue Dec 28 02:32:20 GMT 2021] e1000 0000:00:03.0 eth0: (PCI:33MHz:32-bit) 08:00:27:26:5a:6b
[Tue Dec 28 02:32:20 GMT 2021] e1000 0000:00:03.0 eth0: Intel(R) PRO/1000 Network Connection
[Tue Dec 28 02:32:20 GMT 2021] e1000 0000:00:03.0 enp0s3: renamed from eth0
[Tue Dec 28 02:32:20 GMT 2021] sr 2:0:0:0: Attached scsi CD-ROM sr0
[Tue Dec 28 02:32:20 GMT 2021] sd 1:0:0:0: [sda] 209715200 512-byte logical blocks: (107 GB/100 GiB)
[Tue Dec 28 02:32:20 GMT 2021] sd 1:0:0:0: [sda] Write Protect is off
[Tue Dec 28 02:32:20 GMT 2021] sd 1:0:0:0: [sda] Mode Sense: 00 3a 00 00
[Tue Dec 28 02:32:20 GMT 2021] sd 1:0:0:0: [sda] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
[Tue Dec 28 02:32:20 GMT 2021]  sda: sda1 sda2 < sda5 >
[Tue Dec 28 02:32:20 GMT 2021] usb 1-1: new full-speed USB device number 2 using ohci-pci
[Tue Dec 28 02:32:20 GMT 2021] sd 1:0:0:0: [sda] Attached SCSI disk
[Tue Dec 28 02:32:20 GMT 2021] PM: Image not found (code -22)
[Tue Dec 28 02:32:20 GMT 2021] usb 1-1: New USB device found, idVendor=80ee, idProduct=0021, bcdDevice= 1.00
[Tue Dec 28 02:32:20 GMT 2021] usb 1-1: New USB device strings: Mfr=1, Product=3, SerialNumber=0
[Tue Dec 28 02:32:20 GMT 2021] usb 1-1: Product: USB Tablet
[Tue Dec 28 02:32:20 GMT 2021] usb 1-1: Manufacturer: VirtualBox
[Tue Dec 28 02:32:20 GMT 2021] hid: raw HID events driver (C) Jiri Kosina
[Tue Dec 28 02:32:20 GMT 2021] usbcore: registered new interface driver usbhid
[Tue Dec 28 02:32:20 GMT 2021] usbhid: USB HID core driver
[Tue Dec 28 02:32:20 GMT 2021] input: VirtualBox USB Tablet as /devices/pci0000:00/0000:00:06.0/usb1/1-1/1-
1:1.0/0003:80EE:0021.0001/input/input6
[Tue Dec 28 02:32:20 GMT 2021] hid-generic 0003:80EE:0021.0001: input,hidraw0: USB HID v1.10 Mouse [VirtualBox USB
Tablet] on usb-0000:00:06.0-1/input0
[Tue Dec 28 02:32:21 GMT 2021] EXT4-fs (sda1): mounted filesystem with ordered data mode. Opts: (null)
[Tue Dec 28 02:32:21 GMT 2021] Not activating Mandatory Access Control as /sbin/tomoyo-init does not exist.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Inserted module 'autofs4'
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: systemd 247.3-6 running in system mode. (+PAM +AUDIT +SELINUX +IMA
+APPARMOR +SMACK +SYSVINIT +UTMP +LIBCRYPTSETUP +GCRYPT +GNUTLS +ACL +XZ +LZ4 +ZSTD +SECCOMP +BLKID +ELFUTILS +KMOD
+IDN2 -IDN +PCRE2 default-hierarchy=unified)
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Detected virtualization oracle.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Detected architecture x86-64.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Set hostname to <coredump>.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: /lib/systemd/system/plymouth-start.service:16: Unit configured to use
KillMode=none. This is unsafe, as it disables systemd's process lifecycle management for the service. Please update
your service to use a safer KillMode=, such as 'mixed' or 'control-group'. Support for KillMode=none is deprecated and
will eventually be removed.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Queued start job for default target Graphical Interface.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Created slice system-getty.slice.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Created slice system-modprobe.slice.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Created slice User and Session Slice.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Started Forward Password Requests to Wall Directory Watch.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Set up automount Arbitrary Executable File Formats File System Automount
Point.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Reached target User and Group Name Lookups.
```

```
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Reached target Remote File Systems.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Reached target Slices.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Reached target System Time Set.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Reached target System Time Synchronized.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Listening on Syslog Socket.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Listening on fsck to fsckd communication Socket.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Listening on initctl Compatibility Named Pipe.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Listening on Journal Audit Socket.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Listening on Journal Socket (/dev/log).
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Listening on Journal Socket.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Listening on udev Control Socket.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Listening on udev Kernel Socket.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounting Huge Pages File System...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounting POSIX Message Queue File System...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounting Kernel Debug File System...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounting Kernel Trace File System...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Set the console keyboard layout...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Create list of static device nodes for the current kernel...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Load Kernel Module configfs...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Load Kernel Module drm...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Load Kernel Module fuse...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Condition check resulted in Set Up Additional Binary Formats being skipped.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Condition check resulted in File System Check on Root Device being skipped.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Journal Service...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Load Kernel Modules...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Remount Root and Kernel File Systems...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Coldplug All udev Devices...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounted Huge Pages File System.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounted POSIX Message Queue File System.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounted Kernel Debug File System.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounted Kernel Trace File System.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Finished Create list of static device nodes for the current kernel.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: modprobe@configfs.service: Succeeded.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Finished Load Kernel Module configfs.
[Tue Dec 28 02:32:21 GMT 2021] fuse: init (API version 7.32)
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: modprobe@drm.service: Succeeded.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Finished Load Kernel Module drm.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: modprobe@fuse.service: Succeeded.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Finished Load Kernel Module fuse.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounting FUSE Control File System...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounting Kernel Configuration File System...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounted FUSE Control File System.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Mounted Kernel Configuration File System.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Finished Load Kernel Modules.
[Tue Dec 28 02:32:21 GMT 2021] EXT4-fs (sda1): re-mounted. Opts: errors=remount-ro
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Apply Kernel Variables...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Finished Remount Root and Kernel File Systems.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Condition check resulted in Rebuild Hardware Database being skipped.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Condition check resulted in Platform Persistent Storage Archival being
skipped.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Load/Save Random Seed...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Create System Users...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Finished Load/Save Random Seed.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Condition check resulted in First Boot Complete being skipped.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Finished Create System Users.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Starting Create Static Device Nodes in /dev...
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Finished Apply Kernel Variables.
[Tue Dec 28 02:32:21 GMT 2021] systemd[1]: Started Journal Service.
[Tue Dec 28 02:32:21 GMT 2021] systemd-journald[238]: Received client request to flush runtime journal.
[Tue Dec 28 02:32:22 GMT 2021] audit: type=1400 audit(1640658742.096:2): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-senddoc" pid=275 comm="apparmor_parser"
[Tue Dec 28 02:32:22 GMT 2021] audit: type=1400 audit(1640658742.096:3): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="nvidia_modprobe" pid=277 comm="apparmor_parser"
[Tue Dec 28 02:32:22 GMT 2021] audit: type=1400 audit(1640658742.096:4): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="nvidia_modprobe//kmod" pid=277 comm="apparmor_parser"
[Tue Dec 28 02:32:22 GMT 2021] audit: type=1400 audit(1640658742.096:5): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="/usr/bin/man" pid=274 comm="apparmor_parser"
[Tue Dec 28 02:32:22 GMT 2021] audit: type=1400 audit(1640658742.096:6): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="man_filter" pid=274 comm="apparmor_parser"
[Tue Dec 28 02:32:22 GMT 2021] audit: type=1400 audit(1640658742.096:7): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="man_groff" pid=274 comm="apparmor_parser"
[Tue Dec 28 02:32:22 GMT 2021] audit: type=1400 audit(1640658742.116:8): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="lsb_release" pid=278 comm="apparmor_parser"
[Tue Dec 28 02:32:22 GMT 2021] audit: type=1400 audit(1640658742.128:9): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-oopslash" pid=281 comm="apparmor_parser"
```

```
[Tue Dec 28 02:32:22 GMT 2021] audit: type=1400 audit(1640658742.132:10): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-xpdfimport" pid=282 comm="apparmor_parser"
[Tue Dec 28 02:32:22 GMT 2021] ACPI: AC Adapter [AC] (off-line)
[Tue Dec 28 02:32:22 GMT 2021] sr 2:0:0:0: Attached scsi generic sg0 type 5
[Tue Dec 28 02:32:22 GMT 2021] sd 1:0:0:0: Attached scsi generic sg1 type 0
[Tue Dec 28 02:32:22 GMT 2021] vboxguest: loading out-of-tree module taints kernel.
[Tue Dec 28 02:32:22 GMT 2021] input: PC Speaker as /devices/platform/pcspkr/input/input7
[Tue Dec 28 02:32:22 GMT 2021] vboxguest: module verification failed: signature and/or required key missing - tainting
kernel
[Tue Dec 28 02:32:22 GMT 2021] vgdrvHeartbeatInit: Setting up heartbeat to trigger every 2000 milliseconds
[Tue Dec 28 02:32:22 GMT 2021] input: Unspecified device as /devices/pci0000:00/0000:00:04.0/input/input8
[Tue Dec 28 02:32:22 GMT 2021] vboxguest: Successfully loaded version 6.1.30 r148432
[Tue Dec 28 02:32:22 GMT 2021] vboxguest: misc device minor 61, IRQ 20, I/O port d040, MMIO at 00000000f0400000 (size
0x400000)
[Tue Dec 28 02:32:22 GMT 2021] vboxguest: Successfully loaded version 6.1.30 r148432 (interface 0x00010004)
[Tue Dec 28 02:32:22 GMT 2021] Adding 998396k swap on /dev/sda5.  Priority:-2 extents:1 across:998396k FS
[Tue Dec 28 02:32:22 GMT 2021] RAPL PMU: API unit is 2^-32 Joules, 0 fixed counters, 10737418240 ms ovfl timer
[Tue Dec 28 02:32:22 GMT 2021] cryptd: max_cpu_qlen set to 1000
[Tue Dec 28 02:32:22 GMT 2021] AVX2 version of gcm_enc/dec engaged.
[Tue Dec 28 02:32:22 GMT 2021] AES CTR mode by8 optimization enabled
[Tue Dec 28 02:32:22 GMT 2021] snd_intel8x0 0000:00:05.0: allow list rate for 1028:0177 is 48000
[Tue Dec 28 02:32:23 GMT 2021] intel_pmc_core intel_pmc_core.0:  initialized
[Tue Dec 28 02:32:25 GMT 2021] e1000: enp0s3 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[Tue Dec 28 02:32:25 GMT 2021] IPv6: ADDRCONF(NETDEV_CHANGE): enp0s3: link becomes ready
[Tue Dec 28 02:32:31 GMT 2021] vboxvideo: loading version 6.1.30 r148432
[Tue Dec 28 02:32:31 GMT 2021] 02:32:31.558157 main     VBoxService 6.1.30 r148432 (verbosity: 0) linux.amd64 (Nov 22
2021 16:16:32) release log
                               02:32:31.558160 main     Log opened 2021-12-28T02:32:31.558150000Z
[Tue Dec 28 02:32:31 GMT 2021] 02:32:31.558251 main     OS Product: Linux
[Tue Dec 28 02:32:31 GMT 2021] 02:32:31.558282 main     OS Release: 5.10.0-10-amd64
[Tue Dec 28 02:32:31 GMT 2021] 02:32:31.558308 main     OS Version: #1 SMP Debian 5.10.84-1 (2021-12-08)
[Tue Dec 28 02:32:31 GMT 2021] 02:32:31.558332 main     Executable: /opt/VBoxGuestAdditions-6.1.30/sbin/VBoxService
                               02:32:31.558332 main     Process ID: 740
                               02:32:31.558333 main     Package type: LINUX_64BITS_GENERIC
[Tue Dec 28 02:32:31 GMT 2021] 02:32:31.559603 main     6.1.30 r148432 started. Verbose level = 0
[Tue Dec 28 02:32:31 GMT 2021] 02:32:31.560561 main     vbglR3GuestCtrlDetectPeekGetCancelSupport: Supported (#1)
[Tue Dec 28 02:32:36 GMT 2021] rfkill: input handler disabled
[Tue Dec 28 02:32:41 GMT 2021] systemd-journald[238]: File /var/log/journal/7a35ae5c9d954e019d1b34858d5e1923/user-
1000.journal corrupted or uncleanly shut down, renaming and replacing.
[Tue Dec 28 02:32:41 GMT 2021] rfkill: input handler enabled
[Tue Dec 28 02:32:44 GMT 2021] rfkill: input handler disabled
[Tue Dec 28 02:36:54 GMT 2021] sysrq: Trigger a crash
[Tue Dec 28 02:36:54 GMT 2021] Kernel panic - not syncing: sysrq triggered crash
[Tue Dec 28 02:36:54 GMT 2021] CPU: 3 PID: 2135 Comm: tee Kdump: loaded Tainted: G           OE     5.10.0-10-amd64 #1
Debian 5.10.84-1
[Tue Dec 28 02:36:54 GMT 2021] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Tue Dec 28 02:36:54 GMT 2021] Call Trace:
[Tue Dec 28 02:36:54 GMT 2021]  dump_stack+0x6b/0x83
[Tue Dec 28 02:36:54 GMT 2021]  panic+0x101/0x2d7
[Tue Dec 28 02:36:54 GMT 2021]  ? printk+0x58/0x6f
[Tue Dec 28 02:36:54 GMT 2021]  sysrq_handle_crash+0x16/0x20
[Tue Dec 28 02:36:54 GMT 2021]  __handle_sysrq.cold+0x43/0x113
[Tue Dec 28 02:36:54 GMT 2021]  write_sysrq_trigger+0x24/0x40
[Tue Dec 28 02:36:54 GMT 2021]  proc_reg_write+0x51/0x90
[Tue Dec 28 02:36:54 GMT 2021]  vfs_write+0xc0/0x260
[Tue Dec 28 02:36:54 GMT 2021]  ksys_write+0x5f/0xe0
[Tue Dec 28 02:36:54 GMT 2021]  do_syscall_64+0x33/0x80
[Tue Dec 28 02:36:54 GMT 2021]  entry_SYSCALL_64_after_hwframe+0x44/0xa9
[Tue Dec 28 02:36:54 GMT 2021] RIP: 0033:0x7f1ddc1f0f33
[Tue Dec 28 02:36:54 GMT 2021] Code: 8b 15 61 ef 0c 00 f7 d8 64 89 02 48 c7 c0 ff ff ff ff eb b7 0f 1f 00 64 8b 04 25
18 00 00 00 85 c0 75 14 b8 01 00 00 00 0f 05 <48> 3d 00 f0 ff ff 77 55 c3 0f 1f 40 00 48 83 ec 28 48 89 54 24 18
[Tue Dec 28 02:36:54 GMT 2021] RSP: 002b:00007ffea91896f8 EFLAGS: 00000246 ORIG_RAX: 0000000000000001
[Tue Dec 28 02:36:54 GMT 2021] RAX: ffffffffffffffda RBX: 0000000000000002 RCX: 00007f1ddc1f0f33
[Tue Dec 28 02:36:54 GMT 2021] RDX: 0000000000000002 RSI: 00007ffea9189810 RDI: 0000000000000003
[Tue Dec 28 02:36:54 GMT 2021] RBP: 00007ffea9189810 R08: 0000000000000000 R09: 0000000000000001
[Tue Dec 28 02:36:54 GMT 2021] R10: fffffffffffff286 R11: 0000000000000246 R12: 0000000000000002
[Tue Dec 28 02:36:54 GMT 2021] R13: 000055be3051d4a0 R14: 0000000000000002 R15: 00007f1ddc2c18a0
```

Note: Many commands have many options. Please check individual help entries.

```
crash> help dmesg
```

6. Check memory summary, computer, and network information:

```
crash> kmem -i
                 PAGES        TOTAL      PERCENTAGE
    TOTAL MEM   973399       3.7 GB         ----
         FREE   583421       2.2 GB   59% of TOTAL MEM
         USED   389978       1.5 GB   40% of TOTAL MEM
       SHARED    94513     369.2 MB    9% of TOTAL MEM
      BUFFERS    10263      40.1 MB    1% of TOTAL MEM
       CACHED   208254     813.5 MB   21% of TOTAL MEM
         SLAB    10668      41.7 MB    1% of TOTAL MEM

   TOTAL HUGE        0            0         ----
    HUGE FREE        0            0    0% of TOTAL HUGE

   TOTAL SWAP   249599       975 MB         ----
    SWAP USED        0            0    0% of TOTAL SWAP
    SWAP FREE   249599       975 MB  100% of TOTAL SWAP

 COMMIT LIMIT   736298       2.8 GB         ----
    COMMITTED   892999       3.4 GB  121% of TOTAL LIMIT

crash> mach
          MACHINE TYPE: x86_64
           MEMORY SIZE: 4 GB
                  CPUS: 4
            HYPERVISOR: KVM
       PROCESSOR SPEED: 1991 Mhz
                    HZ: 250
             PAGE SIZE: 4096
   KERNEL VIRTUAL BASE: ffff9a2b40000000
   KERNEL VMALLOC BASE: ffffa77fc0000000
   KERNEL VMEMMAP BASE: ffffc9ac00000000
      KERNEL START MAP: ffffffff80000000
   KERNEL MODULES BASE: ffffffffc0000000
     KERNEL STACK SIZE: 16384
        IRQ STACK SIZE: 16384
            IRQ STACKS:
                 CPU 0: ffffa77fc0000000
                 CPU 1: ffffa77fc00c5000
                 CPU 2: ffffa77fc00f1000
                 CPU 3: ffffa77fc011d000
DOUBLEFAULT STACK SIZE: 8192
    DOUBLEFAULT STACKS:
                 CPU 0: fffffe0000009000
                 CPU 1: fffffe0000044000
                 CPU 2: fffffe000007f000
                 CPU 3: fffffe00000ba000
        NMI STACK SIZE: 8192
            NMI STACKS:
                 CPU 0: fffffe000000c000
                 CPU 1: fffffe0000047000
                 CPU 2: fffffe0000082000
                 CPU 3: fffffe00000bd000
      DEBUG STACK SIZE: 8192
          DEBUG STACKS:
                 CPU 0: fffffe000000f000
                 CPU 1: fffffe000004a000
                 CPU 2: fffffe0000085000
                 CPU 3: fffffe00000c0000
```

```
        MCE STACK SIZE: 8192
            MCE STACKS:
                 CPU 0: fffffe0000012000
                 CPU 1: fffffe000004d000
                 CPU 2: fffffe0000088000
                 CPU 3: fffffe00000c3000
  (unknown) STACK SIZE: 0
      (unknown) STACKS:
                 CPU 0: fffffe0000017000
                 CPU 1: fffffe0000052000
                 CPU 2: fffffe000008d000
                 CPU 3: fffffe00000c8000
```

```
crash> net
   NET_DEVICE     NAME   IP ADDRESS(ES)
ffff9a2c403be000  lo     127.0.0.1
ffff9a2c58c5c000  enp0s3 10.0.2.15
```

7. List all processes:

```
crash> ps
   PID    PPID  CPU       TASK        ST  %MEM     VSZ    RSS  COMM
>     0      0   0  ffffffff91213940  RU   0.0       0      0  [swapper/0]
>     0      0   1  ffff9a2c4024df00  RU   0.0       0      0  [swapper/1]
>     0      0   2  ffff9a2c402697c0  RU   0.0       0      0  [swapper/2]
      0      0   3  ffff9a2c4026df00  RU   0.0       0      0  [swapper/3]
      1      0   3  ffff9a2c401f4740  IN   0.2  164092  10312  systemd
      2      0   1  ffff9a2c401f2f80  IN   0.0       0      0  [kthreadd]
      3      2   0  ffff9a2c401f0000  ID   0.0       0      0  [rcu_gp]
      4      2   0  ffff9a2c401f17c0  ID   0.0       0      0  [rcu_par_gp]
      5      2   0  ffff9a2c401f5f00  ID   0.0       0      0  [kworker/0:0]
      6      2   0  ffff9a2c402297c0  ID   0.0       0      0  [kworker/0:0H]
      7      2   0  ffff9a2c4022df00  ID   0.0       0      0  [kworker/0:1]
      8      2   3  ffff9a2c4022c740  ID   0.0       0      0  [kworker/u8:0]
      9      2   0  ffff9a2c4022af80  ID   0.0       0      0  [mm_percpu_wq]
     10      2   0  ffff9a2c40228000  IN   0.0       0      0  [rcu_tasks_rude_]
     11      2   0  ffff9a2c4024c740  IN   0.0       0      0  [rcu_tasks_trace]
     12      2   0  ffff9a2c4024af80  IN   0.0       0      0  [ksoftirqd/0]
     13      2   1  ffff9a2c40248000  ID   0.0       0      0  [rcu_sched]
     14      2   0  ffff9a2c402497c0  IN   0.0       0      0  [migration/0]
     15      2   0  ffff9a2c4026c740  IN   0.0       0      0  [cpuhp/0]
     16      2   1  ffff9a2c4026af80  IN   0.0       0      0  [cpuhp/1]
     17      2   1  ffff9a2c40268000  IN   0.0       0      0  [migration/1]
     18      2   1  ffff9a2c4028af80  IN   0.0       0      0  [ksoftirqd/1]
     19      2   1  ffff9a2c40288000  ID   0.0       0      0  [kworker/1:0]
     20      2   1  ffff9a2c402897c0  ID   0.0       0      0  [kworker/1:0H]
     21      2   2  ffff9a2c4028df00  IN   0.0       0      0  [cpuhp/2]
     22      2   2  ffff9a2c4028c740  IN   0.0       0      0  [migration/2]
     23      2   2  ffff9a2c402b4740  IN   0.0       0      0  [ksoftirqd/2]
     24      2   2  ffff9a2c402b2f80  ID   0.0       0      0  [kworker/2:0]
     25      2   2  ffff9a2c402b0000  ID   0.0       0      0  [kworker/2:0H]
     26      2   3  ffff9a2c402b17c0  IN   0.0       0      0  [cpuhp/3]
     27      2   3  ffff9a2c402b5f00  IN   0.0       0      0  [migration/3]
     28      2   3  ffff9a2c402e17c0  IN   0.0       0      0  [ksoftirqd/3]
     29      2   3  ffff9a2c402e5f00  ID   0.0       0      0  [kworker/3:0]
     30      2   3  ffff9a2c402e4740  ID   0.0       0      0  [kworker/3:0H]
     32      2   1  ffff9a2c4031af80  ID   0.0       0      0  [kworker/u8:1]
     33      2   2  ffff9a2c40342f80  ID   0.0       0      0  [kworker/u8:2]
     34      2   0  ffff9a2c40340000  IN   0.0       0      0  [kdevtmpfs]
     35      2   1  ffff9a2c403417c0  ID   0.0       0      0  [netns]
```

```
 36      2   2  ffff9a2c40345f00  IN   0.0       0      0  [kauditd]
 37      2   1  ffff9a2c40344740  IN   0.0       0      0  [khungtaskd]
 38      2   3  ffff9a2c5bd717c0  IN   0.0       0      0  [oom_reaper]
 39      2   1  ffff9a2c5bd75f00  ID   0.0       0      0  [writeback]
 40      2   1  ffff9a2c5bd74740  IN   0.0       0      0  [kcompactd0]
 41      2   1  ffff9a2c5bd72f80  IN   0.0       0      0  [ksmd]
 42      2   1  ffff9a2c5bd70000  IN   0.0       0      0  [khugepaged]
 44      2   3  ffff9a2c5bdddf00  ID   0.0       0      0  [kworker/3:1]
 52      2   1  ffff9a2c5bdf0000  ID   0.0       0      0  [kworker/1:1]
 62      2   0  ffff9a2c403a2f80  ID   0.0       0      0  [kintegrityd]
 63      2   0  ffff9a2c4082c740  ID   0.0       0      0  [kblockd]
 64      2   1  ffff9a2c4082af80  ID   0.0       0      0  [blkcg_punt_bio]
 65      2   3  ffff9a2c40828000  ID   0.0       0      0  [edac-poller]
 66      2   1  ffff9a2c408297c0  ID   0.0       0      0  [devfreq_wq]
 67      2   0  ffff9a2c4082df00  ID   0.0       0      0  [kworker/0:1H]
 69      2   3  ffff9a2c47ef0000  IN   0.0       0      0  [kswapd0]
 70      2   1  ffff9a2c47ef17c0  ID   0.0       0      0  [kthrotld]
 71      2   3  ffff9a2c47ef5f00  ID   0.0       0      0  [acpi_thermal_pm]
 72      2   1  ffff9a2c47ef4740  ID   0.0       0      0  [ipv6_addrconf]
 77      2   2  ffff9a2c403a4740  ID   0.0       0      0  [kworker/2:1]
 82      2   2  ffff9a2c403a0000  ID   0.0       0      0  [kstrp]
 85      2   3  ffff9a2c4039df00  ID   0.0       0      0  [zswap-shrink]
 86      2   2  ffff9a2c403997c0  ID   0.0       0      0  [kworker/u9:0]
108      2   1  ffff9a2c58e65f00  ID   0.0       0      0  [kworker/1:1H]
122      2   3  ffff9a2c58e14740  ID   0.0       0      0  [kworker/3:2]
130      2   3  ffff9a2c58dc17c0  ID   0.0       0      0  [kworker/3:1H]
131      2   0  ffff9a2c4039c740  ID   0.0       0      0  [ata_sff]
133      2   1  ffff9a2c40398000  IN   0.0       0      0  [scsi_eh_0]
134      2   1  ffff9a2c50d94740  ID   0.0       0      0  [scsi_tmf_0]
137      2   3  ffff9a2c4031c740  ID   0.0       0      0  [kworker/3:3]
138      2   1  ffff9a2c591d2f80  IN   0.0       0      0  [scsi_eh_1]
139      2   3  ffff9a2c591d0000  ID   0.0       0      0  [scsi_tmf_1]
140      2   2  ffff9a2c50d92f80  IN   0.0       0      0  [scsi_eh_2]
141      2   0  ffff9a2c50d90000  ID   0.0       0      0  [scsi_tmf_2]
142      2   0  ffff9a2c5bddc740  ID   0.0       0      0  [kworker/u8:3]
143      2   1  ffff9a2c403a17c0  IN   0.0       0      0  [irq/18-vmwgfx]
144      2   1  ffff9a2c403a5f00  ID   0.0       0      0  [ttm_swap]
145      2   1  ffff9a2c402e2f80  IN   0.0       0      0  [card0-crtc0]
146      2   1  ffff9a2c402e0000  IN   0.0       0      0  [card0-crtc1]
147      2   1  ffff9a2c50efdf00  IN   0.0       0      0  [card0-crtc2]
148      2   1  ffff9a2c50efc740  IN   0.0       0      0  [card0-crtc3]
149      2   1  ffff9a2c50efaf80  IN   0.0       0      0  [card0-crtc4]
150      2   1  ffff9a2c50ef8000  IN   0.0       0      0  [card0-crtc5]
151      2   1  ffff9a2c50ef97c0  IN   0.0       0      0  [card0-crtc6]
152      2   1  ffff9a2c50f10000  IN   0.0       0      0  [card0-crtc7]
153      2   2  ffff9a2c50f117c0  ID   0.0       0      0  [kworker/2:1H]
154      2   3  ffff9a2c5bdd8000  ID   0.0       0      0  [kworker/3:4]
155      2   1  ffff9a2c5bdd97c0  ID   0.0       0      0  [kworker/1:2]
157      2   0  ffff9a2c517edf00  ID   0.0       0      0  [kworker/0:2]
159      2   2  ffff9a2c517eaf80  ID   0.0       0      0  [kworker/2:2]
197      2   1  ffff9a2c47ef2f80  IN   0.0       0      0  [jbd2/sda1-8]
198      2   1  ffff9a2c50f12f80  ID   0.0       0      0  [ext4-rsv-conver]
238      1   1  ffff9a2c403197c0  IN   0.4   42260  17140  systemd-journal
259      1   0  ffff9a2c59ca97c0  IN   0.1   23408   6432  systemd-udevd
336      2   2  ffff9a2c5a2d2f80  ID   0.0       0      0  [iprt-VBoxWQueue]
451      1   1  ffff9a2c51cb97c0  IN   0.2  236304   7512  accounts-daemon
454      1   1  ffff9a2c51cbdf00  IN   0.1    7272   3968  avahi-daemon
456      1   0  ffff9a2c51cb8000  IN   0.1    6684   2876  cron
459      1   2  ffff9a2c51cbaf80  IN   0.1    9748   6116  dbus-daemon
465      1   2  ffff9a2c51cbc740  IN   0.4  254472  17032  NetworkManager
```

```
  467      1   1  ffff9a2c591d4740  IN   0.2  236304   7512  gmain
  478      1   3  ffff9a2c475e0000  IN   0.2  235884  10160  polkitd
  479      1   2  ffff9a2c59cac740  IN   0.1  220740   6840  rsyslogd
  481      1   1  ffff9a2c59caaf80  IN   0.1  232780   6120  switcheroo-cont
  484      1   1  ffff9a2c59cadf00  IN   0.2   22092   7464  systemd-logind
  485      1   2  ffff9a2c59ca8000  IN   0.3  393692  12892  udisksd
  487      1   3  ffff9a2c58f34740  IN   0.1   14560   6548  wpa_supplicant
  489      1   2  ffff9a2c4746df00  IN   0.2  235884  10160  gmain
  490    454   3  ffff9a2c58e15f00  IN   0.0    7092   1348  avahi-daemon
  494      1   1  ffff9a2c50220000  IN   0.1  220740   6840  in:imuxsock
  495      1   3  ffff9a2c50222f80  IN   0.1  220740   6840  in:imklog
  499      1   1  ffff9a2c50225f00  IN   0.1  220740   6840  rs:main Q:Reg
  500      1   2  ffff9a2c475ec740  IN   0.1  232780   6120  gmain
  504      1   3  ffff9a2c58f317c0  IN   0.3  393692  12892  gmain
  505      2   3  ffff9a2c502217c0  ID   0.0       0      0  [cryptd]
  515      1   2  ffff9a2c50224740  IN   0.2  235884  10160  gdbus
  516      1   1  ffff9a2c58dc0000  IN   0.1  232780   6120  gdbus
  517      1   2  ffff9a2c50d95f00  IN   0.3  393692  12892  gdbus
  518      1   1  ffff9a2c50d917c0  IN   0.2  236304   7512  gdbus
  525      1   1  ffff9a2c47468000  IN   0.4  254472  17032  gmain
  528      1   3  ffff9a2c412e2f80  IN   0.3  314784  13096  ModemManager
  529      1   2  ffff9a2c4746c740  IN   0.4  254472  17032  gdbus
  539      1   2  ffff9a2c48c08000  IN   0.5  118724  24060  unattended-upgr
  546      2   1  ffff9a2c475e17c0  ID   0.0       0      0  [kworker/1:3]
  549      1   3  ffff9a2c58e117c0  IN   0.3  393692  12892  probing-thread
  560      1   0  ffff9a2c48c0af80  IN   0.3  314784  13096  gmain
  577      1   0  ffff9a2c412e4740  IN   0.3  314784  13096  gdbus
  614      1   2  ffff9a2c517ec740  IN   0.3  393692  12892  cleanup
  650      1   3  ffff9a2c5a2d4740  IN   0.5  118724  24060  gmain
  745      1   0  ffff9a2c41894740  IN   0.1  293568   3624  VBoxService
  747      1   3  ffff9a2c41895f00  IN   0.2  239628   8600  gdm3
  748      1   0  ffff9a2c517e97c0  IN   0.1  293568   3624  RTThrdPP
  749      1   0  ffff9a2c412e5f00  IN   0.1  293568   3624  control
  750      1   1  ffff9a2c4039af80  IN   0.1  293568   3624  timesync
  751      1   2  ffff9a2c58dc5f00  IN   0.1  293568   3624  vminfo
  752      1   1  ffff9a2c58dc4740  IN   0.1  293568   3624  cpuhotplug
  753      1   3  ffff9a2c58dc2f80  IN   0.1  293568   3624  memballoon
  754      1   3  ffff9a2c58e10000  IN   0.1  293568   3624  vmstats
  755      1   0  ffff9a2c58e64740  IN   0.1  293568   3624  automount
  756      1   2  ffff9a2c459a97c0  IN   0.2  239628   8600  gmain
  757      1   2  ffff9a2c459ac740  IN   0.2  239628   8600  gdbus
  790      1   0  ffff9a2c475e8000  IN   0.1  153692   3328  rtkit-daemon
  792      1   2  ffff9a2c58e617c0  IN   0.1  153692   3328  rtkit-daemon
  793      1   0  ffff9a2c58e60000  IN   0.1  153692   3328  rtkit-daemon
  883      1   2  ffff9a2b4bc85f00  IN   0.2  247080  10652  upowerd
  886      1   1  ffff9a2b46fc8000  IN   0.2  247080  10652  gmain
  887      1   2  ffff9a2b46fc97c0  IN   0.2  247080  10652  gdbus
  955      1   3  ffff9a2b554c0000  IN   0.8  364656  35600  packagekitd
  959      1   0  ffff9a2b55654740  IN   0.8  364656  35600  gmain
  960      1   0  ffff9a2b55652f80  IN   0.8  364656  35600  gdbus
 1092      1   2  ffff9a2b5ab0df00  IN   0.3  242868  13404  colord
 1095      1   2  ffff9a2b4be15f00  IN   0.3  242868  13404  gmain
 1099      1   3  ffff9a2b4be14740  IN   0.3  242868  13404  gdbus
 1158    747   1  ffff9a2b5ab02f80  IN   0.2  166624  10032  gdm-session-wor
 1159    747   1  ffff9a2b519d17c0  IN   0.2  166624  10032  gmain
 1160    747   1  ffff9a2b519d0000  IN   0.2  166624  10032  gdbus
 1163      1   3  ffff9a2b62e18000  IN   0.2   15744   9372  systemd
 1164   1163   0  ffff9a2b62e197c0  IN   0.1  167096   4520  (sd-pam)
 1183   1163   1  ffff9a2b62e1af80  IN   0.1   90572   5712  pipewire
 1184   1163   3  ffff9a2c591d17c0  IN   0.6 1156112  28768  pulseaudio
```

```
1186   1163   1  ffff9a2b51988000  IN   0.5  509528  25252  tracker-miner-f
1187   1163   2  ffff9a2c583d17c0  IN   0.1   90572   5712  pipewire
1190   1163   3  ffff9a2b5198df00  IN   0.1    9036   5492  dbus-daemon
1203   1163   1  ffff9a2b5ab00000  IN   0.5  509528  25252  gmain
1205      1   0  ffff9a2b5187df00  IN   0.2  237356   9684  gnome-keyring-d
1206      1   3  ffff9a2b5a99df00  IN   0.2  237356   9684  gmain
1207   1163   1  ffff9a2b5ab0af80  IN   0.5  509528  25252  gdbus
1208   1163   3  ffff9a2b554c4740  IN   0.5  509528  25252  dconf worker
1209      1   0  ffff9a2b5a998000  IN   0.2  237356   9684  gdbus
1210   1163   3  ffff9a2b5a91df00  IN   0.2  236900   7636  gvfsd
1211   1163   1  ffff9a2b554c5f00  IN   0.2  236900   7636  gmain
1212   1163   0  ffff9a2c517e8000  IN   0.2  236900   7636  gdbus
1215   1163   3  ffff9a2c48c097c0  IN   0.2  379924   8620  gvfsd-fuse
1218   1163   1  ffff9a2b62c6c740  IN   0.2  379924   8620  gvfsd-fuse
1219   1163   3  ffff9a2c48c0df00  IN   0.2  379924   8620  gvfsd-fuse
1220   1163   3  ffff9a2b5198af80  IN   0.2  379924   8620  gmain
1221   1163   1  ffff9a2b62cac740  IN   0.2  379924   8620  gdbus
1223   1183   1  ffff9a2b5aa6df00  IN   0.1   85300   6536  pipewire-media-
1224   1158   1  ffff9a2b62ca8000  IN   0.1  158836   5856  gdm-wayland-ses
1225   1183   3  ffff9a2b62d02f80  IN   0.1   85300   6536  pipewire-media-
1226   1163   3  ffff9a2b62caaf80  IN   0.3  349052  14820  gvfs-udisks2-vo
1227   1163   1  ffff9a2b62ca97c0  IN   0.2  379924   8620  gvfs-fuse-sub
1228   1158   1  ffff9a2b62cadf00  IN   0.1  158836   5856  gmain
1229   1158   0  ffff9a2b62d54740  IN   0.1  158836   5856  gdbus
1230   1224   0  ffff9a2b62d52f80  IN   0.3  297996  16036  gnome-session-b
1231   1163   2  ffff9a2b62c6af80  IN   0.3  349052  14820  gmain
1233   1163   2  ffff9a2b62c68000  IN   0.3  349052  14820  gdbus
1241   1163   3  ffff9a2b62d00000  IN   0.3  349052  14820  dconf worker
1251   1163   3  ffff9a2b65c00000  IN   0.1  235108   6976  gvfs-gphoto2-vo
1255   1163   0  ffff9a2b5ab0c740  IN   0.1  235108   6976  gmain
1257   1163   3  ffff9a2b65c4af80  IN   0.1  235108   6976  gdbus
1263   1163   3  ffff9a2b65c05f00  IN   0.2  311556   9824  gvfs-afc-volume
1264   1163   2  ffff9a2b62dedf00  IN   0.2  311556   9824  gvfs-afc-volume
1265   1163   2  ffff9a2b62dec740  IN   0.2  311556   9824  gmain
1267   1163   0  ffff9a2b65c017c0  IN   0.2  311556   9824  gdbus
1269   1163   3  ffff9a2b65c4df00  IN   0.1  233064   6432  gvfs-goa-volume
1274   1163   0  ffff9a2c474697c0  IN   0.1  233064   6432  gmain
1275   1163   0  ffff9a2b62c697c0  IN   0.1  233064   6432  gdbus
1277   1163   3  ffff9a2c58e12f80  IN   0.8  550096  38708  goa-daemon
1284   1224   1  ffff9a2b65d78000  IN   0.3  297996  16036  gmain
1285   1163   1  ffff9a2b65c04740  IN   0.6 1156112  28768  alsa-sink-Intel
1286   1224   0  ffff9a2b65d7df00  IN   0.3  297996  16036  gdbus
1287   1224   1  ffff9a2b65d7c740  IN   0.3  297996  16036  dconf worker
1288   1163   0  ffff9a2b65d7af80  IN   0.1   88176   5020  gnome-session-c
1289   1163   0  ffff9a2b62d05f00  IN   0.1    5964   4132  ssh-agent
1291   1163   3  ffff9a2b62d04740  IN   0.4  519724  17260  gnome-session-b
1292   1163   2  ffff9a2b65c497c0  IN   0.1   88176   5020  gmain
1294   1163   0  ffff9a2b65d1df00  IN   0.8  550096  38708  gmain
1296   1163   0  ffff9a2b65c4c740  IN   0.8  550096  38708  gdbus
1297   1163   0  ffff9a2b65d1af80  IN   0.8  550096  38708  dconf worker
1300   1163   0  ffff9a2b65c48000  IN   0.2  311788  11304  goa-identity-se
1301   1163   2  ffff9a2b65ee8000  IN   0.2  311788  11304  gmain
1303   1163   2  ffff9a2c58f35f00  IN   0.4  519724  17260  gmain
1305   1163   1  ffff9a2c58f30000  IN   0.4  519724  17260  gdbus
1306   1163   2  ffff9a2b62de8000  IN   0.2  311788  11304  gdbus
1308   1163   3  ffff9a2b65eec740  IN   0.1  232872   6392  gvfs-mtp-volume
1309   1163   0  ffff9a2b65efc740  IN   0.1  232872   6392  gmain
1311   1163   0  ffff9a2b65eeaf80  IN   0.1  232872   6392  gdbus
1313   1163   3  ffff9a2b65f7c740  IN   0.4  519724  17260  dconf worker
1319      1   2  ffff9a2b65c02f80  IN   0.2  237356   9684  timer
```

```
1320   1163   3  ffff9a2b68815f00  IN   0.5  509528  25252  pool-tracker-mi
1324   1163   3  ffff9a2b65d197c0  IN   0.6 1156112  28768  alsa-source-Int
1327   1163   1  ffff9a2b68860000  IN   5.6 5187156 265988  gnome-shell
1330   1291   0  ffff9a2b68810000  IN   0.2  307284   8612  at-spi-bus-laun
1332   1291   3  ffff9a2b68862f80  IN   0.2  307284   8612  gmain
1333   1291   3  ffff9a2b688617c0  IN   0.2  307284   8612  dconf worker
1335   1291   0  ffff9a2b65ef97c0  IN   0.2  307284   8612  gdbus
1336   1330   0  ffff9a2b68982f80  IN   0.1    8040   4436  dbus-daemon
1339   1163   3  ffff9a2b62db17c0  IN   5.6 5187156 265988  gmain
1343   1163   0  ffff9a2b688117c0  IN   5.6 5187156 265988  gdbus
1348   1163   3  ffff9a2b62db0000  IN   5.6 5187156 265988  dconf worker
1349   1163   1  ffff9a2b62d50000  IN   5.6 5187156 265988  llvmpipe-0
1350   1163   2  ffff9a2b62d55f00  IN   5.6 5187156 265988  llvmpipe-1
1351   1163   3  ffff9a2b62d517c0  IN   5.6 5187156 265988  llvmpipe-2
1352   1163   1  ffff9a2b689eaf80  IN   5.6 5187156 265988  llvmpipe-3
1353   1163   3  ffff9a2b689e8000  IN   5.6 5187156 265988  gnome-shell
1354   1163   0  ffff9a2b689e97c0  IN   5.6 5187156 265988  gnome-shell
1355   1163   2  ffff9a2b689edf00  IN   5.6 5187156 265988  gnome-shell
1356   1163   3  ffff9a2b689ec740  IN   5.6 5187156 265988  gnome-shell
1357   1163   0  ffff9a2b68a30000  IN   5.6 5187156 265988  gnome-s:disk$0
1358   1163   2  ffff9a2b68a317c0  IN   5.6 5187156 265988  gnome-s:disk$1
1359   1163   3  ffff9a2b68a35f00  IN   5.6 5187156 265988  gnome-s:disk$2
1360   1163   2  ffff9a2b68a34740  IN   5.6 5187156 265988  gnome-s:disk$3
1361   1163   2  ffff9a2b68985f00  IN   5.6 5187156 265988  JS Helper
1362   1163   1  ffff9a2b65d1c740  IN   5.6 5187156 265988  JS Helper
1363   1163   1  ffff9a2b68864740  IN   5.6 5187156 265988  JS Helper
1364   1163   0  ffff9a2b68865f00  IN   5.6 5187156 265988  JS Helper
1365   1327   2  ffff9a2b68a7af80  IN   1.0 1045344  46496  Xwayland
1385   1163   3  ffff9a2b68a797c0  IN   0.1  232788   5916  xdg-permission-
1386   1163   1  ffff9a2b443adf00  IN   0.1  232788   5916  gmain
1389   1163   2  ffff9a2b68998000  IN   0.5  581408  23344  gnome-shell-cal
1390   1163   3  ffff9a2b4bc84740  IN   0.1  232788   5916  gdbus
1391   1163   1  ffff9a2b4409df00  IN   0.5  581408  23344  gmain
1393   1163   1  ffff9a2c475e5f00  IN   0.5  581408  23344  gdbus
1394   1163   2  ffff9a2b44098000  IN   0.5  581408  23344  dconf worker
1395   1163   2  ffff9a2b4409c740  IN   0.5  581408  23344  gnome-shell-cal
1396   1163   3  ffff9a2b68a7df00  IN   0.5  392816  25336  evolution-sourc
1397   1163   2  ffff9a2b6ba72f80  IN   0.5  392816  25336  gmain
1398   1163   1  ffff9a2b6ba70000  IN   0.5  392816  25336  dconf worker
1399   1163   2  ffff9a2b6ba7df00  IN   0.5  392816  25336  gdbus
1403   1163   1  ffff9a2b6ba74740  IN   0.5  581408  23344  pool-gnome-shel
1404   1163   1  ffff9a2b6bac5f00  IN   0.7  857384  30796  evolution-calen
1405   1163   3  ffff9a2b6bad17c0  IN   0.7  857384  30796  gmain
1406   1163   0  ffff9a2b6ba7c740  IN   0.7  857384  30796  gdbus
1407   1163   3  ffff9a2b6ba7af80  IN   0.7  857384  30796  dconf worker
1408   1163   2  ffff9a2b6ba78000  IN   0.7  857384  30796  evolution-calen
1409   1163   3  ffff9a2b6ba797c0  IN   0.7  857384  30796  pool-evolution-
1412   1163   3  ffff9a2b6bb55f00  IN   0.7  857384  30796  pool-evolution-
1413   1163   3  ffff9a2b68a32f80  IN   0.1  156012   5628  dconf-service
1415   1163   1  ffff9a2b6bb52f80  IN   0.7  857384  30796  pool-evolution-
1416   1163   2  ffff9a2b6bb617c0  IN   0.7  857384  30796  evolution-calen
1417   1163   3  ffff9a2b6bad5f00  IN   0.1  156012   5628  gmain
1418   1163   2  ffff9a2b6bad4740  IN   0.1  156012   5628  gdbus
1419   1163   1  ffff9a2b6bac4740  IN   0.6  741856  29072  evolution-addre
1420   1163   3  ffff9a2b6bb65f00  IN   0.6  741856  29072  gmain
1421   1163   1  ffff9a2b6bb64740  IN   0.6  741856  29072  gdbus
1424   1163   3  ffff9a2b6bad0000  IN   0.2  165668   7364  at-spi2-registr
1425   1163   1  ffff9a2b6bb50000  IN   0.6 2735516  27744  gjs
1427   1163   0  ffff9a2b6bb60000  IN   0.6  741856  29072  dconf worker
1428   1163   1  ffff9a2b70032f80  IN   0.6  741856  29072  evolution-addre
```

```
1429   1163   1  ffff9a2b70038000  IN   0.2  165668   7364  gmain
1431   1163   2  ffff9a2b6bac0000  IN   0.1  306852   6640  gsd-a11y-settin
1432   1163   0  ffff9a2b700397c0  IN   0.2  165668   7364  gdbus
1433   1163   0  ffff9a2b6bac17c0  IN   0.5  450788  25220  gsd-color
1434   1163   1  ffff9a2b700d97c0  IN   0.3  376132  16424  gsd-datetime
1435   1163   0  ffff9a2b700ddf00  IN   0.2  308860   7968  gsd-housekeepin
1438   1163   1  ffff9a2b700dc740  IN   0.5  341900  24528  gsd-keyboard
1439   1163   2  ffff9a2b700daf80  IN   0.6  718144  29708  gsd-media-keys
1440   1163   2  ffff9a2b6bb517c0  IN   0.1  306852   6640  gmain
1443   1163   3  ffff9a2b702817c0  IN   0.1  306852   6640  gdbus
1444   1163   1  ffff9a2b700317c0  IN   0.6  741856  29072  pool-evolution-
1445   1163   2  ffff9a2b700d8000  IN   0.6  450232  27272  gsd-power
1446   1163   3  ffff9a2b70285f00  IN   0.6 2735516  27744  JS Helper
1447   1163   1  ffff9a2b70284740  IN   0.6 2735516  27744  JS Helper
1448   1163   2  ffff9a2b70282f80  IN   0.6 2735516  27744  JS Helper
1449   1163   0  ffff9a2b70280000  IN   0.6 2735516  27744  JS Helper
1450   1163   1  ffff9a2b702cdf00  IN   0.2  320192  10880  gsd-print-notif
1451   1163   2  ffff9a2b702cc740  IN   0.1  454268   6380  gsd-rfkill
1452   1163   1  ffff9a2b702caf80  IN   0.1  232700   5992  gsd-screensaver
1453   1163   3  ffff9a2b702c8000  IN   0.2  308860   7968  gmain
1455   1291   0  ffff9a2b7003af80  IN   1.7  856356  79032  gnome-software
1457   1163   2  ffff9a2b703a4740  IN   0.2  308860   7968  gdbus
1458   1163   1  ffff9a2b70034740  IN   0.3  376132  16424  gmain
1460   1163   1  ffff9a2b71435f00  IN   0.1  232700   5992  gmain
1462   1163   2  ffff9a2b71432f80  IN   0.1  232700   5992  gdbus
1463   1163   2  ffff9a2b71430000  IN   0.3  376132  16424  gdbus
1466   1163   1  ffff9a2b703a2f80  IN   0.2  462196  10512  gsd-sharing
1467   1163   2  ffff9a2b440a17c0  IN   0.1  454268   6380  gmain
1471   1163   0  ffff9a2b703a17c0  IN   0.2  459984  10184  gsd-smartcard
1472   1163   2  ffff9a2b71582f80  IN   0.1  454268   6380  gdbus
1473   1163   0  ffff9a2b703a5f00  IN   0.2  319496  10308  gsd-sound
1474   1163   1  ffff9a2b714a4740  IN   0.2  320192  10880  gmain
1475   1163   2  ffff9a2b714a5f00  IN   0.2  320192  10880  gdbus
1476   1163   2  ffff9a2b714a2f80  IN   0.2  462196  10512  gmain
1478   1163   1  ffff9a2b714a0000  IN   0.2  462196  10512  dconf worker
1479   1163   1  ffff9a2b715d2f80  IN   0.2  455828   7136  gsd-usb-protect
1480   1163   1  ffff9a2b714a17c0  IN   0.2  462196  10512  gdbus
1481   1163   0  ffff9a2b715d0000  IN   0.5  341900  24528  gmain
1483   1163   2  ffff9a2b715d5f00  IN   0.5  342320  22668  gsd-wacom
1484   1163   0  ffff9a2b715d4740  IN   0.2  459984  10184  gmain
1485   1163   1  ffff9a2b70030000  IN   0.2  455828   7136  gmain
1486   1163   1  ffff9a2b714317c0  IN   0.2  455828   7136  gdbus
1487   1291   1  ffff9a2b714ec740  IN   1.5  660528  69492  evolution-alarm
1488   1163   1  ffff9a2b716a17c0  IN   0.2  459984  10184  gdbus
1489   1163   0  ffff9a2b716a5f00  IN   0.2  319496  10308  gmain
1490   1163   1  ffff9a2b716a4740  IN   0.5  341900  24528  dconf worker
1491   1163   1  ffff9a2b716faf80  IN   0.6  450232  27272  gmain
1493   1163   1  ffff9a2b716a2f80  IN   0.5  341900  24528  gdbus
1494   1163   1  ffff9a2b716f97c0  IN   0.6  450232  27272  dconf worker
1495   1163   1  ffff9a2b716a0000  IN   0.2  319496  10308  gdbus
1496   1163   2  ffff9a2b716fdf00  IN   0.6  450232  27272  gdbus
1498   1291   1  ffff9a2b714eaf80  IN   0.2  231792   8860  gsd-disk-utilit
1502   1291   1  ffff9a2b75ce5f00  IN   0.2  231792   8860  gmain
1504   1291   0  ffff9a2b75ce2f80  IN   0.2  231792   8860  gdbus
1505   1163   1  ffff9a2b71580000  IN   0.6 2735516  27744  gmain
1508   1163   0  ffff9a2b7173df00  IN   0.2  308860   7968  dconf worker
1511   1163   1  ffff9a2b715817c0  IN   0.1  306852   6640  dconf worker
1513   1163   0  ffff9a2b75d42f80  IN   0.3  376132  16424  dconf worker
1521   1163   0  ffff9a2b7173af80  IN   0.2  319496  10308  dconf worker
1523   1163   3  ffff9a2b75db8000  IN   0.2  455828   7136  dconf worker
```

```
1525   1163   1  ffff9a2b75d44740  IN   0.3  344808  15080  gsd-printer
1529   1163   1  ffff9a2b71738000  IN   0.2  459984  10184  dconf worker
1530   1163   2  ffff9a2b75dc17c0  IN   0.6 2735516  27744  gdbus
1532   1163   0  ffff9a2b75e0df00  IN   0.2  459984  10184  pool-gsd-smartc
1534   1163   1  ffff9a2b75dbdf00  IN   0.6  718144  29708  gmain
1537   1163   1  ffff9a2b75d45f00  IN   0.6  718144  29708  dconf worker
1548   1163   3  ffff9a2b75c90000  IN   0.5  450788  25220  gmain
1551   1163   1  ffff9a2b75c95f00  IN   0.5  450788  25220  dconf worker
1552   1163   1  ffff9a2b75e65f00  IN   0.3  344808  15080  gmain
1554   1163   2  ffff9a2b75dc5f00  IN   0.5  342320  22668  gmain
1556   1163   1  ffff9a2b75e52f80  IN   0.0   19888   1244  VBoxClient
1557   1163   3  ffff9a2b75e50000  IN   0.6  718144  29708  gdbus
1558   1556   1  ffff9a2b75e517c0  IN   0.1  152024   4376  VBoxClient
1560   1556   0  ffff9a2b75e54740  IN   0.1  152024   4376  RTThrdPP
1567   1163   1  ffff9a2b7003df00  IN   0.5  342320  22668  dconf worker
1568   1556   3  ffff9a2b75e82f80  IN   0.1  152024   4376  SHCLX11
1572   1163   3  ffff9a2b75ed4740  IN   0.3  344808  15080  gdbus
1573   1163   0  ffff9a2b75ed2f80  IN   0.5  450788  25220  gdbus
1576   1163   1  ffff9a2b75f24740  IN   0.5  342320  22668  gdbus
1577   1163   1  ffff9a2b75e80000  IN   0.0   19888   1232  VBoxClient
1578   1577   1  ffff9a2b75f22f80  IN   0.1  152124   3224  VBoxClient
1586   1291   1  ffff9a2b75ed0000  IN   1.7  856356  79032  gmain
1589   1163   1  ffff9a2b75e08000  IN   0.0   19888   1252  VBoxClient
1590   1589   1  ffff9a2b75e0af80  IN   0.1   85904   2436  VBoxDRMClient
1591   1291   0  ffff9a2b75e64740  IN   1.7  856356  79032  gdbus
1592   1163   3  ffff9a2b75f70000  IN   0.0   19888   1248  VBoxClient
1594   1592   2  ffff9a2b75f717c0  IN   0.1  152640   3476  VBoxClient
1595   1291   3  ffff9a2b75e617c0  IN   1.7  856356  79032  dconf worker
1598   1291   1  ffff9a2b7003c740  IN   1.5  660528  69492  gmain
1600   1291   1  ffff9a2b6bb54740  IN   1.5  660528  69492  dconf worker
1601   1291   0  ffff9a2b75e62f80  IN   1.5  660528  69492  gdbus
1612   1327   1  ffff9a2b70035f00  IN   1.0 1045344  46496  llvmpipe-0
1613   1327   2  ffff9a2b75dbc740  IN   1.0 1045344  46496  llvmpipe-1
1614   1327   1  ffff9a2b75f75f00  IN   1.0 1045344  46496  llvmpipe-2
1615   1327   0  ffff9a2b75f72f80  IN   1.0 1045344  46496  llvmpipe-3
1616   1327   3  ffff9a2b714e8000  IN   1.0 1045344  46496  Xwayland
1617   1327   1  ffff9a2b714edf00  IN   1.0 1045344  46496  Xwayland
1618   1327   2  ffff9a2b7898af80  IN   1.0 1045344  46496  Xwayland
1619   1327   0  ffff9a2b78988000  IN   1.0 1045344  46496  Xwayland
1620   1327   3  ffff9a2b789897c0  IN   1.0 1045344  46496  Xwaylan:disk$0
1621   1327   1  ffff9a2b7898df00  IN   1.0 1045344  46496  Xwaylan:disk$1
1622   1327   0  ffff9a2b7898c740  IN   1.0 1045344  46496  Xwaylan:disk$2
1623   1327   2  ffff9a2b789d4740  IN   1.0 1045344  46496  Xwaylan:disk$3
1626   1577   1  ffff9a2b75e84740  IN   0.1  152124   3224  RTThrdPP
1627   1577   3  ffff9a2b789d2f80  IN   0.1  152124   3224  X11 events
1628   1592   1  ffff9a2b71584740  IN   0.1  152640   3476  RTThrdPP
1629   1592   2  ffff9a2b789d0000  IN   0.1  152640   3476  dndHGCM
1630   1592   2  ffff9a2b789d17c0  IN   0.1  152640   3476  dndX11
1633   1327   1  ffff9a2b703a0000  IN   0.3  384788  13280  ibus-daemon
1634   1163   2  ffff9a2b75ce0000  IN   1.3 1366760  62344  gsd-xsettings
1638   1327   0  ffff9a2b75ce17c0  IN   0.3  384788  13280  gmain
1639   1327   1  ffff9a2b75c94740  IN   0.3  384788  13280  gdbus
1644   1633   0  ffff9a2b6899c740  IN   0.2  233724   7212  ibus-dconf
1645      1   2  ffff9a2b6899df00  IN   0.6  376592  26432  fwupd
1646   1633   1  ffff9a2b75c92f80  IN   0.6  345896  26112  ibus-extension-
1651   1633   0  ffff9a2b717a8000  IN   0.2  233724   7212  gmain
1653   1633   1  ffff9a2b717a97c0  IN   0.2  233724   7212  gdbus
1654   1163   2  ffff9a2b717ac740  IN   1.3 1218808  59468  ibus-x11
1658   1633   2  ffff9a2b717adf00  IN   0.6  345896  26112  gmain
1660   1163   0  ffff9a2b75e55f00  IN   0.1  233576   6972  ibus-portal
```

```
1661   1633   0  ffff9a2b75dc0000  IN   0.6  345896  26112  dconf worker
1662   1633   1  ffff9a2b75dc2f80  IN   0.2  233724   7212  dconf worker
1663   1633   1  ffff9a2b716fc740  IN   0.6  345896  26112  gdbus
1664   1163   0  ffff9a2b65f797c0  IN   0.1  233576   6972  gmain
1665   1163   3  ffff9a2b65f7af80  IN   0.1  233576   6972  gdbus
1666   1291   2  ffff9a2b717397c0  IN   1.5  660528  69492  evolution-alarm
1672      1   1  ffff9a2b518faf80  IN   0.6  376592  26432  gmain
1673      1   2  ffff9a2b518fdf00  IN   0.6  376592  26432  libusb_event
1674      1   1  ffff9a2b518f8000  IN   0.6  376592  26432  GUsbEventThread
1677      1   2  ffff9a2b75dbaf80  IN   0.6  376592  26432  gdbus
1678   1163   2  ffff9a2b5aa70000  IN   1.3 1366760  62344  llvmpipe-0
1679   1163   2  ffff9a2b5aa74740  IN   1.3 1366760  62344  llvmpipe-1
1680   1163   2  ffff9a2b5aa717c0  IN   1.3 1366760  62344  llvmpipe-2
1681   1163   2  ffff9a2b5aa75f00  IN   1.3 1366760  62344  llvmpipe-3
1682   1163   2  ffff9a2b4bdc2f80  IN   1.3 1366760  62344  gsd-xsettings
1683   1163   2  ffff9a2b4bdc0000  IN   1.3 1366760  62344  gsd-xsettings
1684   1163   2  ffff9a2b4bdc5f00  IN   1.3 1366760  62344  gsd-xsettings
1685   1163   2  ffff9a2b4bdc17c0  IN   1.3 1366760  62344  gsd-xsettings
1686   1163   2  ffff9a2b4bdc4740  IN   1.3 1366760  62344  gsd-xse:disk$0
1687   1163   2  ffff9a2b4bc82f80  IN   1.3 1366760  62344  gsd-xse:disk$1
1688   1163   2  ffff9a2b4bc817c0  IN   1.3 1366760  62344  gsd-xse:disk$2
1689   1163   2  ffff9a2b5a91c740  IN   1.3 1366760  62344  gsd-xse:disk$3
1690   1163   2  ffff9a2c475e97c0  IN   1.3 1218808  59468  llvmpipe-0
1691   1163   2  ffff9a2b65efdf00  IN   1.3 1218808  59468  llvmpipe-1
1692   1163   2  ffff9a2b65ef8000  IN   1.3 1218808  59468  llvmpipe-2
1693   1163   2  ffff9a2b4bea17c0  IN   1.3 1218808  59468  llvmpipe-3
1694   1163   2  ffff9a2b4bea2f80  IN   1.3 1218808  59468  ibus-x11
1695   1163   2  ffff9a2b4bea0000  IN   1.3 1218808  59468  ibus-x11
1696   1163   2  ffff9a2b4bea4740  IN   1.3 1218808  59468  ibus-x11
1697   1163   2  ffff9a2b6bac2f80  IN   1.3 1218808  59468  ibus-x11
1698   1163   2  ffff9a2b440a4740  IN   1.3 1218808  59468  ibus-x1:disk$0
1699   1163   2  ffff9a2b440a5f00  IN   1.3 1218808  59468  ibus-x1:disk$1
1700   1163   2  ffff9a2b440a0000  IN   1.3 1218808  59468  ibus-x1:disk$2
1701   1163   2  ffff9a2b440a2f80  IN   1.3 1218808  59468  ibus-x1:disk$3
1702   1633   2  ffff9a2b554a8000  IN   0.2  159900   7244  ibus-engine-sim
1703   1163   3  ffff9a2b75e0c740  IN   1.3 1366760  62344  gmain
1704   1163   2  ffff9a2b442c8000  IN   1.3 1366760  62344  gdbus
1705   1633   2  ffff9a2b442caf80  IN   0.2  159900   7244  gmain
1706   1633   2  ffff9a2c583d5f00  IN   0.2  159900   7244  gdbus
1707   1163   0  ffff9a2c583d0000  IN   1.3 1366760  62344  dconf worker
1708   1163   2  ffff9a2b554aaf80  IN   1.3 1218808  59468  gmain
1709   1163   2  ffff9a2b554adf00  IN   1.3 1218808  59468  gdbus
1718   1291   2  ffff9a2b46fdc740  IN   1.5  660528  69492  evolution-alarm
1737   1163   3  ffff9a2b46fe4740  IN   5.6 5187156 265988  pool-gnome-shel
1738   1163   1  ffff9a2b518f97c0  IN   5.6 5187156 265988  pool-gnome-shel
1739   1163   2  ffff9a2b518fc740  IN   5.6 5187156 265988  pool-gnome-shel
1740   1163   3  ffff9a2b789d5f00  IN   5.6 5187156 265988  pool-gnome-shel
1745   1163   3  ffff9a2b46fd97c0  IN   1.3  725172  62280  nautilus
1746   1163   3  ffff9a2b5a904740  IN   1.3  725172  62280  gmain
1747   1163   1  ffff9a2b5a9017c0  IN   1.3  725172  62280  gdbus
1748   1210   1  ffff9a2b68980000  IN   0.2  311012   8580  gvfsd-trash
1749   1210   3  ffff9a2b5aa6af80  IN   0.2  311012   8580  gmain
1750   1210   0  ffff9a2b5aa68000  IN   0.2  311012   8580  gdbus
1753   1163   3  ffff9a2b65d797c0  IN   1.3  725172  62280  pool-org.gnome.
1754   1163   0  ffff9a2b46fe17c0  IN   1.3  725172  62280  dconf worker
1760   1210   3  ffff9a2b46e04740  IN   0.2  310640   7900  gvfsd-burn
1761   1210   1  ffff9a2b5a918000  IN   0.2  310640   7900  gmain
1762   1210   1  ffff9a2b5aa6c740  IN   0.2  310640   7900  gdbus
2077   1291   0  ffff9a2b4bc32f80  IN   1.7  856356  79032  pool-org.gnome.
2078   1291   2  ffff9a2b4bc317c0  IN   1.7  856356  79032  pool-org.gnome.
```

```
   2079   1291   1  ffff9a2b4bc30000  IN   1.7  856356  79032  pool-org.gnome.
   2080   1291   0  ffff9a2b4bc35f00  IN   1.7  856356  79032  pool-org.gnome.
   2087   1163   2  ffff9a2b5a99c740  IN   0.1  159328   6204  gvfsd-metadata
   2088   1163   0  ffff9a2c5bdf5f00  IN   0.1  159328   6204  gmain
   2089   1163   0  ffff9a2c5bdf2f80  IN   0.1  159328   6204  gdbus
   2100   1163   2  ffff9a2b5a99af80  IN   0.9  400740  43628  gnome-terminal-
   2101   1163   0  ffff9a2b46fe5f00  IN   0.9  400740  43628  gmain
   2103   1163   2  ffff9a2b46fe0000  IN   0.9  400740  43628  dconf worker
   2104   1163   1  ffff9a2c5bdf4740  IN   0.9  400740  43628  gdbus
   2105   2100   1  ffff9a2b5a9997c0  IN   0.1    8116   4900  bash
   2124   1163   2  ffff9a2b4bc34740  IN   5.6 5187156 265988  threaded-ml
   2130    259   1  ffff9a2c45924740  IN   0.1   23408   4236  systemd-udevd
   2131    259   3  ffff9a2c45925f00  IN   0.1   23408   4236  systemd-udevd
   2134   2105   0  ffff9a2b46fd8000  IN   0.1   10644   5192  sudo
>  2135   2134   3  ffff9a2c45920000  RU   0.0    5304   1800  tee
```

8. List CPU queues:

```
crash> runq
CPU 0 RUNQUEUE: ffff9a2c5bc2fcc0
  CURRENT: PID: 0      TASK: ffffffff91213940  COMMAND: "swapper/0"
  RT PRIO_ARRAY: ffff9a2c5bc2ff00
     [no tasks queued]
  CFS RB_ROOT: ffff9a2c5bc2fd70
     [no tasks queued]

CPU 1 RUNQUEUE: ffff9a2c5bcafcc0
  CURRENT: PID: 0      TASK: ffff9a2c4024df00  COMMAND: "swapper/1"
  RT PRIO_ARRAY: ffff9a2c5bcaff00
     [no tasks queued]
  CFS RB_ROOT: ffff9a2c5bcafd70
     [no tasks queued]

CPU 2 RUNQUEUE: ffff9a2c5bd2fcc0
  CURRENT: PID: 0      TASK: ffff9a2c402697c0  COMMAND: "swapper/2"
  RT PRIO_ARRAY: ffff9a2c5bd2ff00
     [no tasks queued]
  CFS RB_ROOT: ffff9a2c5bd2fd70
     [no tasks queued]

CPU 3 RUNQUEUE: ffff9a2c5bdafcc0
  CURRENT: PID: 2135   TASK: ffff9a2c45920000  COMMAND: "tee"
  RT PRIO_ARRAY: ffff9a2c5bdaff00
     [no tasks queued]
  CFS RB_ROOT: ffff9a2c5bdafd70
     [no tasks queued]
```

9. Set the current task to PID 2134 and then to the task running to CPU 1, and then to the panicked task:

```
crash> set 2134
    PID: 2134
COMMAND: "sudo"
   TASK: ffff9a2b46fd8000  [THREAD_INFO: ffff9a2b46fd8000]
    CPU: 0
  STATE: TASK_INTERRUPTIBLE
```

```
crash> set -c 1
    PID: 0
COMMAND: "swapper/1"
   TASK: ffff9a2c4024df00  (1 of 4)  [THREAD_INFO: ffff9a2c4024df00]
    CPU: 1
  STATE: TASK_RUNNING (ACTIVE)

crash> set -p
    PID: 2135
COMMAND: "tee"
   TASK: ffff9a2c45920000  [THREAD_INFO: ffff9a2c45920000]
    CPU: 3
  STATE: TASK_RUNNING (PANIC)
```

10. Display the stack trace of the bash process without and with source code, and dump raw stack data:

```
crash> set 2105
    PID: 2105
COMMAND: "bash"
   TASK: ffff9a2b5a9997c0  [THREAD_INFO: ffff9a2b5a9997c0]
    CPU: 1
  STATE: TASK_INTERRUPTIBLE #0 [ffffa77fc1f0bdc8] __schedule at ffffffff904c0112

crash> bt
PID: 2105   TASK: ffff9a2b5a9997c0  CPU: 1   COMMAND: "bash"
#1 [ffffa77fc1f0be58] schedule at ffffffff904c0746
 #2 [ffffa77fc1f0be70] do_wait at ffffffff8fc8bd7f
 #3 [ffffa77fc1f0beb0] kernel_wait4 at ffffffff8fc8d1d6
 #4 [ffffa77fc1f0bf40] do_syscall_64 at ffffffff904b3883
 #5 [ffffa77fc1f0bf50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007fb46aa3c1c6  RSP: 00007fff03321608  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 000000000000000a  RCX: 00007fb46aa3c1c6
    RDX: 000000000000000a  RSI: 00007fff03321620  RDI: 00000000ffffffff
    RBP: 0000000000000000   R8: 0000000000000000   R9: 0000000000000000
    R10: 0000000000000000  R11: 0000000000000246  R12: 0000000000000000
    R13: 0000000000000000  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: 000000000000003d  CS: 0033  SS: 002b

crash> bt -l
PID: 2105   TASK: ffff9a2b5a9997c0  CPU: 1   COMMAND: "bash"
 #0 [ffffa77fc1f0bdc8] __schedule at ffffffff904c0112
    debian/build/build_amd64_none_amd64/kernel/sched/core.c: 3791
 #1 [ffffa77fc1f0be58] schedule at ffffffff904c0746
    debian/build/build_amd64_none_amd64/arch/x86/include/asm/bitops.h: 206
 #2 [ffffa77fc1f0be70] do_wait at ffffffff8fc8bd7f
    debian/build/build_amd64_none_amd64/kernel/exit.c: 1473
 #3 [ffffa77fc1f0beb0] kernel_wait4 at ffffffff8fc8d1d6
    debian/build/build_amd64_none_amd64/kernel/exit.c: 1617
 #4 [ffffa77fc1f0bf40] do_syscall_64 at ffffffff904b3883
    debian/build/build_amd64_none_amd64/arch/x86/entry/common.c: 46
 #5 [ffffa77fc1f0bf50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    /build/linux-3cXDux/linux-5.10.84/arch/x86/entry/entry_64.S: 127
    RIP: 00007fb46aa3c1c6  RSP: 00007fff03321608  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 000000000000000a  RCX: 00007fb46aa3c1c6
    RDX: 000000000000000a  RSI: 00007fff03321620  RDI: 00000000ffffffff
    RBP: 0000000000000000   R8: 0000000000000000   R9: 0000000000000000
    R10: 0000000000000000  R11: 0000000000000246  R12: 0000000000000000
    R13: 0000000000000000  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: 000000000000003d  CS: 0033  SS: 002b
```

```
crash> bt -r
PID: 2105   TASK: ffff9a2b5a9997c0  CPU: 1   COMMAND: "bash"
ffffa77fc1f08000:  0000000057ac6e9d 0000000000000000
ffffa77fc1f08010:  0000000000000000 0000000000000000
ffffa77fc1f08020:  0000000000000000 0000000000000000
ffffa77fc1f08030:  0000000000000000 0000000000000000
ffffa77fc1f08040:  0000000000000000 0000000000000000
[...]
ffffa77fc1f0b400:  0000000000000000 0000000000000000
ffffa77fc1f0b410:  0000000000000000 0000000000000000
ffffa77fc1f0b420:  0000000000000000 0000000000000000
ffffa77fc1f0b430:  0000000000000000 0000000000000000
ffffa77fc1f0b440:  0000000000000000 0000000000000000
ffffa77fc1f0b450:  0000000000000000 0000000000000000
ffffa77fc1f0b460:  0000000000000000 0000000000000000
ffffa77fc1f0b470:  0000000000000000 0000000000000000
ffffa77fc1f0b480:  0000000000000000 0000000000000000
ffffa77fc1f0b490:  0000000000000000 0000000000000000
ffffa77fc1f0b4a0:  0000000000000000 0000000000000000
ffffa77fc1f0b4b0:  0000000000000000 0000000000000000
ffffa77fc1f0b4c0:  ffff9a2c5bdafcc0 ffff9a2c5bdafcc0
ffffa77fc1f0b4d0:  kvm_sched_clock_read+13 0000000000000001
ffffa77fc1f0b4e0:  ffff9a2c5bdafe80 0000000000000018
ffffa77fc1f0b4f0:  ffff9a2c5bdb0610 ffff9a2c5bdafd40
ffffa77fc1f0b500:  ffff9a2c5bdb04d0 update_blocked_averages+512
ffffa77fc1f0b510:  0000000000000000 0000000000000000
ffffa77fc1f0b520:  ffff9a2c5bdafcc0 0100000000000000
ffffa77fc1f0b530:  0000000000000086 ffffa77fc1f0b5b0
ffffa77fc1f0b540:  ffff9a2c5bdafcc0 000000000002fcc0
ffffa77fc1f0b550:  000000000002fcc0 0000000000000003
ffffa77fc1f0b560:  cpumask_next_and+26 update_sd_lb_stats.constprop.0+2068
ffffa77fc1f0b570:  ffff9a2c402c0f60 ffff9a2c402c0880
ffffa77fc1f0b580:  0000000000000003 ffffa77fc1f0b640
ffffa77fc1f0b590:  0000000300000000 ffffa77fc1f0b6c0
ffffa77fc1f0b5a0:  0000000000000000 0000000700000007
ffffa77fc1f0b5b0:  0000000000000000 0000000000000037
ffffa77fc1f0b5c0:  0000000000000400 0000000000000034
ffffa77fc1f0b5d0:  0000000000000037 0000000100000001
ffffa77fc1f0b5e0:  0000000100000000 0000000000000000
ffffa77fc1f0b5f0:  0000000000000000 0000000000000000
ffffa77fc1f0b600:  695bd83cf55d9800 ffffa77fc1f0b788
ffffa77fc1f0b610:  ffff9a2c402c0880 00000000ffffffff
ffffa77fc1f0b620:  ffff9a2c40300a00 0000000000000000
ffffa77fc1f0b630:  ffffa77fc1f0b7e0 find_busiest_group+65
ffffa77fc1f0b640:  ffff9a2c402c0f40 ffff9a2c402c0880
ffffa77fc1f0b650:  000000000000009d 0000000000001000
ffffa77fc1f0b660:  0000000000000000 0000000000000000
ffffa77fc1f0b670:  0000000000000000 0000000000000037
ffffa77fc1f0b680:  0000000000000400 0000000000000034
ffffa77fc1f0b690:  0000000000000037 0000000100000001
ffffa77fc1f0b6a0:  0000000100000000 0000000000000000
ffffa77fc1f0b6b0:  0000000000000000 0000000000000000
ffffa77fc1f0b6c0:  ffff9a2c5bcafd40 0000003ec3fa8c72
ffffa77fc1f0b6d0:  0000000000000001 0000000000000000
ffffa77fc1f0b6e0:  ffff9a2b5a999840 ffff9a2c5bc9c980
ffffa77fc1f0b6f0:  0000000000000000 kvm_sched_clock_read+13
ffffa77fc1f0b700:  sched_clock+5    sched_clock_cpu+12
ffffa77fc1f0b710:  ffff9a2c5bc9c980 0000000000000000
ffffa77fc1f0b720:  record_times+21  ffff9a2c5bc9c980
ffffa77fc1f0b730:  psi_system       0000000000000000
```

```
ffffa77fc1f0b740:  psi_group_change+65 000000015bcafd40
ffffa77fc1f0b750:  ffff9a2c00000001 psi_system
ffffa77fc1f0b760:  psi_system       ffff9a2b5a9997c0
ffffa77fc1f0b770:  ffff9a2c4031af80 0000000000000001
ffffa77fc1f0b780:  0000000000000000 finish_task_switch+114
ffffa77fc1f0b790:  fair_sched_class ffff9a2c4031af80
ffffa77fc1f0b7a0:  ffff9a2c5bcafcc0 0000000000000096
ffffa77fc1f0b7b0:  ffff9a2c5bcafcc0 ffff9a2c5bcafd40
ffffa77fc1f0b7c0:  0000003f31598869 0000000000000001
ffffa77fc1f0b7d0:  0000000000000000 ffff9a2b5a999840
ffffa77fc1f0b7e0:  ffff9a2c5bc9c980 0000000000000000
ffffa77fc1f0b7f0:  kvm_sched_clock_read+13 sched_clock+5
ffffa77fc1f0b800:  sched_clock_cpu+12 ffff9a2c5bc9c980
ffffa77fc1f0b810:  0000000000000000 record_times+21
ffffa77fc1f0b820:  ffff9a2c5bc9c980 psi_system
ffffa77fc1f0b830:  0000000000000000 psi_group_change+65
ffffa77fc1f0b840:  000000015bcafd40 ffff9a2c00000001
ffffa77fc1f0b850:  psi_system       psi_system
ffffa77fc1f0b860:  ffff9a2b5a9997c0 ffff9a2c40342f80
ffffa77fc1f0b870:  0000000000000001 0000000000000000
ffffa77fc1f0b880:  finish_task_switch+114 fair_sched_class
ffffa77fc1f0b890:  ffff9a2c40342f80 xas_load+5
ffffa77fc1f0b8a0:  find_get_entry+209 ffff9a2c404362b8
ffffa77fc1f0b8b0:  0000000000582021 ffffffff00210000
ffffa77fc1f0b8c0:  ffff9a2c472f0b68 ffff9a2b4bec87d0
ffffa77fc1f0b8d0:  ffffa77fc1f0b9b0 ffffc9ac002fb200
ffffa77fc1f0b8e0:  ffffc9ac00a24588 get_partial_node+266
ffffa77fc1f0b8f0:  0000000000582230 ffff9a2c5bcb4510
ffffa77fc1f0b900:  ffff9a2c40041140 ffff9a2c4019cd00
ffffa77fc1f0b910:  0000000000000000 ffff9a2b75709000
ffffa77fc1f0b920:  0000001300000cc0 ffff9a2c40041150
ffffa77fc1f0b930:  ffff9a2b5a9997c0 0000000000000287
ffffa77fc1f0b940:  00000020404362b0 0000000000000287
ffffa77fc1f0b950:  000000205a9997c0 0000000000000001
ffffa77fc1f0b960:  ffff9a2b5a9997c0 0000012c00000010
ffffa77fc1f0b970:  0000000000001000 695bd83cf55d9800
ffffa77fc1f0b980:  0000000000000000 0000000000000002
ffffa77fc1f0b990:  kernel_init_free_pages+70 prep_new_page+167
ffffa77fc1f0b9a0:  0000000000000000 0000000000000004
ffffa77fc1f0b9b0:  695bd83cf55d9800 00000000000000d0
ffffa77fc1f0b9c0:  0000000000000001 0000000000000000
ffffa77fc1f0b9d0:  ffff9a2b55665000 0000000000000cc0
ffffa77fc1f0b9e0:  ffff9a2b757090c8 __memcg_kmem_charge+49
ffffa77fc1f0b9f0:  0000000000000287 695bd83cf55d9800
ffffa77fc1f0ba00:  0000000000000206 00000000000000d0
ffffa77fc1f0ba10:  ffff9a2b78969ac0 0000000000000000
ffffa77fc1f0ba20:  __mod_memcg_lruvec_state+33 0000000000000000
ffffa77fc1f0ba30:  ffff9a2b78969ac0 0000000000000246
ffffa77fc1f0ba40:  memcg_slab_post_alloc_hook+392 0000000000000001
ffffa77fc1f0ba50:  00000cc078969ac0 ffff9a2b75709e10
ffffa77fc1f0ba60:  0000000000000cc0 ffff9a2c4019cd00
ffffa77fc1f0ba70:  vm_area_dup+33   ffff9a2c4019cd00
ffffa77fc1f0ba80:  ffff9a2b75709bb8 kmem_cache_alloc+237
ffffa77fc1f0ba90:  ffff9a2b75709c80 ffff9a2b78969ac0
ffffa77fc1f0baa0:  695bd83cf55d9800 ffff9a2b441b4bb8
ffffa77fc1f0bab0:  ffff9a2b62d60cc0 ffff9a2b75c89dc0
ffffa77fc1f0bac0:  0000000000000000 ffff9a2b75709be0
ffffa77fc1f0bad0:  vm_area_dup+33   00007fff033f2000
ffffa77fc1f0bae0:  00007fff033f4000 0000000000000000
ffffa77fc1f0baf0:  ffff9a2b441b4000 ffff9a2b441b4020
```

```
ffffa77fc1f0bb00:  0000000000000000 0000000000000000
ffffa77fc1f0bb10:  0000000000000000 ffff9a2b62d60cc0
ffffa77fc1f0bb20:  0000000000000025 0000000008040075
ffffa77fc1f0bb30:  0000000000000000 0000000000000000
ffffa77fc1f0bb40:  0000000000000000 0000000000000000
ffffa77fc1f0bb50:  ffff9a2b441b4c30 ffff9a2b441b4c30
ffffa77fc1f0bb60:  0000000000000000 special_mapping_vmops
ffffa77fc1f0bb70:  kernel_init_free_pages+70 prep_new_page+167
ffffa77fc1f0bb80:  0000000000000000 0000000000000004
ffffa77fc1f0bb90:  000000000000487e 0000000000000003
ffffa77fc1f0bba0:  ffff9a2c5ffd25c0 get_page_from_freelist+4301
ffffa77fc1f0bbb0:  0000000000000000 ffff9a2b441b4000
ffffa77fc1f0bbc0:  ffff9a2b441b4020 0000000000000010
ffffa77fc1f0bbd0:  0000000000000000 ffffc9ac00a613c0
ffffa77fc1f0bbe0:  ffffa77fc1f0bcf0 ffff9a2c5ffd3cd0
ffffa77fc1f0bbf0:  0000090100000000 0000000000000001
ffffa77fc1f0bc00:  ffff9a2c5bcb4290 ffff9a2c5bcb42a0
ffffa77fc1f0bc10:  0000000000000000 cpumask_next+23
ffffa77fc1f0bc20:  ffff9a2c5ffd26b0 ffff9a2c5bcb4280
ffffa77fc1f0bc30:  0000000000000000 0000000000000000
ffffa77fc1f0bc40:  00000000000000f0 00000000000000c0
ffffa77fc1f0bc50:  ffff9a2c00000100 ffff9a2c5ffd2680
ffffa77fc1f0bc60:  ffff9a2c5ffd2b80 ffff9a2c402c03c0
ffffa77fc1f0bc70:  0000000000000246 00100cca00000001
ffffa77fc1f0bc80:  ffff9a2c5bcacd00 0000000900000001
ffffa77fc1f0bc90:  0000000000000000 ffffffffffffffff
ffffa77fc1f0bca0:  0000000000000001 0000000000000287
ffffa77fc1f0bcb0:  00000020f55d9800 0000000000000001
ffffa77fc1f0bcc0:  ffff9a2b5a9997c0 0000010000000010
ffffa77fc1f0bcd0:  0000000000001000 0000000000000006
ffffa77fc1f0bce0:  ffff9a2c5bcad5e0 0000000000000001
ffffa77fc1f0bcf0:  ffff9a2b78969ac0 ffff9a2c5bcafd40
ffffa77fc1f0bd00:  00000040467bb919 0000000000000001
ffffa77fc1f0bd10:  0000000000000000 ffff9a2b5a999840
ffffa77fc1f0bd20:  ffff9a2b5a999840 update_load_avg+122
ffffa77fc1f0bd30:  0000000000000009 ffff9a2c5bcafd40
ffffa77fc1f0bd40:  ffff9a2b5a999840 ffff9a2b5a9997c0
ffffa77fc1f0bd50:  0000000000000009 ffff9a2b5a999840
ffffa77fc1f0bd60:  dequeue_entity+198 newidle_balance+642
ffffa77fc1f0bd70:  ffffa77fc1f0be10 0000000000000000
ffffa77fc1f0bd80:  ffff9a2c5bcafcc0 695bd83cf55d9800
ffffa77fc1f0bd90:  ffff9a2c5bcafcc0 ffffa77fc1f0be50
ffffa77fc1f0bda0:  ffffa77fc1f0be10 ffff9a2b5a9997c0
ffffa77fc1f0bdb0:  ffff9a2c5bcafd40 ffff9a2c5bcafcc0
ffffa77fc1f0bdc0:  pick_next_task_fair+57 ffff9a2c5bcafcc0
ffffa77fc1f0bdd0:  ffff9a2b5a9997c0 ffff9a2c5bcafcc0
ffffa77fc1f0bde0:  ffff9a2c4024df00 fair_sched_class
ffffa77fc1f0bdf0:  ffffa77fc1f0be50 __schedule+642
ffffa77fc1f0be00:  ffff9a2b5a99a190 00000000000003e8
ffffa77fc1f0be10:  wait_consider_task+2503 ffff9a2c00000004
ffffa77fc1f0be20:  695bd83cf55d9800 ffff9a2b5a9997c0
ffffa77fc1f0be30:  ffff9a2b5a9997c0 ffff9a2b5a9997c0
ffffa77fc1f0be40:  ffffa77fc1f0bee0 ffff9a2b5a9997b0
ffffa77fc1f0be50:  ffff9a2b5a9997c0 schedule+70
ffffa77fc1f0be60:  ffffa77fc1f0beb8 ffff9a2b5a99a0c0
ffffa77fc1f0be70:  do_wait+431      0000000000000000
ffffa77fc1f0be80:  000000000000000e 00007fff03321620
ffffa77fc1f0be90:  0000000000000000 0000000000000000
ffffa77fc1f0bea0:  0000000000000004 0000000000000000
ffffa77fc1f0beb0:  kernel_wait4+166 0000000e00000004
```

```
ffffa77fc1f0bec0:  0000000000000000 0000000000000000
ffffa77fc1f0bed0:  0000000000000000 0000000000000000
ffffa77fc1f0bee0:  ffff9a2b00000000 ffff9a2b5a9997c0
ffffa77fc1f0bef0:  child_wait_callback ffff9a2c418a5ea8
ffffa77fc1f0bf00:  ffff9a2c418a5ea8 0000000000000000
ffffa77fc1f0bf10:  695bd83cf55d9800 0000000000000000
ffffa77fc1f0bf20:  ffffa77fc1f0bf58 0000000000000000
ffffa77fc1f0bf30:  0000000000000000 0000000000000000
ffffa77fc1f0bf40:  do_syscall_64+51 0000000000000000
ffffa77fc1f0bf50:  entry_SYSCALL_64_after_hwframe+68 0000000000000000
ffffa77fc1f0bf60:  0000000000000000 0000000000000000
ffffa77fc1f0bf70:  0000000000000000 0000000000000000
ffffa77fc1f0bf80:  000000000000000a 0000000000000246
ffffa77fc1f0bf90:  0000000000000000 0000000000000000
ffffa77fc1f0bfa0:  0000000000000000 ffffffffffffffda
ffffa77fc1f0bfb0:  00007fb46aa3c1c6 000000000000000a
ffffa77fc1f0bfc0:  00007fff03321620 00000000ffffffff
ffffa77fc1f0bfd0:  000000000000003d 00007fb46aa3c1c6
ffffa77fc1f0bfe0:  0000000000000033 0000000000000246
ffffa77fc1f0bff0:  00007fff03321608 000000000000002b

crash> bt -f
PID: 2105   TASK: ffff9a2b5a9997c0  CPU: 1   COMMAND: "bash"
 #0 [ffffa77fc1f0bdc8] __schedule at ffffffff904c0112
    ffffa77fc1f0bdd0: ffff9a2b5a9997c0 ffff9a2c5bcafcc0
    ffffa77fc1f0bde0: ffff9a2c4024df00 ffffffff90d74c60
    ffffa77fc1f0bdf0: ffffa77fc1f0be50 ffffffff904c0112
    ffffa77fc1f0be00: ffff9a2b5a99a190 00000000000003e8
    ffffa77fc1f0be10: ffffffff8fc8bb17 ffff9a2c00000004
    ffffa77fc1f0be20: 695bd83cf55d9800 ffff9a2b5a9997c0
    ffffa77fc1f0be30: ffff9a2b5a9997c0 ffff9a2b5a9997c0
    ffffa77fc1f0be40: ffffa77fc1f0bee0 ffff9a2b5a9997b0
    ffffa77fc1f0be50: ffff9a2b5a9997c0 ffffffff904c0746
 #1 [ffffa77fc1f0be58] schedule at ffffffff904c0746
    ffffa77fc1f0be60: ffffa77fc1f0beb8 ffff9a2b5a99a0c0
    ffffa77fc1f0be70: ffffffff8fc8bd7f
 #2 [ffffa77fc1f0be70] do_wait at ffffffff8fc8bd7f
    ffffa77fc1f0be78: 0000000000000000 000000000000000e
    ffffa77fc1f0be88: 00007fff03321620 0000000000000000
    ffffa77fc1f0be98: 0000000000000000 0000000000000004
    ffffa77fc1f0bea8: 0000000000000000 ffffffff8fc8d1d6
 #3 [ffffa77fc1f0beb0] kernel_wait4 at ffffffff8fc8d1d6
    ffffa77fc1f0beb8: 0000000e00000004 0000000000000000
    ffffa77fc1f0bec8: 0000000000000000 0000000000000000
    ffffa77fc1f0bed8: 0000000000000000 ffff9a2b00000000
    ffffa77fc1f0bee8: ffff9a2b5a9997c0 ffffffff8fc8ab70
    ffffa77fc1f0bef8: ffff9a2c418a5ea8 ffff9a2c418a5ea8
    ffffa77fc1f0bf08: 0000000000000000 695bd83cf55d9800
    ffffa77fc1f0bf18: 0000000000000000 ffffa77fc1f0bf58
    ffffa77fc1f0bf28: 0000000000000000 0000000000000000
    ffffa77fc1f0bf38: 0000000000000000 ffffffff904b3883
 #4 [ffffa77fc1f0bf40] do_syscall_64 at ffffffff904b3883
    ffffa77fc1f0bf48: 0000000000000000 ffffffff9060008c
 #5 [ffffa77fc1f0bf50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007fb46aa3c1c6  RSP: 00007fff03321608  RFLAGS: 00000246
```

11. Show virtual memory layout for the current process context:

```
crash> vm
PID: 2105   TASK: ffff9a2b5a9997c0  CPU: 1   COMMAND: "bash"
       MM               PGD          RSS    TOTAL_VM
ffff9a2b62d60cc0  ffff9a2b5a868000  4900k    8116k
      VMA           START       END     FLAGS FILE
ffff9a2c5044a7d0 5621cf7ca000 5621cf7f8000 8000871 /usr/bin/bash
ffff9a2c5044abb8 5621cf7f8000 5621cf8b3000 8000875 /usr/bin/bash
ffff9a2c5044a708 5621cf8b3000 5621cf8eb000 8000871 /usr/bin/bash
ffff9a2c5044a898 5621cf8ec000 5621cf8ef000 8100871 /usr/bin/bash
ffff9a2b440b5898 5621cf8ef000 5621cf8f8000 8100873 /usr/bin/bash
ffff9a2c5044a258 5621cf8f8000 5621cf903000 8100073
ffff9a2b440b50c8 5621cfcc7000 5621cfe28000 8100073
ffff9a2b440b5d48 7fb46a65e000 7fb46a661000 8000071 /usr/lib/x86_64-linux-gnu/libnss_files-2.31.so
ffff9a2b440b53e8 7fb46a661000 7fb46a668000 8000075 /usr/lib/x86_64-linux-gnu/libnss_files-2.31.so
ffff9a2b440b5258 7fb46a668000 7fb46a66a000 8000071 /usr/lib/x86_64-linux-gnu/libnss_files-2.31.so
ffff9a2b440b5000 7fb46a66a000 7fb46a66b000 8100071 /usr/lib/x86_64-linux-gnu/libnss_files-2.31.so
ffff9a2b440b5708 7fb46a66b000 7fb46a66c000 8100073 /usr/lib/x86_64-linux-gnu/libnss_files-2.31.so
ffff9a2b440b5bb8 7fb46a66c000 7fb46a672000 8100073
ffff9a2b440b5190 7fb46a687000 7fb46a96e000 8000071 /usr/lib/locale/locale-archive
ffff9a2b440b5320 7fb46a96e000 7fb46a971000 8100073
ffff9a2b441b40c8 7fb46a971000 7fb46a996000 8000071 /usr/lib/x86_64-linux-gnu/libc-2.31.so
ffff9a2b441b44b0 7fb46a996000 7fb46aae1000 8000075 /usr/lib/x86_64-linux-gnu/libc-2.31.so
ffff9a2b441b4e10 7fb46aae1000 7fb46ab2b000 8000071 /usr/lib/x86_64-linux-gnu/libc-2.31.so
ffff9a2b441b4c80 7fb46ab2b000 7fb46ab2c000 8000070 /usr/lib/x86_64-linux-gnu/libc-2.31.so
ffff9a2b440b5578 7fb46ab2c000 7fb46ab2f000 8100071 /usr/lib/x86_64-linux-gnu/libc-2.31.so
ffff9a2b440b57d0 7fb46ab2f000 7fb46ab32000 8100073 /usr/lib/x86_64-linux-gnu/libc-2.31.so
ffff9a2b440b5960 7fb46ab32000 7fb46ab36000 8100073
ffff9a2b441b47d0 7fb46ab36000 7fb46ab37000 8000071 /usr/lib/x86_64-linux-gnu/libdl-2.31.so
ffff9a2b441b4ed8 7fb46ab37000 7fb46ab39000 8000075 /usr/lib/x86_64-linux-gnu/libdl-2.31.so
ffff9a2b441b4a28 7fb46ab39000 7fb46ab3a000 8000071 /usr/lib/x86_64-linux-gnu/libdl-2.31.so
ffff9a2b441b4898 7fb46ab3a000 7fb46ab3b000 8100071 /usr/lib/x86_64-linux-gnu/libdl-2.31.so
ffff9a2b440b5a28 7fb46ab3b000 7fb46ab3c000 8100073 /usr/lib/x86_64-linux-gnu/libdl-2.31.so
ffff9a2b441b4640 7fb46ab3c000 7fb46ab4a000 8000071 /usr/lib/x86_64-linux-gnu/libtinfo.so.6.2
ffff9a2b441b4960 7fb46ab4a000 7fb46ab58000 8000075 /usr/lib/x86_64-linux-gnu/libtinfo.so.6.2
ffff9a2b441b4578 7fb46ab58000 7fb46ab66000 8000071 /usr/lib/x86_64-linux-gnu/libtinfo.so.6.2
ffff9a2b441b4320 7fb46ab66000 7fb46ab6a000 8100071 /usr/lib/x86_64-linux-gnu/libtinfo.so.6.2
ffff9a2b440b5ed8 7fb46ab6a000 7fb46ab6b000 8100073 /usr/lib/x86_64-linux-gnu/libtinfo.so.6.2
ffff9a2b441b4af0 7fb46ab6b000 7fb46ab6d000 8100073
ffff9a2b440b54b0 7fb46ab7b000 7fb46ab82000 80000d1 /usr/lib/x86_64-linux-gnu/gconv/gconv-modules.cache
ffff9a2c5044a320 7fb46ab82000 7fb46ab83000 8000871 /usr/lib/x86_64-linux-gnu/ld-2.31.so
ffff9a2c5044a0c8 7fb46ab83000 7fb46aba3000 8000875 /usr/lib/x86_64-linux-gnu/ld-2.31.so
ffff9a2b441b4d48 7fb46aba3000 7fb46abab000 8000871 /usr/lib/x86_64-linux-gnu/ld-2.31.so
ffff9a2b441b4258 7fb46abac000 7fb46abad000 8100871 /usr/lib/x86_64-linux-gnu/ld-2.31.so
ffff9a2b440b5c80 7fb46abad000 7fb46abae000 8100873 /usr/lib/x86_64-linux-gnu/ld-2.31.so
ffff9a2b441b43e8 7fb46abae000 7fb46abaf000 8100073
ffff9a2b75c43640 7fff03303000 7fff03324000 100173
ffff9a2b441b4000 7fff033ee000 7fff033f2000 c044411
ffff9a2b441b4bb8 7fff033f2000 7fff033f4000 8040075
```

12. List opened files for the current process context:

```
crash> files
PID: 2105   TASK: ffff9a2b5a9997c0  CPU: 1   COMMAND: "bash"
ROOT: /    CWD: /home/coredump
 FD       FILE             DENTRY           INODE       TYPE PATH
  0 ffff9a2b46f00000 ffff9a2b7845df00 ffff9a2b75379a20 CHR  /dev/pts/0
  1 ffff9a2b46f00000 ffff9a2b7845df00 ffff9a2b75379a20 CHR  /dev/pts/0
  2 ffff9a2b46f00000 ffff9a2b7845df00 ffff9a2b75379a20 CHR  /dev/pts/0
255 ffff9a2b46f00000 ffff9a2b7845df00 ffff9a2b75379a20 CHR  /dev/pts/0
```

13. Dump memory contents as pointers without and with symbolic information:

```
crash> bt
PID: 2105   TASK: ffff9a2b5a9997c0  CPU: 1   COMMAND: "bash"
 #0 [ffffa77fc1f0bdc8] __schedule at ffffffff904c0112
 #1 [ffffa77fc1f0be58] schedule at ffffffff904c0746
 #2 [ffffa77fc1f0be70] do_wait at ffffffff8fc8bd7f
 #3 [ffffa77fc1f0beb0] kernel_wait4 at ffffffff8fc8d1d6
 #4 [ffffa77fc1f0bf40] do_syscall_64 at ffffffff904b3883
 #5 [ffffa77fc1f0bf50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007fb46aa3c1c6  RSP: 00007fff03321608  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 000000000000000a  RCX: 00007fb46aa3c1c6
    RDX: 000000000000000a  RSI: 00007fff03321620  RDI: 00000000ffffffff
    RBP: 0000000000000000   R8: 0000000000000000   R9: 0000000000000000
    R10: 0000000000000000  R11: 0000000000000246  R12: 0000000000000000
    R13: 0000000000000000  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: 000000000000003d  CS: 0033  SS: 002b

crash> rd -64 ffffa77fc1f0bdc8 50
ffffa77fc1f0bdc8:  ffff9a2c5bcafcc0 ffff9a2b5a9997c0   ...[,......Z+...
ffffa77fc1f0bdd8:  ffff9a2c5bcafcc0 ffff9a2c4024df00   ...[,.....$@,...
ffffa77fc1f0bde8:  ffffffff90d74c60 ffffa77fc1f0be50   `L......P.......
ffffa77fc1f0bdf8:  ffffffff904c0112 ffff9a2b5a99a190   ..L........Z+...
ffffa77fc1f0be08:  00000000000003e8 ffffffff8fc8bb17   ................
ffffa77fc1f0be18:  ffff9a2c00000004 695bd83cf55d9800   ....,.....].<.[i
ffffa77fc1f0be28:  ffff9a2b5a9997c0 ffff9a2b5a9997c0   ...Z+......Z+...
ffffa77fc1f0be38:  ffff9a2b5a9997c0 ffffa77fc1f0bee0   ...Z+...........
ffffa77fc1f0be48:  ffff9a2b5a9997b0 ffff9a2b5a9997c0   ...Z+......Z+...
ffffa77fc1f0be58:  ffffffff904c0746 ffffa77fc1f0beb8   F.L.............
ffffa77fc1f0be68:  ffff9a2b5a99a0c0 ffffffff8fc8bd7f   ...Z+...........
ffffa77fc1f0be78:  0000000000000000 000000000000000e   ................
ffffa77fc1f0be88:  00007fff03321620 0000000000000000    .2.............
ffffa77fc1f0be98:  0000000000000000 0000000000000004   ................
ffffa77fc1f0bea8:  0000000000000000 ffffffff8fc8d1d6   ................
ffffa77fc1f0beb8:  0000000e00000004 0000000000000000   ................
ffffa77fc1f0bec8:  0000000000000000 0000000000000000   ................
ffffa77fc1f0bed8:  0000000000000000 ffff9a2b00000000   ............+...
ffffa77fc1f0bee8:  ffff9a2b5a9997c0 ffffffff8fc8ab70   ...Z+...p.......
ffffa77fc1f0bef8:  ffff9a2c418a5ea8 ffff9a2c418a5ea8   .^.A,....^.A,...
ffffa77fc1f0bf08:  0000000000000000 695bd83cf55d9800   .........].<.[i
ffffa77fc1f0bf18:  0000000000000000 ffffa77fc1f0bf58   ........X.......
ffffa77fc1f0bf28:  0000000000000000 0000000000000000   ................
ffffa77fc1f0bf38:  0000000000000000 ffffffff904b3883   .........8K.....
ffffa77fc1f0bf48:  0000000000000000 ffffffff9060008c   .........`.....

crash> rd -SS -64 ffffa77fc1f0bdc8 50
ffffa77fc1f0bdc8:  ffff9a2c5bcafcc0 [ffff9a2b5a9997c0:task_struct]
ffffa77fc1f0bdd8:  ffff9a2c5bcafcc0 [ffff9a2c4024df00:task_struct]
ffffa77fc1f0bde8:  fair_sched_class ffffa77fc1f0be50
ffffa77fc1f0bdf8:  __schedule+642   [ffff9a2b5a99a190:task_struct]
ffffa77fc1f0be08:  00000000000003e8 wait_consider_task+2503
ffffa77fc1f0be18:  ffff9a2c00000004 695bd83cf55d9800
ffffa77fc1f0be28:  [ffff9a2b5a9997c0:task_struct] [ffff9a2b5a9997c0:task_struct]
ffffa77fc1f0be38:  [ffff9a2b5a9997c0:task_struct] ffffa77fc1f0bee0
ffffa77fc1f0be48:  [ffff9a2b5a9997b0:task_struct] [ffff9a2b5a9997c0:task_struct]
ffffa77fc1f0be58:  schedule+70      ffffa77fc1f0beb8
ffffa77fc1f0be68:  [ffff9a2b5a99a0c0:task_struct] do_wait+431
ffffa77fc1f0be78:  0000000000000000 000000000000000e
ffffa77fc1f0be88:  00007fff03321620 0000000000000000
ffffa77fc1f0be98:  0000000000000000 0000000000000004
```

```
ffffa77fc1f0bea8:  0000000000000000 kernel_wait4+166
ffffa77fc1f0beb8:  0000000e00000004 0000000000000000
ffffa77fc1f0bec8:  0000000000000000 0000000000000000
ffffa77fc1f0bed8:  0000000000000000 ffff9a2b00000000
ffffa77fc1f0bee8:  [ffff9a2b5a9997c0:task_struct] child_wait_callback
ffffa77fc1f0bef8:  [ffff9a2c418a5ea8:signal_cache] [ffff9a2c418a5ea8:signal_cache]
ffffa77fc1f0bf08:  0000000000000000 695bd83cf55d9800
ffffa77fc1f0bf18:  0000000000000000 ffffa77fc1f0bf58
ffffa77fc1f0bf28:  0000000000000000 0000000000000000
ffffa77fc1f0bf38:  0000000000000000 do_syscall_64+51
ffffa77fc1f0bf48:  0000000000000000 entry_SYSCALL_64_after_hwframe+68
```

14. Verify the return address by disassembly:

```
crash> bt
PID: 2105   TASK: ffff9a2b5a9997c0  CPU: 1   COMMAND: "bash"
 #0 [ffffa77fc1f0bdc8] __schedule at ffffffff904c0112
 #1 [ffffa77fc1f0be58] schedule at ffffffff904c0746
 #2 [ffffa77fc1f0be70] do_wait at ffffffff8fc8bd7f
 #3 [ffffa77fc1f0beb0] kernel_wait4 at ffffffff8fc8d1d6
 #4 [ffffa77fc1f0bf40] do_syscall_64 at ffffffff904b3883
 #5 [ffffa77fc1f0bf50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007fb46aa3c1c6  RSP: 00007fff03321608  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 000000000000000a  RCX: 00007fb46aa3c1c6
    RDX: 000000000000000a  RSI: 00007fff03321620  RDI: 00000000ffffffff
    RBP: 0000000000000000   R8: 0000000000000000   R9: 0000000000000000
    R10: 0000000000000000  R11: 0000000000000246  R12: 0000000000000000
    R13: 0000000000000000  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: 000000000000003d  CS: 0033  SS: 002b

crash> dis schedule
0xffffffff904c0700 <schedule>:  nopl   0x0(%rax,%rax,1) [FTRACE NOP]
0xffffffff904c0705 <schedule+5>:        push   %rbp
0xffffffff904c0706 <schedule+6>:        mov    %gs:0x1bbc0,%rbp
0xffffffff904c070f <schedule+15>:       push   %rbx
0xffffffff904c0710 <schedule+16>:       mov    0x10(%rbp),%rax
0xffffffff904c0714 <schedule+20>:       test   %rax,%rax
0xffffffff904c0717 <schedule+23>:       je     0xffffffff904c0736 <schedule+54>
0xffffffff904c0719 <schedule+25>:       mov    0x24(%rbp),%eax
0xffffffff904c071c <schedule+28>:       test   $0x30,%al
0xffffffff904c071e <schedule+30>:       je     0xffffffff904c072c <schedule+44>
0xffffffff904c0720 <schedule+32>:       mov    %rbp,%rdi
0xffffffff904c0723 <schedule+35>:       test   $0x20,%al
0xffffffff904c0725 <schedule+37>:       je     0xffffffff904c078c <schedule+140>
0xffffffff904c0727 <schedule+39>:       call   0xffffffff8fca7ae0 <wq_worker_sleeping>
0xffffffff904c072c <schedule+44>:       cmpq   $0x0,0xba0(%rbp)
0xffffffff904c0734 <schedule+52>:       je     0xffffffff904c0765 <schedule+101>
0xffffffff904c0736 <schedule+54>:       mov    %gs:0x1bbc0,%rbx
0xffffffff904c073f <schedule+63>:       xor    %edi,%edi
0xffffffff904c0741 <schedule+65>:       call   0xffffffff904bfe90 <__schedule>
0xffffffff904c0746 <schedule+70>:       mov    (%rbx),%rax
0xffffffff904c0749 <schedule+73>:       test   $0x8,%al
0xffffffff904c074b <schedule+75>:       jne    0xffffffff904c073f <schedule+63>
0xffffffff904c074d <schedule+77>:       mov    0x24(%rbp),%eax
0xffffffff904c0750 <schedule+80>:       test   $0x30,%al
0xffffffff904c0752 <schedule+82>:       je     0xffffffff904c0762 <schedule+98>
0xffffffff904c0754 <schedule+84>:       mov    %rbp,%rdi
0xffffffff904c0757 <schedule+87>:       test   $0x20,%al
0xffffffff904c0759 <schedule+89>:       je     0xffffffff904c0785 <schedule+133>
0xffffffff904c075b <schedule+91>:       pop    %rbx
```

```
0xffffffff904c075c <schedule+92>:       pop    %rbp
0xffffffff904c075d <schedule+93>:       jmp    0xffffffff8fca7ab0 <wq_worker_running>
0xffffffff904c0762 <schedule+98>:       pop    %rbx
0xffffffff904c0763 <schedule+99>:       pop    %rbp
0xffffffff904c0764 <schedule+100>:      ret
0xffffffff904c0765 <schedule+101>:      mov    0xbb8(%rbp),%rdi
0xffffffff904c076c <schedule+108>:      test   %rdi,%rdi
0xffffffff904c076f <schedule+111>:      je     0xffffffff904c0736 <schedule+54>
0xffffffff904c0771 <schedule+113>:      mov    (%rdi),%rax
0xffffffff904c0774 <schedule+116>:      cmp    %rax,%rdi
0xffffffff904c0777 <schedule+119>:      je     0xffffffff904c0793 <schedule+147>
0xffffffff904c0779 <schedule+121>:      mov    $0x1,%esi
0xffffffff904c077e <schedule+126>:      call   0xffffffff9001a390 <blk_flush_plug_list>
0xffffffff904c0783 <schedule+131>:      jmp    0xffffffff904c0736 <schedule+54>
0xffffffff904c0785 <schedule+133>:      pop    %rbx
0xffffffff904c0786 <schedule+134>:      pop    %rbp
0xffffffff904c0787 <schedule+135>:      jmp    0xffffffff8ff385d0 <io_wq_worker_running>
0xffffffff904c078c <schedule+140>:      call   0xffffffff8ff38610 <io_wq_worker_sleeping>
0xffffffff904c0791 <schedule+145>:      jmp    0xffffffff904c072c <schedule+44>
0xffffffff904c0793 <schedule+147>:      mov    0x10(%rdi),%rdx
0xffffffff904c0797 <schedule+151>:      lea    0x10(%rdi),%rax
0xffffffff904c079b <schedule+155>:      cmp    %rax,%rdx
0xffffffff904c079e <schedule+158>:      jne    0xffffffff904c0779 <schedule+121>
0xffffffff904c07a0 <schedule+160>:      jmp    0xffffffff904c0736 <schedule+54>
```

Note: To emulate backward disassembly similar to the **ub** WinDbg command, use the **-r** flag:

```
crash> dis -r ffffffff904c0746
0xffffffff904c0700 <schedule>:  nopl   0x0(%rax,%rax,1) [FTRACE NOP]
0xffffffff904c0705 <schedule+5>:        push   %rbp
0xffffffff904c0706 <schedule+6>:        mov    %gs:0x1bbc0,%rbp
0xffffffff904c070f <schedule+15>:       push   %rbx
0xffffffff904c0710 <schedule+16>:       mov    0x10(%rbp),%rax
0xffffffff904c0714 <schedule+20>:       test   %rax,%rax
0xffffffff904c0717 <schedule+23>:       je     0xffffffff904c0736 <schedule+54>
0xffffffff904c0719 <schedule+25>:       mov    0x24(%rbp),%eax
0xffffffff904c071c <schedule+28>:       test   $0x30,%al
0xffffffff904c071e <schedule+30>:       je     0xffffffff904c072c <schedule+44>
0xffffffff904c0720 <schedule+32>:       mov    %rbp,%rdi
0xffffffff904c0723 <schedule+35>:       test   $0x20,%al
0xffffffff904c0725 <schedule+37>:       je     0xffffffff904c078c <schedule+140>
0xffffffff904c0727 <schedule+39>:       call   0xffffffff8fca7ae0 <wq_worker_sleeping>
0xffffffff904c072c <schedule+44>:       cmpq   $0x0,0xba0(%rbp)
0xffffffff904c0734 <schedule+52>:       je     0xffffffff904c0765 <schedule+101>
0xffffffff904c0736 <schedule+54>:       mov    %gs:0x1bbc0,%rbx
0xffffffff904c073f <schedule+63>:       xor    %edi,%edi
0xffffffff904c0741 <schedule+65>:       call   0xffffffff904bfe90 <__schedule>
0xffffffff904c0746 <schedule+70>:       mov    (%rbx),%rax
```

15. Finally, we can see the backtrace of every PID/TID (task) in the system:

```
crash> foreach bt
PID: 0      TASK: ffffffff91213940  CPU: 0   COMMAND: "swapper/0"
 #0 [fffffe000000de50] crash_nmi_callback at ffffffff8fc58e43
 #1 [fffffe000000de58] nmi_handle at ffffffff8fc2e168
 #2 [fffffe000000dea0] default_do_nmi at ffffffff904b4fe2
 #3 [fffffe000000dec8] exc_nmi at ffffffff904b51ff
 #4 [fffffe000000def0] end_repeat_nmi at ffffffff906014db
    [exception RIP: native_safe_halt+14]
    RIP: ffffffff904c3eee  RSP: ffffffff91203eb8  RFLAGS: 00000206
    RAX: ffffffff904c3d90  RBX: 0000000000000000  RCX: ffff9a2c5bc309c0
    RDX: 000000000002e20a  RSI: ffffffff91203e50  RDI: 000000404ebfff26
    RBP: ffffffff91213940   R8: 0000000000000001   R9: 0000000000015400
    R10: 0000000000015400  R11: 0000000000000000  R12: 0000000000000000
    R13: 0000000000000000  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
--- <NMI exception stack> ---
 #5 [ffffffff91203eb8] native_safe_halt at ffffffff904c3eee
 #6 [ffffffff91203eb8] default_idle at ffffffff904c3d9a
 #7 [ffffffff91203ec0] default_idle_call at ffffffff904c4008
 #8 [ffffffff91203ec8] do_idle at ffffffff8fcc17a8
 #9 [ffffffff91203f08] cpu_startup_entry at ffffffff8fcc19c9
#10 [ffffffff91203f18] start_kernel at ffffffff9183609c
#11 [ffffffff91203f50] secondary_startup_64_no_verify at ffffffff8fc000f5

PID: 0      TASK: ffff9a2c4024df00  CPU: 1   COMMAND: "swapper/1"
 #0 [fffffe0000048e50] crash_nmi_callback at ffffffff8fc58e43
 #1 [fffffe0000048e58] nmi_handle at ffffffff8fc2e168
 #2 [fffffe0000048ea0] default_do_nmi at ffffffff904b4fe2
 #3 [fffffe0000048ec8] exc_nmi at ffffffff904b51ff
 #4 [fffffe0000048ef0] end_repeat_nmi at ffffffff906014db
    [exception RIP: native_safe_halt+14]
    RIP: ffffffff904c3eee  RSP: ffffa77fc0083ef0  RFLAGS: 00000216
    RAX: ffffffff904c3d90  RBX: 0000000000000001  RCX: ffff9a2c5bcb09c0
    RDX: 000000000003289e  RSI: ffffa77fc0083e88  RDI: 000000404ebfff26
    RBP: ffff9a2c4024df00   R8: 0000000000000001   R9: 0000000000005400
    R10: 0000000000005400  R11: 0000000000000000  R12: 0000000000000000
    R13: 0000000000000000  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
--- <NMI exception stack> ---
 #5 [ffffa77fc0083ef0] native_safe_halt at ffffffff904c3eee
 #6 [ffffa77fc0083ef0] default_idle at ffffffff904c3d9a
 #7 [ffffa77fc0083ef8] default_idle_call at ffffffff904c4008
 #8 [ffffa77fc0083f00] do_idle at ffffffff8fcc17a8
 #9 [ffffa77fc0083f40] cpu_startup_entry at ffffffff8fcc19c9
#10 [ffffa77fc0083f50] secondary_startup_64_no_verify at ffffffff8fc000f5

PID: 0      TASK: ffff9a2c402697c0  CPU: 2   COMMAND: "swapper/2"
 #0 [fffffe0000083e50] crash_nmi_callback at ffffffff8fc58e43
 #1 [fffffe0000083e58] nmi_handle at ffffffff8fc2e168
 #2 [fffffe0000083ea0] default_do_nmi at ffffffff904b4fe2
 #3 [fffffe0000083ec8] exc_nmi at ffffffff904b51ff
 #4 [fffffe0000083ef0] end_repeat_nmi at ffffffff906014db
    [exception RIP: need_update+33]
    RIP: ffffffff8fe2ec31  RSP: ffffa77fc008be98  RFLAGS: 00000006
    RAX: ffff9a2c5ffd2b80  RBX: ffff9a2c5bd34280  RCX: 0000000000000000
    RDX: ffff9a2c5ffd3700  RSI: 0000000000000000  RDI: ffff9a2c5ffd25c0
    RBP: ffff9a2c5ffd2b80   R8: 0000000000000000   R9: 0000004047fb3f97
    R10: 0000000000000000  R11: 0000000000000000  R12: 0000000000000002
```

```
    R13: 0000000000000002  R14: 0000000000000000  R15: 000000404ebfff26
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
--- <NMI exception stack> ---
 #5 [ffffa77fc008be98] need_update at ffffffff8fe2ec31
 #6 [ffffa77fc008beb0] quiet_vmstat at ffffffff8fe30da1
 #7 [ffffa77fc008beb8] tick_nohz_idle_stop_tick at ffffffff8fd2a6ae
 #8 [ffffa77fc008bf00] do_idle at ffffffff8fcc17a3
 #9 [ffffa77fc008bf40] cpu_startup_entry at ffffffff8fcc19c9
#10 [ffffa77fc008bf50] secondary_startup_64_no_verify at ffffffff8fc000f5

PID: 0      TASK: ffff9a2c4026df00  CPU: 3   COMMAND: "swapper/3"
 #0 [ffffa77fc0093e60] __schedule at ffffffff904c0112
 #1 [ffffa77fc0093ef0] schedule_idle at ffffffff904c0a48
 #2 [ffffa77fc0093f00] do_idle at ffffffff8fcc1707
 #3 [ffffa77fc0093f40] cpu_startup_entry at ffffffff8fcc19c9
 #4 [ffffa77fc0093f50] secondary_startup_64_no_verify at ffffffff8fc000f5

PID: 1      TASK: ffff9a2c401f4740  CPU: 3   COMMAND: "systemd"
 #0 [ffffa77fc0013d60] __schedule at ffffffff904c0112
 #1 [ffffa77fc0013df0] schedule at ffffffff904c0746
 #2 [ffffa77fc0013e08] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc0013e78] do_epoll_wait at ffffffff8ff1a28a
 #4 [ffffa77fc0013f38] __x64_sys_epoll_wait at ffffffff8ff1a39a
 #5 [ffffa77fc0013f40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc0013f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007fb9ab61e116  RSP: 00007ffee2cb9c50  RFLAGS: 00000293
    RAX: ffffffffffffffda  RBX: 000056311694d0e0  RCX: 00007fb9ab61e116
    RDX: 000000000000009b  RSI: 0000563116ab9ff0  RDI: 0000000000000004
    RBP: ffffffffffffffff   R8: 0000000000000000   R9: ed497f7459b37c70
    R10: 00000000ffffffff  R11: 0000000000000293  R12: 0000000000000001
    R13: 000000000000009b  R14: 0000000000000000  R15: 00005631159e2b4e
    ORIG_RAX: 00000000000000e8  CS: 0033  SS: 002b

PID: 2      TASK: ffff9a2c401f2f80  CPU: 1   COMMAND: "kthreadd"
 #0 [ffffa77fc001be08] __schedule at ffffffff904c0112
 #1 [ffffa77fc001be98] schedule at ffffffff904c0746
 #2 [ffffa77fc001beb0] kthreadd at ffffffff8fcae036
 #3 [ffffa77fc001bf50] ret_from_fork at ffffffff8fc04442

PID: 3      TASK: ffff9a2c401f0000  CPU: 0   COMMAND: "rcu_gp"
 #0 [ffffa77fc0023df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0023e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0023e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0023f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0023f50] ret_from_fork at ffffffff8fc04442

PID: 4      TASK: ffff9a2c401f17c0  CPU: 0   COMMAND: "rcu_par_gp"
 #0 [ffffa77fc002bdf0] __schedule at ffffffff904c0112
 #1 [ffffa77fc002be80] schedule at ffffffff904c0746
 #2 [ffffa77fc002be98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc002bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc002bf50] ret_from_fork at ffffffff8fc04442

PID: 5      TASK: ffff9a2c401f5f00  CPU: 0   COMMAND: "kworker/0:0"
 #0 [ffffa77fc0033e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc0033eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc0033ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc0033f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0033f50] ret_from_fork at ffffffff8fc04442
```

```
PID: 6      TASK: ffff9a2c402297c0  CPU: 0   COMMAND: "kworker/0:0H"
 #0 [ffffa77fc003be28] __schedule at ffffffff904c0112
 #1 [ffffa77fc003beb8] schedule at ffffffff904c0746
 #2 [ffffa77fc003bed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc003bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc003bf50] ret_from_fork at ffffffff8fc04442

PID: 7      TASK: ffff9a2c4022df00  CPU: 0   COMMAND: "kworker/0:1"
 #0 [ffffa77fc0043e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc0043eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc0043ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc0043f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0043f50] ret_from_fork at ffffffff8fc04442

PID: 8      TASK: ffff9a2c4022c740  CPU: 3   COMMAND: "kworker/u8:0"
 #0 [ffffa77fc004be28] __schedule at ffffffff904c0112
 #1 [ffffa77fc004beb8] schedule at ffffffff904c0746
 #2 [ffffa77fc004bed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc004bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc004bf50] ret_from_fork at ffffffff8fc04442

PID: 9      TASK: ffff9a2c4022af80  CPU: 0   COMMAND: "mm_percpu_wq"
 #0 [ffffa77fc0053df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0053e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0053e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0053f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0053f50] ret_from_fork at ffffffff8fc04442

PID: 10     TASK: ffff9a2c40228000  CPU: 0   COMMAND: "rcu_tasks_rude_"
 #0 [ffffa77fc005be08] __schedule at ffffffff904c0112
 #1 [ffffa77fc005be98] schedule at ffffffff904c0746
 #2 [ffffa77fc005beb0] rcu_tasks_kthread at ffffffff8fd0413f
 #3 [ffffa77fc005bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc005bf50] ret_from_fork at ffffffff8fc04442

PID: 11     TASK: ffff9a2c4024c740  CPU: 0   COMMAND: "rcu_tasks_trace"
 #0 [ffffa77fc0063e08] __schedule at ffffffff904c0112
 #1 [ffffa77fc0063e98] schedule at ffffffff904c0746
 #2 [ffffa77fc0063eb0] rcu_tasks_kthread at ffffffff8fd0413f
 #3 [ffffa77fc0063f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0063f50] ret_from_fork at ffffffff8fc04442

PID: 12     TASK: ffff9a2c4024af80  CPU: 0   COMMAND: "ksoftirqd/0"
 #0 [ffffa77fc006be40] __schedule at ffffffff904c0112
 #1 [ffffa77fc006bed0] schedule at ffffffff904c0746
 #2 [ffffa77fc006bee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc006bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc006bf50] ret_from_fork at ffffffff8fc04442

PID: 13     TASK: ffff9a2c40248000  CPU: 1   COMMAND: "rcu_sched"
 #0 [ffffa77fc0073d98] __schedule at ffffffff904c0112
 #1 [ffffa77fc0073e28] schedule at ffffffff904c0746
 #2 [ffffa77fc0073e40] schedule_timeout at ffffffff904c333b
 #3 [ffffa77fc0073e98] rcu_gp_kthread at ffffffff8fd098bb
 #4 [ffffa77fc0073f10] kthread at ffffffff8fcac91b
 #5 [ffffa77fc0073f50] ret_from_fork at ffffffff8fc04442

PID: 14     TASK: ffff9a2c402497c0  CPU: 0   COMMAND: "migration/0"
 #0 [ffffa77fc007be40] __schedule at ffffffff904c0112
 #1 [ffffa77fc007bed0] schedule at ffffffff904c0746
```

```
 #2 [ffffa77fc007bee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc007bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc007bf50] ret_from_fork at ffffffff8fc04442

PID: 15     TASK: ffff9a2c4026c740  CPU: 0   COMMAND: "cpuhp/0"
 #0 [ffffa77fc009be40] __schedule at ffffffff904c0112
 #1 [ffffa77fc009bed0] schedule at ffffffff904c0746
 #2 [ffffa77fc009bee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc009bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc009bf50] ret_from_fork at ffffffff8fc04442

PID: 16     TASK: ffff9a2c4026af80  CPU: 1   COMMAND: "cpuhp/1"
 #0 [ffffa77fc00a3e40] __schedule at ffffffff904c0112
 #1 [ffffa77fc00a3ed0] schedule at ffffffff904c0746
 #2 [ffffa77fc00a3ee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc00a3f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00a3f50] ret_from_fork at ffffffff8fc04442

PID: 17     TASK: ffff9a2c40268000  CPU: 1   COMMAND: "migration/1"
 #0 [ffffa77fc00abe40] __schedule at ffffffff904c0112
 #1 [ffffa77fc00abed0] schedule at ffffffff904c0746
 #2 [ffffa77fc00abee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc00abf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00abf50] ret_from_fork at ffffffff8fc04442

PID: 18     TASK: ffff9a2c4028af80  CPU: 1   COMMAND: "ksoftirqd/1"
 #0 [ffffa77fc00b3e40] __schedule at ffffffff904c0112
 #1 [ffffa77fc00b3ed0] schedule at ffffffff904c0746
 #2 [ffffa77fc00b3ee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc00b3f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00b3f50] ret_from_fork at ffffffff8fc04442

PID: 19     TASK: ffff9a2c40288000  CPU: 1   COMMAND: "kworker/1:0"
 #0 [ffffa77fc00bbe28] __schedule at ffffffff904c0112
 #1 [ffffa77fc00bbeb8] schedule at ffffffff904c0746
 #2 [ffffa77fc00bbed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc00bbf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00bbf50] ret_from_fork at ffffffff8fc04442

PID: 20     TASK: ffff9a2c402897c0  CPU: 1   COMMAND: "kworker/1:0H"
 #0 [ffffa77fc00c3e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc00c3eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc00c3ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc00c3f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00c3f50] ret_from_fork at ffffffff8fc04442

PID: 21     TASK: ffff9a2c4028df00  CPU: 2   COMMAND: "cpuhp/2"
 #0 [ffffa77fc00cfe40] __schedule at ffffffff904c0112
 #1 [ffffa77fc00cfed0] schedule at ffffffff904c0746
 #2 [ffffa77fc00cfee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc00cff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00cff50] ret_from_fork at ffffffff8fc04442

PID: 22     TASK: ffff9a2c4028c740  CPU: 2   COMMAND: "migration/2"
 #0 [ffffa77fc00d7e40] __schedule at ffffffff904c0112
 #1 [ffffa77fc00d7ed0] schedule at ffffffff904c0746
 #2 [ffffa77fc00d7ee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc00d7f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00d7f50] ret_from_fork at ffffffff8fc04442
```

```
PID: 23     TASK: ffff9a2c402b4740  CPU: 2   COMMAND: "ksoftirqd/2"
 #0 [ffffa77fc00dfe40] __schedule at ffffffff904c0112
 #1 [ffffa77fc00dfed0] schedule at ffffffff904c0746
 #2 [ffffa77fc00dfee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc00dff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00dff50] ret_from_fork at ffffffff8fc04442

PID: 24     TASK: ffff9a2c402b2f80  CPU: 2   COMMAND: "kworker/2:0"
 #0 [ffffa77fc00e7e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc00e7eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc00e7ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc00e7f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00e7f50] ret_from_fork at ffffffff8fc04442

PID: 25     TASK: ffff9a2c402b0000  CPU: 2   COMMAND: "kworker/2:0H"
 #0 [ffffa77fc00efe28] __schedule at ffffffff904c0112
 #1 [ffffa77fc00efeb8] schedule at ffffffff904c0746
 #2 [ffffa77fc00efed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc00eff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00eff50] ret_from_fork at ffffffff8fc04442

PID: 26     TASK: ffff9a2c402b17c0  CPU: 3   COMMAND: "cpuhp/3"
 #0 [ffffa77fc00fbe40] __schedule at ffffffff904c0112
 #1 [ffffa77fc00fbed0] schedule at ffffffff904c0746
 #2 [ffffa77fc00fbee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc00fbf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc00fbf50] ret_from_fork at ffffffff8fc04442

PID: 27     TASK: ffff9a2c402b5f00  CPU: 3   COMMAND: "migration/3"
 #0 [ffffa77fc0103e40] __schedule at ffffffff904c0112
 #1 [ffffa77fc0103ed0] schedule at ffffffff904c0746
 #2 [ffffa77fc0103ee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc0103f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0103f50] ret_from_fork at ffffffff8fc04442

PID: 28     TASK: ffff9a2c402e17c0  CPU: 3   COMMAND: "ksoftirqd/3"
 #0 [ffffa77fc010be40] __schedule at ffffffff904c0112
 #1 [ffffa77fc010bed0] schedule at ffffffff904c0746
 #2 [ffffa77fc010bee8] smpboot_thread_fn at ffffffff8fcb38db
 #3 [ffffa77fc010bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc010bf50] ret_from_fork at ffffffff8fc04442

PID: 29     TASK: ffff9a2c402e5f00  CPU: 3   COMMAND: "kworker/3:0"
 #0 [ffffa77fc0113e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc0113eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc0113ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc0113f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0113f50] ret_from_fork at ffffffff8fc04442

PID: 30     TASK: ffff9a2c402e4740  CPU: 3   COMMAND: "kworker/3:0H"
 #0 [ffffa77fc011be28] __schedule at ffffffff904c0112
 #1 [ffffa77fc011beb8] schedule at ffffffff904c0746
 #2 [ffffa77fc011bed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc011bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc011bf50] ret_from_fork at ffffffff8fc04442

PID: 32     TASK: ffff9a2c4031af80  CPU: 1   COMMAND: "kworker/u8:1"
 #0 [ffffa77fc012fe28] __schedule at ffffffff904c0112
 #1 [ffffa77fc012feb8] schedule at ffffffff904c0746
 #2 [ffffa77fc012fed0] worker_thread at ffffffff8fca6ba1
```

```
 #3 [ffffa77fc012ff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc012ff50] ret_from_fork at ffffffff8fc04442

PID: 33     TASK: ffff9a2c40342f80  CPU: 2   COMMAND: "kworker/u8:2"
 #0 [ffffa77fc0137e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc0137eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc0137ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc0137f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0137f50] ret_from_fork at ffffffff8fc04442

PID: 34     TASK: ffff9a2c40340000  CPU: 0   COMMAND: "kdevtmpfs"
 #0 [ffffa77fc013fe48] __schedule at ffffffff904c0112
 #1 [ffffa77fc013fed8] schedule at ffffffff904c0746
 #2 [ffffa77fc013fef0] devtmpfsd at ffffffff904bb64b
 #3 [ffffa77fc013ff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc013ff50] ret_from_fork at ffffffff8fc04442

PID: 35     TASK: ffff9a2c403417c0  CPU: 1   COMMAND: "netns"
 #0 [ffffa77fc0147df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0147e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0147e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0147f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0147f50] ret_from_fork at ffffffff8fc04442

PID: 36     TASK: ffff9a2c40345f00  CPU: 2   COMMAND: "kauditd"
 #0 [ffffa77fc014fe00] __schedule at ffffffff904c0112
 #1 [ffffa77fc014fe90] schedule at ffffffff904c0746
 #2 [ffffa77fc014fea8] kauditd_thread at ffffffff8fd56d25
 #3 [ffffa77fc014ff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc014ff50] ret_from_fork at ffffffff8fc04442

PID: 37     TASK: ffff9a2c40344740  CPU: 1   COMMAND: "khungtaskd"
 #0 [ffffa77fc0157dc0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0157e50] schedule at ffffffff904c0746
 #2 [ffffa77fc0157e68] schedule_timeout at ffffffff904c333b
 #3 [ffffa77fc0157ec0] watchdog at ffffffff8fd67ca0
 #4 [ffffa77fc0157f10] kthread at ffffffff8fcac91b
 #5 [ffffa77fc0157f50] ret_from_fork at ffffffff8fc04442

PID: 38     TASK: ffff9a2c5bd717c0  CPU: 3   COMMAND: "oom_reaper"
 #0 [ffffa77fc015fe00] __schedule at ffffffff904c0112
 #1 [ffffa77fc015fe90] schedule at ffffffff904c0746
 #2 [ffffa77fc015fea8] oom_reaper at ffffffff8fe124a6
 #3 [ffffa77fc015ff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc015ff50] ret_from_fork at ffffffff8fc04442

PID: 39     TASK: ffff9a2c5bd75f00  CPU: 1   COMMAND: "writeback"
 #0 [ffffa77fc0167df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0167e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0167e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0167f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0167f50] ret_from_fork at ffffffff8fc04442

PID: 40     TASK: ffff9a2c5bd74740  CPU: 1   COMMAND: "kcompactd0"
 #0 [ffffa77fc016fd98] __schedule at ffffffff904c0112
 #1 [ffffa77fc016fe28] schedule at ffffffff904c0746
 #2 [ffffa77fc016fe40] schedule_timeout at ffffffff904c333b
 #3 [ffffa77fc016fe98] kcompactd at ffffffff8fe3e08a
 #4 [ffffa77fc016ff10] kthread at ffffffff8fcac91b
 #5 [ffffa77fc016ff50] ret_from_fork at ffffffff8fc04442
```

```
PID: 41     TASK: ffff9a2c5bd72f80  CPU: 1   COMMAND: "ksmd"
 #0 [ffffa77fc0177da0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0177e30] schedule at ffffffff904c0746
 #2 [ffffa77fc0177e48] ksm_scan_thread at ffffffff8fe8c092
 #3 [ffffa77fc0177f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0177f50] ret_from_fork at ffffffff8fc04442

PID: 42     TASK: ffff9a2c5bd70000  CPU: 1   COMMAND: "khugepaged"
 #0 [ffffa77fc017fc88] __schedule at ffffffff904c0112
 #1 [ffffa77fc017fd18] schedule at ffffffff904c0746
 #2 [ffffa77fc017fd30] schedule_timeout at ffffffff904c333b
 #3 [ffffa77fc017fd88] khugepaged at ffffffff8fea4b5d
 #4 [ffffa77fc017ff10] kthread at ffffffff8fcac91b
 #5 [ffffa77fc017ff50] ret_from_fork at ffffffff8fc04442

PID: 44     TASK: ffff9a2c5bdddf00  CPU: 3   COMMAND: "kworker/3:1"
 #0 [ffffa77fc018fe28] __schedule at ffffffff904c0112
 #1 [ffffa77fc018feb8] schedule at ffffffff904c0746
 #2 [ffffa77fc018fed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc018ff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc018ff50] ret_from_fork at ffffffff8fc04442

PID: 52     TASK: ffff9a2c5bdf0000  CPU: 1   COMMAND: "kworker/1:1"
 #0 [ffffa77fc01cfe28] __schedule at ffffffff904c0112
 #1 [ffffa77fc01cfeb8] schedule at ffffffff904c0746
 #2 [ffffa77fc01cfed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc01cff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc01cff50] ret_from_fork at ffffffff8fc04442

PID: 62     TASK: ffff9a2c403a2f80  CPU: 0   COMMAND: "kintegrityd"
 #0 [ffffa77fc0903df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0903e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0903e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0903f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0903f50] ret_from_fork at ffffffff8fc04442

PID: 63     TASK: ffff9a2c4082c740  CPU: 0   COMMAND: "kblockd"
 #0 [ffffa77fc090bdf0] __schedule at ffffffff904c0112
 #1 [ffffa77fc090be80] schedule at ffffffff904c0746
 #2 [ffffa77fc090be98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc090bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc090bf50] ret_from_fork at ffffffff8fc04442

PID: 64     TASK: ffff9a2c4082af80  CPU: 1   COMMAND: "blkcg_punt_bio"
 #0 [ffffa77fc0913df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0913e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0913e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0913f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0913f50] ret_from_fork at ffffffff8fc04442

PID: 65     TASK: ffff9a2c40828000  CPU: 3   COMMAND: "edac-poller"
 #0 [ffffa77fc091bdf0] __schedule at ffffffff904c0112
 #1 [ffffa77fc091be80] schedule at ffffffff904c0746
 #2 [ffffa77fc091be98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc091bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc091bf50] ret_from_fork at ffffffff8fc04442

PID: 66     TASK: ffff9a2c408297c0  CPU: 1   COMMAND: "devfreq_wq"
 #0 [ffffa77fc0923df0] __schedule at ffffffff904c0112
```

```
 #1 [ffffa77fc0923e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0923e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0923f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0923f50] ret_from_fork at ffffffff8fc04442

PID: 67     TASK: ffff9a2c4082df00  CPU: 0   COMMAND: "kworker/0:1H"
 #0 [ffffa77fc092be28] __schedule at ffffffff904c0112
 #1 [ffffa77fc092beb8] schedule at ffffffff904c0746
 #2 [ffffa77fc092bed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc092bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc092bf50] ret_from_fork at ffffffff8fc04442

PID: 69     TASK: ffff9a2c47ef0000  CPU: 3   COMMAND: "kswapd0"
 #0 [ffffa77fc01b7df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc01b7e80] schedule at ffffffff904c0746
 #2 [ffffa77fc01b7e98] kswapd at ffffffff8fe24bf5
 #3 [ffffa77fc01b7f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc01b7f50] ret_from_fork at ffffffff8fc04442

PID: 70     TASK: ffff9a2c47ef17c0  CPU: 1   COMMAND: "kthrotld"
 #0 [ffffa77fc01bfdf0] __schedule at ffffffff904c0112
 #1 [ffffa77fc01bfe80] schedule at ffffffff904c0746
 #2 [ffffa77fc01bfe98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc01bff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc01bff50] ret_from_fork at ffffffff8fc04442

PID: 71     TASK: ffff9a2c47ef5f00  CPU: 3   COMMAND: "acpi_thermal_pm"
 #0 [ffffa77fc01c7df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc01c7e80] schedule at ffffffff904c0746
 #2 [ffffa77fc01c7e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc01c7f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc01c7f50] ret_from_fork at ffffffff8fc04442

PID: 72     TASK: ffff9a2c47ef4740  CPU: 1   COMMAND: "ipv6_addrconf"
 #0 [ffffa77fc01d7df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc01d7e80] schedule at ffffffff904c0746
 #2 [ffffa77fc01d7e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc01d7f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc01d7f50] ret_from_fork at ffffffff8fc04442

PID: 77     TASK: ffff9a2c403a4740  CPU: 2   COMMAND: "kworker/2:1"
 #0 [ffffa77fc01dfe28] __schedule at ffffffff904c0112
 #1 [ffffa77fc01dfeb8] schedule at ffffffff904c0746
 #2 [ffffa77fc01dfed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc01dff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc01dff50] ret_from_fork at ffffffff8fc04442

PID: 82     TASK: ffff9a2c403a0000  CPU: 2   COMMAND: "kstrp"
 #0 [ffffa77fc0207df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0207e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0207e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0207f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0207f50] ret_from_fork at ffffffff8fc04442

PID: 85     TASK: ffff9a2c4039df00  CPU: 3   COMMAND: "zswap-shrink"
 #0 [ffffa77fc021fdf0] __schedule at ffffffff904c0112
 #1 [ffffa77fc021fe80] schedule at ffffffff904c0746
 #2 [ffffa77fc021fe98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc021ff10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc021ff50] ret_from_fork at ffffffff8fc04442
```

```
PID: 86     TASK: ffff9a2c403997c0  CPU: 2   COMMAND: "kworker/u9:0"
 #0 [ffffa77fc0227e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc0227eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc0227ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc0227f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0227f50] ret_from_fork at ffffffff8fc04442

PID: 108    TASK: ffff9a2c58e65f00  CPU: 1   COMMAND: "kworker/1:1H"
 #0 [ffffa77fc0283e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc0283eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc0283ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc0283f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0283f50] ret_from_fork at ffffffff8fc04442

PID: 122    TASK: ffff9a2c58e14740  CPU: 3   COMMAND: "kworker/3:2"
 #0 [ffffa77fc01e7e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc01e7eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc01e7ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc01e7f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc01e7f50] ret_from_fork at ffffffff8fc04442

PID: 130    TASK: ffff9a2c58dc17c0  CPU: 3   COMMAND: "kworker/3:1H"
 #0 [ffffa77fc026be28] __schedule at ffffffff904c0112
 #1 [ffffa77fc026beb8] schedule at ffffffff904c0746
 #2 [ffffa77fc026bed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc026bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc026bf50] ret_from_fork at ffffffff8fc04442

PID: 131    TASK: ffff9a2c4039c740  CPU: 0   COMMAND: "ata_sff"
 #0 [ffffa77fc0273df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0273e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0273e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0273f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0273f50] ret_from_fork at ffffffff8fc04442

PID: 133    TASK: ffff9a2c40398000  CPU: 1   COMMAND: "scsi_eh_0"
 #0 [ffffa77fc031bdf0] __schedule at ffffffff904c0112
 #1 [ffffa77fc031be80] schedule at ffffffff904c0746
 #2 [ffffa77fc031be98] scsi_error_handler at ffffffffc0194463 [scsi_mod]
 #3 [ffffa77fc031bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc031bf50] ret_from_fork at ffffffff8fc04442

PID: 134    TASK: ffff9a2c50d94740  CPU: 1   COMMAND: "scsi_tmf_0"
 #0 [ffffa77fc0323df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0323e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0323e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0323f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0323f50] ret_from_fork at ffffffff8fc04442

PID: 137    TASK: ffff9a2c4031c740  CPU: 3   COMMAND: "kworker/3:3"
 #0 [ffffa77fc02c3e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc02c3eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc02c3ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc02c3f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc02c3f50] ret_from_fork at ffffffff8fc04442

PID: 138    TASK: ffff9a2c591d2f80  CPU: 1   COMMAND: "scsi_eh_1"
 #0 [ffffa77fc02cbdf0] __schedule at ffffffff904c0112
 #1 [ffffa77fc02cbe80] schedule at ffffffff904c0746
```

```
 #2 [ffffa77fc02cbe98] scsi_error_handler at ffffffffc0194463 [scsi_mod]
 #3 [ffffa77fc02cbf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc02cbf50] ret_from_fork at ffffffff8fc04442

PID: 139    TASK: ffff9a2c591d0000  CPU: 3   COMMAND: "scsi_tmf_1"
 #0 [ffffa77fc02d3df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc02d3e80] schedule at ffffffff904c0746
 #2 [ffffa77fc02d3e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc02d3f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc02d3f50] ret_from_fork at ffffffff8fc04442

PID: 140    TASK: ffff9a2c50d92f80  CPU: 2   COMMAND: "scsi_eh_2"
 #0 [ffffa77fc02bbdf0] __schedule at ffffffff904c0112
 #1 [ffffa77fc02bbe80] schedule at ffffffff904c0746
 #2 [ffffa77fc02bbe98] scsi_error_handler at ffffffffc0194463 [scsi_mod]
 #3 [ffffa77fc02bbf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc02bbf50] ret_from_fork at ffffffff8fc04442

PID: 141    TASK: ffff9a2c50d90000  CPU: 0   COMMAND: "scsi_tmf_2"
 #0 [ffffa77fc0333df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0333e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0333e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0333f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0333f50] ret_from_fork at ffffffff8fc04442

PID: 142    TASK: ffff9a2c5bddc740  CPU: 0   COMMAND: "kworker/u8:3"
 #0 [ffffa77fc0127e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc0127eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc0127ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc0127f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0127f50] ret_from_fork at ffffffff8fc04442

PID: 143    TASK: ffff9a2c403a17c0  CPU: 1   COMMAND: "irq/18-vmwgfx"
 #0 [ffffa77fc02dbe08] __schedule at ffffffff904c0112
 #1 [ffffa77fc02dbe98] schedule at ffffffff904c0746
 #2 [ffffa77fc02dbeb0] irq_thread at ffffffff8fcf7931
 #3 [ffffa77fc02dbf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc02dbf50] ret_from_fork at ffffffff8fc04442

PID: 144    TASK: ffff9a2c403a5f00  CPU: 1   COMMAND: "ttm_swap"
 #0 [ffffa77fc02e3df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc02e3e80] schedule at ffffffff904c0746
 #2 [ffffa77fc02e3e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc02e3f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc02e3f50] ret_from_fork at ffffffff8fc04442

PID: 145    TASK: ffff9a2c402e2f80  CPU: 1   COMMAND: "card0-crtc0"
 #0 [ffffa77fc02ebe38] __schedule at ffffffff904c0112
 #1 [ffffa77fc02ebec8] schedule at ffffffff904c0746
 #2 [ffffa77fc02ebee0] kthread_worker_fn at ffffffff8fcadb17
 #3 [ffffa77fc02ebf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc02ebf50] ret_from_fork at ffffffff8fc04442

PID: 146    TASK: ffff9a2c402e0000  CPU: 1   COMMAND: "card0-crtc1"
 #0 [ffffa77fc02f3e38] __schedule at ffffffff904c0112
 #1 [ffffa77fc02f3ec8] schedule at ffffffff904c0746
 #2 [ffffa77fc02f3ee0] kthread_worker_fn at ffffffff8fcadb17
 #3 [ffffa77fc02f3f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc02f3f50] ret_from_fork at ffffffff8fc04442
```

```
PID: 147    TASK: ffff9a2c50efdf00  CPU: 1   COMMAND: "card0-crtc2"
 #0 [ffffa77fc02fbe38] __schedule at ffffffff904c0112
 #1 [ffffa77fc02fbec8] schedule at ffffffff904c0746
 #2 [ffffa77fc02fbee0] kthread_worker_fn at ffffffff8fcadb17
 #3 [ffffa77fc02fbf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc02fbf50] ret_from_fork at ffffffff8fc04442

PID: 148    TASK: ffff9a2c50efc740  CPU: 1   COMMAND: "card0-crtc3"
 #0 [ffffa77fc0303e38] __schedule at ffffffff904c0112
 #1 [ffffa77fc0303ec8] schedule at ffffffff904c0746
 #2 [ffffa77fc0303ee0] kthread_worker_fn at ffffffff8fcadb17
 #3 [ffffa77fc0303f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0303f50] ret_from_fork at ffffffff8fc04442

PID: 149    TASK: ffff9a2c50efaf80  CPU: 1   COMMAND: "card0-crtc4"
 #0 [ffffa77fc030be38] __schedule at ffffffff904c0112
 #1 [ffffa77fc030bec8] schedule at ffffffff904c0746
 #2 [ffffa77fc030bee0] kthread_worker_fn at ffffffff8fcadb17
 #3 [ffffa77fc030bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc030bf50] ret_from_fork at ffffffff8fc04442

PID: 150    TASK: ffff9a2c50ef8000  CPU: 1   COMMAND: "card0-crtc5"
 #0 [ffffa77fc0313e38] __schedule at ffffffff904c0112
 #1 [ffffa77fc0313ec8] schedule at ffffffff904c0746
 #2 [ffffa77fc0313ee0] kthread_worker_fn at ffffffff8fcadb17
 #3 [ffffa77fc0313f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0313f50] ret_from_fork at ffffffff8fc04442

PID: 151    TASK: ffff9a2c50ef97c0  CPU: 1   COMMAND: "card0-crtc6"
 #0 [ffffa77fc032be38] __schedule at ffffffff904c0112
 #1 [ffffa77fc032bec8] schedule at ffffffff904c0746
 #2 [ffffa77fc032bee0] kthread_worker_fn at ffffffff8fcadb17
 #3 [ffffa77fc032bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc032bf50] ret_from_fork at ffffffff8fc04442

PID: 152    TASK: ffff9a2c50f10000  CPU: 1   COMMAND: "card0-crtc7"
 #0 [ffffa77fc03f3e38] __schedule at ffffffff904c0112
 #1 [ffffa77fc03f3ec8] schedule at ffffffff904c0746
 #2 [ffffa77fc03f3ee0] kthread_worker_fn at ffffffff8fcadb17
 #3 [ffffa77fc03f3f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc03f3f50] ret_from_fork at ffffffff8fc04442

PID: 153    TASK: ffff9a2c50f117c0  CPU: 2   COMMAND: "kworker/2:1H"
 #0 [ffffa77fc0403e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc0403eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc0403ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc0403f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0403f50] ret_from_fork at ffffffff8fc04442

PID: 154    TASK: ffff9a2c5bdd8000  CPU: 3   COMMAND: "kworker/3:4"
 #0 [ffffa77fc01a7e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc01a7eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc01a7ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc01a7f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc01a7f50] ret_from_fork at ffffffff8fc04442

PID: 155    TASK: ffff9a2c5bdd97c0  CPU: 1   COMMAND: "kworker/1:2"
 #0 [ffffa77fc033be28] __schedule at ffffffff904c0112
 #1 [ffffa77fc033beb8] schedule at ffffffff904c0746
 #2 [ffffa77fc033bed0] worker_thread at ffffffff8fca6ba1
```

```
 #3 [ffffa77fc033bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc033bf50] ret_from_fork at ffffffff8fc04442

PID: 157    TASK: ffff9a2c517edf00  CPU: 0   COMMAND: "kworker/0:2"
 #0 [ffffa77fc034be28] __schedule at ffffffff904c0112
 #1 [ffffa77fc034beb8] schedule at ffffffff904c0746
 #2 [ffffa77fc034bed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc034bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc034bf50] ret_from_fork at ffffffff8fc04442

PID: 159    TASK: ffff9a2c517eaf80  CPU: 2   COMMAND: "kworker/2:2"
 #0 [ffffa77fc0363e28] __schedule at ffffffff904c0112
 #1 [ffffa77fc0363eb8] schedule at ffffffff904c0746
 #2 [ffffa77fc0363ed0] worker_thread at ffffffff8fca6ba1
 #3 [ffffa77fc0363f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0363f50] ret_from_fork at ffffffff8fc04442

PID: 197    TASK: ffff9a2c47ef2f80  CPU: 1   COMMAND: "jbd2/sda1-8"
 #0 [ffffa77fc036bdf8] __schedule at ffffffff904c0112
 #1 [ffffa77fc036be88] schedule at ffffffff904c0746
 #2 [ffffa77fc036bea0] kjournald2 at ffffffffc0598331 [jbd2]
 #3 [ffffa77fc036bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc036bf50] ret_from_fork at ffffffff8fc04442

PID: 198    TASK: ffff9a2c50f12f80  CPU: 1   COMMAND: "ext4-rsv-conver"
 #0 [ffffa77fc037bdf0] __schedule at ffffffff904c0112
 #1 [ffffa77fc037be80] schedule at ffffffff904c0746
 #2 [ffffa77fc037be98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc037bf10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc037bf50] ret_from_fork at ffffffff8fc04442

PID: 238    TASK: ffff9a2c403197c0  CPU: 1   COMMAND: "systemd-journal"
 #0 [ffffa77fc0187d60] __schedule at ffffffff904c0112
 #1 [ffffa77fc0187df0] schedule at ffffffff904c0746
 #2 [ffffa77fc0187e08] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc0187e78] do_epoll_wait at ffffffff8ff1a28a
 #4 [ffffa77fc0187f38] __x64_sys_epoll_wait at ffffffff8ff1a39a
 #5 [ffffa77fc0187f40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc0187f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007fad8a5f2116  RSP: 00007fff1f400280  RFLAGS: 00000293
    RAX: fffffffffffffda  RBX: 000055ebd69463c0  RCX: 00007fad8a5f2116
    RDX: 0000000000000063  RSI: 000055ebd69e62d0  RDI: 0000000000000008
    RBP: ffffffffffffffff   R8: 0000000000000000   R9: 00007fad8a97e000
    R10: 00000000ffffffff  R11: 0000000000000293  R12: 0000000000000001
    R13: 0000000000000063  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: 00000000000000e8  CS: 0033  SS: 002b

PID: 259    TASK: ffff9a2c59ca97c0  CPU: 0   COMMAND: "systemd-udevd"
 #0 [ffffa77fc01afd60] __schedule at ffffffff904c0112
 #1 [ffffa77fc01afdf0] schedule at ffffffff904c0746
 #2 [ffffa77fc01afe08] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc01afe78] do_epoll_wait at ffffffff8ff1a28a
 #4 [ffffa77fc01aff38] __x64_sys_epoll_wait at ffffffff8ff1a39a
 #5 [ffffa77fc01aff40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc01aff50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007ff2260c30d6  RSP: 00007ffdc1970768  RFLAGS: 00000246
    RAX: fffffffffffffda  RBX: 000055a8fe7f6bd0  RCX: 00007ff2260c30d6
    RDX: 000000000000000a  RSI: 000055a8fe98e8c0  RDI: 0000000000000009
    RBP: ffffffffffffffff   R8: 000000000000000a   R9: 000055a8fe9b0e24
    R10: 00000000ffffffff  R11: 0000000000000246  R12: 0000000000000001
```

```
    R13: 000000000000000a  R14: 000055a8fdbd02e6  R15: 0000000000000000
    ORIG_RAX: 00000000000000e8  CS: 0033  SS: 002b

PID: 336    TASK: ffff9a2c5a2d2f80  CPU: 2   COMMAND: "iprt-VBoxWQueue"
 #0 [ffffa77fc0217df0] __schedule at ffffffff904c0112
 #1 [ffffa77fc0217e80] schedule at ffffffff904c0746
 #2 [ffffa77fc0217e98] rescuer_thread at ffffffff8fca718c
 #3 [ffffa77fc0217f10] kthread at ffffffff8fcac91b
 #4 [ffffa77fc0217f50] ret_from_fork at ffffffff8fc04442

PID: 451    TASK: ffff9a2c51cb97c0  CPU: 1   COMMAND: "accounts-daemon"
 #0 [ffffa77fc062f9d8] __schedule at ffffffff904c0112
 #1 [ffffa77fc062fa68] schedule at ffffffff904c0746
 #2 [ffffa77fc062fa80] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc062faf0] do_sys_poll at ffffffff8fedb14b
 #4 [ffffa77fc062ff10] __x64_sys_poll at ffffffff8fedbd77
 #5 [ffffa77fc062ff40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc062ff50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f0ef8ac83ff  RSP: 00007ffea9db3a10  RFLAGS: 00000293
    RAX: ffffffffffffffda  RBX: 00007f0ef8c1b410  RCX: 00007f0ef8ac83ff
    RDX: 00000000ffffffff  RSI: 0000000000000001  RDI: 000055ea20d63d00
    RBP: 000055ea20d63d00   R8: 0000000000000000   R9: 0000000000000002
    R10: 0000000000000004  R11: 0000000000000293  R12: 0000000000000001
    R13: 00007ffea9db3a54  R14: 00000000ffffffff  R15: 000055ea20d53e70
    ORIG_RAX: 0000000000000007  CS: 0033  SS: 002b

PID: 454    TASK: ffff9a2c51cbdf00  CPU: 1   COMMAND: "avahi-daemon"
 #0 [ffffa77fc04bf9d8] __schedule at ffffffff904c0112
 #1 [ffffa77fc04bfa68] schedule at ffffffff904c0746
 #2 [ffffa77fc04bfa80] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc04bfaf0] do_sys_poll at ffffffff8fedb14b
 #4 [ffffa77fc04bff10] __x64_sys_poll at ffffffff8fedbd77
 #5 [ffffa77fc04bff40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc04bff50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f9d0e8f63c3  RSP: 00007fffd451e908  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 0000560c18eaf190  RCX: 00007f9d0e8f63c3
    RDX: 00000000ffffffff  RSI: 000000000000000a  RDI: 0000560c18eb8d10
    RBP: 00007f9d0e4d0740   R8: 0000000000000000   R9: 0000560c18eb3b80
    R10: 0000000000000000  R11: 0000000000000246  R12: 0000000000000000
    R13: 0000560c18eb0e60  R14: 0000560c18eb3ad0  R15: 0000000000000000
    ORIG_RAX: 0000000000000007  CS: 0033  SS: 002b

PID: 456    TASK: ffff9a2c51cb8000  CPU: 0   COMMAND: "cron"
 #0 [ffffa77fc02a3d90] __schedule at ffffffff904c0112
 #1 [ffffa77fc02a3e20] schedule at ffffffff904c0746
 #2 [ffffa77fc02a3e38] do_nanosleep at ffffffff904c34e1
 #3 [ffffa77fc02a3e80] hrtimer_nanosleep at ffffffff8fd1986b
 #4 [ffffa77fc02a3ef8] common_nsleep at ffffffff8fd21620
 #5 [ffffa77fc02a3f00] __x64_sys_clock_nanosleep at ffffffff8fd23c40
 #6 [ffffa77fc02a3f40] do_syscall_64 at ffffffff904b3883
 #7 [ffffa77fc02a3f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f40d150ac0a  RSP: 00007ffd334a1600  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: fffffffffffffe98  RCX: 00007f40d150ac0a
    RDX: 00007ffd334a1640  RSI: 0000000000000000  RDI: 0000000000000000
    RBP: 0000000000000002   R8: 0000000000000008   R9: 0000000000000004
    R10: 00007ffd334a1640  R11: 0000000000000246  R12: 0000000000000000
    R13: 0000000000000000  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: 00000000000000e6  CS: 0033  SS: 002b

PID: 459    TASK: ffff9a2c51cbaf80  CPU: 2   COMMAND: "dbus-daemon"
```

```
 #0 [ffffa77fc0243d60] __schedule at ffffffff904c0112
 #1 [ffffa77fc0243df0] schedule at ffffffff904c0746
 #2 [ffffa77fc0243e08] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc0243e78] do_epoll_wait at ffffffff8ff1a28a
 #4 [ffffa77fc0243f38] __x64_sys_epoll_wait at ffffffff8ff1a39a
 #5 [ffffa77fc0243f40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc0243f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007fa169b670d6  RSP: 00007ffd53dda4a8  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 00007ffd53dda840  RCX: 00007fa169b670d6
    RDX: 0000000000000040  RSI: 00007ffd53dda4b0  RDI: 0000000000000004
    RBP: 00007fa1696dfde8   R8: 0000000000000000   R9: 000055e051c6ab10
    R10: 00000000ffffffff  R11: 0000000000000246  R12: 000055e051cbec20
    R13: ffffffffffffffff  R14: 0000000000000000  R15: 0000000000000001
    ORIG_RAX: 00000000000000e8  CS: 0033  SS: 002b

PID: 465    TASK: ffff9a2c51cbc740  CPU: 2   COMMAND: "NetworkManager"
 #0 [ffffa77fc025f9d8] __schedule at ffffffff904c0112
 #1 [ffffa77fc025fa68] schedule at ffffffff904c0746
 #2 [ffffa77fc025fa80] schedule_hrtimeout_range_clock at ffffffff904c36e8
 #3 [ffffa77fc025faf0] do_sys_poll at ffffffff8fedb14b
 #4 [ffffa77fc025ff10] __x64_sys_poll at ffffffff8fedbdde
 #5 [ffffa77fc025ff40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc025ff50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f372bcf13ff  RSP: 00007ffc90a4d390  RFLAGS: 00000293
    RAX: ffffffffffffffda  RBX: 00007f372be4c410  RCX: 00007f372bcf13ff
    RDX: 0000000000008d3e  RSI: 0000000000000009  RDI: 0000559ea5f09bd0
    RBP: 0000559ea5f09bd0   R8: 0000000000000000   R9: 0000000000000001
    R10: 0000559ea5e66cd0  R11: 0000000000000293  R12: 0000000000000009
    R13: 00007ffc90a4d3d4  R14: 0000000000008d3e  R15: 0000559ea5e68de0
    ORIG_RAX: 0000000000000007  CS: 0033  SS: 002b

PID: 467    TASK: ffff9a2c591d4740  CPU: 1   COMMAND: "gmain"
 #0 [ffffa77fc061f9d8] __schedule at ffffffff904c0112
 #1 [ffffa77fc061fa68] schedule at ffffffff904c0746
 #2 [ffffa77fc061fa80] schedule_hrtimeout_range_clock at ffffffff904c36e8
 #3 [ffffa77fc061faf0] do_sys_poll at ffffffff8fedb14b
 #4 [ffffa77fc061ff10] __x64_sys_poll at ffffffff8fedbdde
 #5 [ffffa77fc061ff40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc061ff50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f0ef8ac83ff  RSP: 00007f0ef815dcb0  RFLAGS: 00000293
    RAX: ffffffffffffffda  RBX: 00007f0ef8c1b410  RCX: 00007f0ef8ac83ff
    RDX: 0000000000000064  RSI: 0000000000000002  RDI: 000055ea20d564d0
    RBP: 000055ea20d564d0   R8: 0000000000000000   R9: 00007f0ef8ce7280
    R10: 00007ffea9df2080  R11: 0000000000000293  R12: 0000000000000002
    R13: 00007f0ef815dcf4  R14: 0000000000000064  R15: 000055ea20d59020
    ORIG_RAX: 0000000000000007  CS: 0033  SS: 002b

PID: 478    TASK: ffff9a2c475e0000  CPU: 3   COMMAND: "polkitd"
 #0 [ffffa77fc03a79d8] __schedule at ffffffff904c0112
 #1 [ffffa77fc03a7a68] schedule at ffffffff904c0746
 #2 [ffffa77fc03a7a80] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc03a7af0] do_sys_poll at ffffffff8fedb14b
 #4 [ffffa77fc03a7f10] __x64_sys_poll at ffffffff8fedbd77
 #5 [ffffa77fc03a7f40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc03a7f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f463f7d93ff  RSP: 00007fffcd3be100  RFLAGS: 00000293
    RAX: ffffffffffffffda  RBX: 00007f463fa14410  RCX: 00007f463f7d93ff
    RDX: 00000000ffffffff  RSI: 0000000000000003  RDI: 0000564a061e4ed0
    RBP: 0000564a061e4ed0   R8: 0000000000000000   R9: 0000000000000002
    R10: 0000000000000018  R11: 0000000000000293  R12: 0000000000000003
```

```
    R13: 00007fffcd3be144  R14: 00000000ffffffff  R15: 0000564a061bd070
    ORIG_RAX: 0000000000000007  CS: 0033  SS: 002b

PID: 479    TASK: ffff9a2c59cac740  CPU: 2   COMMAND: "rsyslogd"
 #0 [ffffa77fc042b888] __schedule at ffffffff904c0112
 #1 [ffffa77fc042b918] schedule at ffffffff904c0746
 #2 [ffffa77fc042b930] schedule_hrtimeout_range_clock at ffffffff904c36e8
 #3 [ffffa77fc042b9a0] do_select at ffffffff8feda19b
 #4 [ffffa77fc042bd30] core_sys_select at ffffffff8fedc012
 #5 [ffffa77fc042beb8] do_pselect.constprop.0 at ffffffff8fedc4ba
 #6 [ffffa77fc042bf30] __x64_sys_pselect6 at ffffffff8fedc624
 #7 [ffffa77fc042bf40] do_syscall_64 at ffffffff904b3883
 #8 [ffffa77fc042bf50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007ff410c549c6  RSP: 00007ffed4c66470  RFLAGS: 00000293
    RAX: ffffffffffffffda  RBX: 0000000000000000  RCX: 00007ff410c549c6
    RDX: 0000000000000000  RSI: 0000000000000000  RDI: 0000000000000000
    RBP: 0000000000000000   R8: 00007ffed4c664a0   R9: 00007ffed4c664b0
    R10: 0000000000000000  R11: 0000000000000293  R12: 00007ffed4c66580
    R13: 00007ffed4c66500  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: 000000000000010e  CS: 0033  SS: 002b

PID: 481    TASK: ffff9a2c59caaf80  CPU: 1   COMMAND: "switcheroo-cont"
 #0 [ffffa77fc044b9d8] __schedule at ffffffff904c0112
 #1 [ffffa77fc044ba68] schedule at ffffffff904c0746
 #2 [ffffa77fc044ba80] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc044baf0] do_sys_poll at ffffffff8fedb14b
 #4 [ffffa77fc044bf10] __x64_sys_poll at ffffffff8fedbd77
 #5 [ffffa77fc044bf40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc044bf50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007fc934bfd3ff  RSP: 00007ffcf1a7f770  RFLAGS: 00000293
    RAX: ffffffffffffffda  RBX: 00007fc934f75410  RCX: 00007fc934bfd3ff
    RDX: 00000000ffffffff  RSI: 0000000000000002  RDI: 000056155d363380
    RBP: 000056155d363380   R8: 0000000000000000   R9: 0000000000000002
    R10: 000000000000001e  R11: 0000000000000293  R12: 0000000000000002
    R13: 00007ffcf1a7f7b4  R14: 00000000ffffffff  R15: 000056155d356c30
    ORIG_RAX: 0000000000000007  CS: 0033  SS: 002b

PID: 484    TASK: ffff9a2c59cadf00  CPU: 1   COMMAND: "systemd-logind"
 #0 [ffffa77fc028bd60] __schedule at ffffffff904c0112
 #1 [ffffa77fc028bdf0] schedule at ffffffff904c0746
 #2 [ffffa77fc028be08] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc028be78] do_epoll_wait at ffffffff8ff1a28a
 #4 [ffffa77fc028bf38] __x64_sys_epoll_wait at ffffffff8ff1a39a
 #5 [ffffa77fc028bf40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc028bf50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f898b83c116  RSP: 00007ffe9bab3020  RFLAGS: 00000293
    RAX: ffffffffffffffda  RBX: 00005638b49cf380  RCX: 00007f898b83c116
    RDX: 0000000000000019  RSI: 00005638b49dc410  RDI: 0000000000000004
    RBP: ffffffffffffffff   R8: 0000000000000000   R9: 0000000000000016
    R10: 00000000ffffffff  R11: 0000000000000293  R12: 0000000000000001
    R13: 0000000000000019  R14: 00007ffe9bab3248  R15: 00007ffe9bab3260
    ORIG_RAX: 00000000000000e8  CS: 0033  SS: 002b

PID: 485    TASK: ffff9a2c59ca8000  CPU: 2   COMMAND: "udisksd"
 #0 [ffffa77fc02939d8] __schedule at ffffffff904c0112
 #1 [ffffa77fc0293a68] schedule at ffffffff904c0746
 #2 [ffffa77fc0293a80] schedule_hrtimeout_range_clock at ffffffff904c36e8
 #3 [ffffa77fc0293af0] do_sys_poll at ffffffff8fedb14b
 #4 [ffffa77fc0293f10] __x64_sys_poll at ffffffff8fedbdde
 #5 [ffffa77fc0293f40] do_syscall_64 at ffffffff904b3883
```

```
 #6 [ffffa77fc0293f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f27a38a43ff  RSP: 00007ffcef9c7040  RFLAGS: 00000293
    RAX: ffffffffffffffda  RBX: 00007f27a39f9410  RCX: 00007f27a38a43ff
    RDX: 0000000000089fa4  RSI: 0000000000000006  RDI: 0000562f64e0fd30
    RBP: 0000562f64e0fd30   R8: 0000000000000000   R9: 0000000000000002
    R10: 00007ffcef9db080  R11: 0000000000000293  R12: 0000000000000006
    R13: 00007ffcef9c7084  R14: 0000000000089fa4  R15: 0000562f64d8a040
    ORIG_RAX: 0000000000000007  CS: 0033  SS: 002b

PID: 487    TASK: ffff9a2c58f34740  CPU: 3   COMMAND: "wpa_supplicant"
 #0 [ffffa77fc0463890] __schedule at ffffffff904c0112
 #1 [ffffa77fc0463920] schedule at ffffffff904c0746
 #2 [ffffa77fc0463938] schedule_hrtimeout_range_clock at ffffffff904c36e8
 #3 [ffffa77fc04639a8] do_select at ffffffff8feda19b
 #4 [ffffa77fc0463d38] core_sys_select at ffffffff8fedc012
 #5 [ffffa77fc0463ec0] kern_select at ffffffff8fedc2ed
 #6 [ffffa77fc0463f38] __x64_sys_select at ffffffff8fedc3b1
 #7 [ffffa77fc0463f40] do_syscall_64 at ffffffff904b3883
 #8 [ffffa77fc0463f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f99a984a866  RSP: 00007ffdeb7d3528  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 000055cd4153bc50  RCX: 00007f99a984a866
    RDX: 000055cd4153b210  RSI: 000055cd4153b180  RDI: 0000000000000005
    RBP: 000055cd40945df0   R8: 00007ffdeb7d3550   R9: 0000000000000012
    R10: 000055cd4153b2a0  R11: 0000000000000246  R12: 000055cd4153b2a0
    R13: 000055cd4153b210  R14: 000055cd4153b180  R15: 000055cd40945da0
    ORIG_RAX: 0000000000000017  CS: 0033  SS: 002b

PID: 489    TASK: ffff9a2c4746df00  CPU: 2   COMMAND: "gmain"
 #0 [ffffa77fc048f9d8] __schedule at ffffffff904c0112
 #1 [ffffa77fc048fa68] schedule at ffffffff904c0746
 #2 [ffffa77fc048fa80] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc048faf0] do_sys_poll at ffffffff8fedb14b
 #4 [ffffa77fc048ff10] __x64_sys_poll at ffffffff8fedbd77
 #5 [ffffa77fc048ff40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc048ff50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f463f7d93ff  RSP: 00007f463f22ecb0  RFLAGS: 00000293
    RAX: ffffffffffffffda  RBX: 00007f463fa14410  RCX: 00007f463f7d93ff
    RDX: 00000000ffffffff  RSI: 0000000000000002  RDI: 0000564a061b7830
    RBP: 0000564a061b7830   R8: 0000000000000000   R9: 00007f4630000080
    R10: 0000000000004022  R11: 0000000000000293  R12: 0000000000000002
    R13: 00007f463f22ecf4  R14: 00000000ffffffff  R15: 0000564a061c0250
    ORIG_RAX: 0000000000000007  CS: 0033  SS: 002b

PID: 490    TASK: ffff9a2c58e15f00  CPU: 3   COMMAND: "avahi-daemon"
 #0 [ffffa77fc01efbb0] __schedule at ffffffff904c0112
 #1 [ffffa77fc01efc40] schedule at ffffffff904c0746
 #2 [ffffa77fc01efc58] schedule_timeout at ffffffff904c33af
 #3 [ffffa77fc01efcb0] unix_stream_read_generic at ffffffff903fafdd
 #4 [ffffa77fc01efd90] unix_stream_recvmsg at ffffffff903fb3b3
 #5 [ffffa77fc01efdd0] sock_read_iter at ffffffff902b7c22
 #6 [ffffa77fc01efe48] new_sync_read at ffffffff8febf0ea
 #7 [ffffa77fc01efed0] vfs_read at ffffffff8fec1c84
 #8 [ffffa77fc01eff08] ksys_read at ffffffff8fec22f7
 #9 [ffffa77fc01eff40] do_syscall_64 at ffffffff904b3883
#10 [ffffa77fc01eff50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007f9d0e9db04e  RSP: 00007fffd451e868  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 0000000000000006  RCX: 00007f9d0e9db04e
    RDX: 0000000000000001  RSI: 00007fffd451e886  RDI: 0000000000000006
    RBP: 00007fffd451e8a0   R8: 0000000000000000   R9: 0000000000000005
    R10: 0000000000000000  R11: 0000000000000246  R12: 00007fffd451e886
```

```
    R13: 0000560c183fe040  R14: 0000560c183fdfc0  R15: 0000000000000000
    ORIG_RAX: 0000000000000000  CS: 0033  SS: 002b

PID: 494    TASK: ffff9a2c50220000  CPU: 1   COMMAND: "in:imuxsock"
 #0 [ffffa77fc06279d8] __schedule at ffffffff904c0112
 #1 [ffffa77fc0627a68] schedule at ffffffff904c0746
 #2 [ffffa77fc0627a80] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc0627af0] do_sys_poll at ffffffff8fedb14b
 #4 [ffffa77fc0627f10] __x64_sys_poll at ffffffff8fedbd77
 #5 [ffffa77fc0627f40] do_syscall_64 at ffffffff904b3883
 #6 [ffffa77fc0627f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007ff410c523ff  RSP: 00007ff4107f5cf0  RFLAGS: 00000293
    RAX: fffffffffffffda  RBX: 0000000000000002  RCX: 00007ff410c523ff
    RDX: 00000000ffffffff  RSI: 0000000000000001  RDI: 00007ff400000b80
    RBP: 0000000000000058   R8: 0000000000000000   R9: 00005557ca0bda08
    R10: 0000000000000000  R11: 0000000000000293  R12: 0000000000000000
    R13: 00007ff400000b80  R14: 00005557c88b5a50  R15: 00007ff410e46840
    ORIG_RAX: 0000000000000007  CS: 0033  SS: 002b

PID: 495    TASK: ffff9a2c50222f80  CPU: 3   COMMAND: "in:imklog"
 #0 [ffffa77fc0473d58] __schedule at ffffffff904c0112
 #1 [ffffa77fc0473de8] schedule at ffffffff904c0746
 #2 [ffffa77fc0473e00] do_syslog at ffffffff8fcf2fe3
 #3 [ffffa77fc0473eb8] kmsg_read at ffffffff8ff758be
 #4 [ffffa77fc0473ed0] vfs_read at ffffffff8fec1c28
 #5 [ffffa77fc0473f08] ksys_read at ffffffff8fec22af
 #6 [ffffa77fc0473f40] do_syscall_64 at ffffffff904b3883
 #7 [ffffa77fc0473f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007ff410e1008c  RSP: 00007ff4103d44d0  RFLAGS: 00000246
    RAX: fffffffffffffda  RBX: 0000000000000000  RCX: 00007ff410e1008c
    RDX: 0000000000001fa0  RSI: 00007ff4103d4d00  RDI: 0000000000000005
    RBP: 00005557ca0b3920   R8: 0000000000000000   R9: 00005557ca0bda08
    R10: 0000000000000000  R11: 0000000000000246  R12: 00007ff4103d4d00
    R13: 0000000000001fa0  R14: 00007ff4103d4d00  R15: 00007ff4103d4d31
    ORIG_RAX: 0000000000000000  CS: 0033  SS: 002b

PID: 499    TASK: ffff9a2c50225f00  CPU: 1   COMMAND: "rs:main Q:Reg"
 #0 [ffffa77fc0443bb8] __schedule at ffffffff904c0112
 #1 [ffffa77fc0443c48] schedule at ffffffff904c0746
 #2 [ffffa77fc0443c60] futex_wait_queue_me at ffffffff8fd2c9a6
 #3 [ffffa77fc0443c98] futex_wait at ffffffff8fd2d4f9
 #4 [ffffa77fc0443db0] do_futex at ffffffff8fd2f324
 #5 [ffffa77fc0443ed0] __x64_sys_futex at ffffffff8fd30416
 #6 [ffffa77fc0443f40] do_syscall_64 at ffffffff904b3883
 #7 [ffffa77fc0443f50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007ff410e0c7b2  RSP: 00007ff40fff3ac0  RFLAGS: 00000282
    RAX: fffffffffffffda  RBX: 0000000000000524  RCX: 00007ff410e0c7b2
    RDX: 0000000000000000  RSI: 0000000000000080  RDI: 00005557ca0bda10
    RBP: 00005557ca0bd9e8   R8: 0000000000000001   R9: 0000000000000001
    R10: 0000000000000000  R11: 0000000000000282  R12: 0000000000000000
    R13: 00005557ca0bd7d0  R14: 00005557ca0bda10  R15: 00007ff40fff3af0
    ORIG_RAX: 00000000000000ca  CS: 0033  SS: 002b

PID: 500    TASK: ffff9a2c475ec740  CPU: 2   COMMAND: "gmain"
 #0 [ffffa77fc047f9d8] __schedule at ffffffff904c0112
 #1 [ffffa77fc047fa68] schedule at ffffffff904c0746
 #2 [ffffa77fc047fa80] schedule_hrtimeout_range_clock at ffffffff904c3760
 #3 [ffffa77fc047faf0] do_sys_poll at ffffffff8fedb14b
 #4 [ffffa77fc047ff10] __x64_sys_poll at ffffffff8fedbd77
 #5 [ffffa77fc047ff40] do_syscall_64 at ffffffff904b3883
```

```
 #6 [ffffa77fc047ff50] entry_SYSCALL_64_after_hwframe at ffffffff9060008c
    RIP: 00007fc934bfd3ff  RSP: 00007fc93459ad30  RFLAGS: 00000293
    RAX: fffffffffffffffda  RBX: 00007fc934f75410  RCX: 00007fc934bfd3ff
    RDX: 00000000ffffffff  RSI: 0000000000000001  RDI: 000056155d35be70
    RBP: 000056155d35be70   R8: 0000000000000000   R9: 00007fc924000080
    R10: 0000000000004022  R11: 0000000000000293  R12: 0000000000000001
    R13: 00007fc93459ad74  R14: 00000000ffffffff  R15: 000056155d35b710
    ORIG_RAX: 0000000000000007  CS: 0033  SS: 002b

 -- MORE --  forward: <SPACE>, <ENTER> or j  backward: b or k  quit: qq
```

16. Individual process/thread (task) structures of interest can also be explored:

```
crash> task ffff9a2c50225f00
PID: 499    TASK: ffff9a2c50225f00  CPU: 1   COMMAND: "rs:main Q:Reg"
struct task_struct {
  thread_info = {
    flags = 0,
    status = 0
  },
  state = 1,
  stack = 0xffffa77fc0440000,
  usage = {
    refs = {
      counter = 1
    }
  },
  flags = 1077936192,
  ptrace = 0,
  on_cpu = 0,
  wake_entry = {
    llist = {
      next = 0x0
    },
    {
      u_flags = 48,
      a_flags = {
        counter = 48
      }
    },
    src = 0,
    dst = 0
  },
  cpu = 1,
  wakee_flips = 0,
  wakee_flip_decay_ts = 4294961286,
  last_wakee = 0xffff9a2c591d4740,
  recent_used_cpu = 1,
  wake_cpu = 1,
  on_rq = 0,
  prio = 120,
  static_prio = 120,
  normal_prio = 120,
  rt_priority = 0,
  sched_class = 0xffffffff90d74c60 <fair_sched_class>,
  se = {
    load = {
      weight = 1048576,
      inv_weight = 4194304
    },
```

```
    run_node = {
      __rb_parent_color = 18446632113579071568,
      rb_right = 0x0,
      rb_left = 0x0
    },
    group_node = {
      next = 0xffff9a2c50225fa8,
      prev = 0xffff9a2c50225fa8
    },
    on_rq = 0,
    exec_start = 276080392731,
-- MORE --  forward: <SPACE>, <ENTER> or j  backward: b or k  quit: qq
```

Exercise K2

- **Goal:** Learn how to navigate a problem kernel dump
- **Patterns:** Exception Stack Trace; NULL Pointer (Data); Execution Residue (Kernel Space); Value References
- \ALCDA-Dumps\Exercise-K2-x64-GDB.pdf

Exercise K2 (x64, GDB)

Goal: Learn how to navigate a problem kernel dump.

Patterns: Exception Stack Trace; NULL Pointer (Data); Execution Residue (Kernel Space); Value References.

1. Load a core dump *dump.202201020022* from the x64/K2 directory and the matching *vmlinux-5.10.0-10-amd64* file from the x64/KSym directory:

```
~/ALCDA2/x64/K2$ crash dump.202201020022 ../KSym/vmlinux-5.10.0-10-amd64

crash 8.0.0++
Copyright (C) 2002-2021  Red Hat, Inc.
Copyright (C) 2004, 2005, 2006, 2010  IBM Corporation
Copyright (C) 1999-2006  Hewlett-Packard Co
Copyright (C) 2005, 2006, 2011, 2012  Fujitsu Limited
Copyright (C) 2006, 2007  VA Linux Systems Japan K.K.
Copyright (C) 2005, 2011, 2020-2021  NEC Corporation
Copyright (C) 1999, 2002, 2007  Silicon Graphics, Inc.
Copyright (C) 1999, 2000, 2001, 2002  Mission Critical Linux, Inc.
Copyright (C) 2015, 2021  VMware, Inc.
This program is free software, covered by the GNU General Public License,
and you are welcome to change it and/or distribute copies of it under
certain conditions.  Enter "help copying" to see the conditions.
This program has absolutely no warranty.  Enter "help warranty" for details.

GNU gdb (GDB) 10.2
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-pc-linux-gnu".
Type "show configuration" for configuration details.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...

      KERNEL: ../KSym/vmlinux-5.10.0-10-amd64  [TAINTED]
    DUMPFILE: dump.202201020022  [PARTIAL DUMP]
        CPUS: 4
        DATE: Sun Jan  2 00:19:33 GMT 2022
      UPTIME: 00:33:31
LOAD AVERAGE: 0.09, 0.07, 0.08
       TASKS: 454
    NODENAME: coredump
     RELEASE: 5.10.0-10-amd64
     VERSION: #1 SMP Debian 5.10.84-1 (2021-12-08)
     MACHINE: x86_64  (1991 Mhz)
      MEMORY: 4 GB
       PANIC: "Oops: 0002 [#1] SMP PTI" (check log for details)
         PID: 3926
     COMMAND: "insmod"
        TASK: ffff8a5b4430af80  [THREAD_INFO: ffff8a5b4430af80]
         CPU: 2
```

```
      STATE: TASK_RUNNING (PANIC)

crash>
```

2. We follow the suggestion to check the log for details, and at the end, we find the bug description, crash RIP that points to the problem source code, the stack pointer, and the stack trace:

```
crash> log -T
[Sat Jan  1 23:46:02 GMT 2022] Linux version 5.10.0-10-amd64 (debian-kernel@lists.debian.org) (gcc-10 (Debian 10.2.1-
6) 10.2.1 20210110, GNU ld (GNU Binutils for Debian) 2.35.2) #1 SMP Debian 5.10.84-1 (2021-12-08)
[Sat Jan  1 23:46:02 GMT 2022] Command line: BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64 root=UUID=9cc5ee1e-5533-4a0b-
a88f-903bf52d812d ro quiet crashkernel=384M-:128M
[Sat Jan  1 23:46:02 GMT 2022] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[Sat Jan  1 23:46:02 GMT 2022] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[Sat Jan  1 23:46:02 GMT 2022] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[Sat Jan  1 23:46:02 GMT 2022] x86/fpu: xstate_offset[2]:  576, xstate_sizes[2]:  256
[Sat Jan  1 23:46:02 GMT 2022] x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using 'standard'
format.
[Sat Jan  1 23:46:02 GMT 2022] BIOS-provided physical RAM map:
[Sat Jan  1 23:46:02 GMT 2022] BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
[Sat Jan  1 23:46:02 GMT 2022] BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved
[Sat Jan  1 23:46:02 GMT 2022] BIOS-e820: [mem 0x00000000000f0000-0x00000000000fffff] reserved
[Sat Jan  1 23:46:02 GMT 2022] BIOS-e820: [mem 0x0000000000100000-0x00000000dffeffff] usable
[Sat Jan  1 23:46:02 GMT 2022] BIOS-e820: [mem 0x00000000dfff0000-0x00000000dfffffff] ACPI data
[Sat Jan  1 23:46:02 GMT 2022] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec00fff] reserved
[Sat Jan  1 23:46:02 GMT 2022] BIOS-e820: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
[Sat Jan  1 23:46:02 GMT 2022] BIOS-e820: [mem 0x00000000fffc0000-0x00000000ffffffff] reserved
[Sat Jan  1 23:46:02 GMT 2022] BIOS-e820: [mem 0x0000000100000000-0x000000011fffffff] usable
[Sat Jan  1 23:46:02 GMT 2022] NX (Execute Disable) protection: active
[Sat Jan  1 23:46:02 GMT 2022] SMBIOS 2.5 present.
[Sat Jan  1 23:46:02 GMT 2022] DMI: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jan  1 23:46:02 GMT 2022] Hypervisor detected: KVM
[Sat Jan  1 23:46:02 GMT 2022] kvm-clock: Using msrs 4b564d01 and 4b564d00
[Sat Jan  1 23:46:02 GMT 2022] kvm-clock: cpu 0, msr 2a0b7001, primary cpu clock
[Sat Jan  1 23:46:02 GMT 2022] kvm-clock: using sched offset of 5343422896 cycles
[Sat Jan  1 23:46:02 GMT 2022] clocksource: kvm-clock: mask: 0xffffffffffffffff max_cycles: 0x1cd42e4dffb,
max_idle_ns: 881590591483 ns
[Sat Jan  1 23:46:02 GMT 2022] tsc: Detected 1991.997 MHz processor
[Sat Jan  1 23:46:02 GMT 2022] e820: update [mem 0x00000000-0x00000fff] usable ==> reserved
[Sat Jan  1 23:46:02 GMT 2022] e820: remove [mem 0x000a0000-0x000fffff] usable
[Sat Jan  1 23:46:02 GMT 2022] last_pfn = 0x120000 max_arch_pfn = 0x400000000
[Sat Jan  1 23:46:02 GMT 2022] MTRR default type: uncachable
[Sat Jan  1 23:46:02 GMT 2022] MTRR variable ranges disabled:
[Sat Jan  1 23:46:02 GMT 2022] Disabled
[Sat Jan  1 23:46:02 GMT 2022] x86/PAT: MTRRs disabled, skipping PAT initialization too.
[Sat Jan  1 23:46:02 GMT 2022] CPU MTRRs all blank - virtualized system.
[Sat Jan  1 23:46:02 GMT 2022] x86/PAT: Configuration [0-7]: WB  WT  UC- UC  WB  WT  UC- UC
[Sat Jan  1 23:46:02 GMT 2022] last_pfn = 0xdfff0 max_arch_pfn = 0x400000000
[Sat Jan  1 23:46:02 GMT 2022] found SMP MP-table at [mem 0x0009fff0-0x0009ffff]
[Sat Jan  1 23:46:02 GMT 2022] kexec: Reserving the low 1M of memory for crashkernel
[Sat Jan  1 23:46:02 GMT 2022] RAMDISK: [mem 0x32ec7000-0x3575afff]
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Early table checksum verification disabled
[Sat Jan  1 23:46:02 GMT 2022] ACPI: RSDP 0x00000000000E0000 000024 (v02 VBOX  )
[Sat Jan  1 23:46:02 GMT 2022] ACPI: XSDT 0x00000000DFFF0030 00003C (v01 VBOX   VBOXXSDT 00000001 ASL  00000061)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: FACP 0x00000000DFFF00F0 0000F4 (v04 VBOX   VBOXFACP 00000001 ASL  00000061)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: DSDT 0x00000000DFFF0480 002325 (v02 VBOX   VBOXBIOS 00000002 INTL 20190509)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: FACS 0x00000000DFFF0200 000040
[Sat Jan  1 23:46:02 GMT 2022] ACPI: FACS 0x00000000DFFF0200 000040
[Sat Jan  1 23:46:02 GMT 2022] ACPI: APIC 0x00000000DFFF0240 00006C (v02 VBOX   VBOXAPIC 00000001 ASL  00000061)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: SSDT 0x00000000DFFF02B0 0001CC (v01 VBOX   VBOXCPUT 00000002 INTL 20190509)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Reserving FACP table memory at [mem 0xdfff00f0-0xdfff01e3]
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Reserving DSDT table memory at [mem 0xdfff0480-0xdfff27a4]
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Reserving FACS table memory at [mem 0xdfff0200-0xdfff023f]
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Reserving FACS table memory at [mem 0xdfff0200-0xdfff023f]
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Reserving APIC table memory at [mem 0xdfff0240-0xdfff02ab]
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Reserving SSDT table memory at [mem 0xdfff02b0-0xdfff047b]
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Local APIC address 0xfee00000
[Sat Jan  1 23:46:02 GMT 2022] No NUMA configuration found
[Sat Jan  1 23:46:02 GMT 2022] Faking a node at [mem 0x0000000000000000-0x000000011fffffff]
[Sat Jan  1 23:46:02 GMT 2022] NODE_DATA(0) allocated [mem 0x11ffd2000-0x11fffbfff]
[Sat Jan  1 23:46:02 GMT 2022] Reserving 128MB of memory at 3440MB for crashkernel (System RAM: 4095MB)
[Sat Jan  1 23:46:02 GMT 2022] Zone ranges:
```

```
[Sat Jan  1 23:46:02 GMT 2022]   DMA      [mem 0x0000000000001000-0x0000000000ffffff]
[Sat Jan  1 23:46:02 GMT 2022]   DMA32    [mem 0x0000000001000000-0x00000000ffffffff]
[Sat Jan  1 23:46:02 GMT 2022]   Normal   [mem 0x0000000100000000-0x000000011fffffff]
[Sat Jan  1 23:46:02 GMT 2022]   Device   empty
[Sat Jan  1 23:46:02 GMT 2022] Movable zone start for each node
[Sat Jan  1 23:46:02 GMT 2022] Early memory node ranges
[Sat Jan  1 23:46:02 GMT 2022]   node   0: [mem 0x0000000000001000-0x000000000009efff]
[Sat Jan  1 23:46:02 GMT 2022]   node   0: [mem 0x0000000000100000-0x00000000dffeffff]
[Sat Jan  1 23:46:02 GMT 2022]   node   0: [mem 0x0000000100000000-0x000000011fffffff]
[Sat Jan  1 23:46:02 GMT 2022] Initmem setup node 0 [mem 0x0000000000001000-0x000000011fffffff]
[Sat Jan  1 23:46:02 GMT 2022] On node 0 totalpages: 1048462
[Sat Jan  1 23:46:02 GMT 2022]   DMA zone: 64 pages used for memmap
[Sat Jan  1 23:46:02 GMT 2022]   DMA zone: 158 pages reserved
[Sat Jan  1 23:46:02 GMT 2022]   DMA zone: 3998 pages, LIFO batch:0
[Sat Jan  1 23:46:02 GMT 2022]   DMA32 zone: 14272 pages used for memmap
[Sat Jan  1 23:46:02 GMT 2022]   DMA32 zone: 913392 pages, LIFO batch:63
[Sat Jan  1 23:46:02 GMT 2022]   Normal zone: 2048 pages used for memmap
[Sat Jan  1 23:46:02 GMT 2022]   Normal zone: 131072 pages, LIFO batch:31
[Sat Jan  1 23:46:02 GMT 2022] On node 0, zone DMA: 1 pages in unavailable ranges
[Sat Jan  1 23:46:02 GMT 2022] On node 0, zone DMA: 97 pages in unavailable ranges
[Sat Jan  1 23:46:02 GMT 2022] On node 0, zone Normal: 16 pages in unavailable ranges
[Sat Jan  1 23:46:02 GMT 2022] ACPI: PM-Timer IO Port: 0x4008
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Local APIC address 0xfee00000
[Sat Jan  1 23:46:02 GMT 2022] IOAPIC[0]: apic_id 4, version 32, address 0xfec00000, GSI 0-23
[Sat Jan  1 23:46:02 GMT 2022] ACPI: INT_SRC_OVR (bus 0 bus_irq 0 global_irq 2 dfl dfl)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: INT_SRC_OVR (bus 0 bus_irq 9 global_irq 9 low level)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: IRQ0 used by override.
[Sat Jan  1 23:46:02 GMT 2022] ACPI: IRQ9 used by override.
[Sat Jan  1 23:46:02 GMT 2022] Using ACPI (MADT) for SMP configuration information
[Sat Jan  1 23:46:02 GMT 2022] smpboot: Allowing 4 CPUs, 0 hotplug CPUs
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0x00000000-0x00000fff]
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0x0009f000-0x0009ffff]
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0x000a0000-0x000effff]
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0x000f0000-0x000fffff]
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0xdfff0000-0xdfffffff]
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0xe0000000-0xfebfffff]
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0xfec00000-0xfec00fff]
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0xfec01000-0xfedfffff]
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0xfee00000-0xfee00fff]
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0xfee01000-0xfffbffff]
[Sat Jan  1 23:46:02 GMT 2022] PM: hibernation: Registered nosave memory: [mem 0xfffc0000-0xffffffff]
[Sat Jan  1 23:46:02 GMT 2022] [mem 0xe0000000-0xfebfffff] available for PCI devices
[Sat Jan  1 23:46:02 GMT 2022] Booting paravirtualized kernel on KVM
[Sat Jan  1 23:46:02 GMT 2022] clocksource: refined-jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns:
7645519600211568 ns
[Sat Jan  1 23:46:02 GMT 2022] setup_percpu: NR_CPUS:8192 nr_cpumask_bits:4 nr_cpu_ids:4 nr_node_ids:1
[Sat Jan  1 23:46:02 GMT 2022] percpu: Embedded 58 pages/cpu s200536 r8192 d28840 u524288
[Sat Jan  1 23:46:02 GMT 2022] pcpu-alloc: s200536 r8192 d28840 u524288 alloc=1*2097152
[Sat Jan  1 23:46:02 GMT 2022] pcpu-alloc: [0] 0 1 2 3
[Sat Jan  1 23:46:02 GMT 2022] kvm-guest: PV spinlocks enabled
[Sat Jan  1 23:46:02 GMT 2022] PV qspinlock hash table entries: 256 (order: 0, 4096 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] Built 1 zonelists, mobility grouping on.  Total pages: 1031920
[Sat Jan  1 23:46:02 GMT 2022] Policy zone: Normal
[Sat Jan  1 23:46:02 GMT 2022] Kernel command line: BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64 root=UUID=9cc5ee1e-5533-
4a0b-a88f-903bf52d812d ro quiet crashkernel=384M-:128M
[Sat Jan  1 23:46:02 GMT 2022] Dentry cache hash table entries: 524288 (order: 10, 4194304 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] Inode-cache hash table entries: 262144 (order: 9, 2097152 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] mem auto-init: stack:off, heap alloc:on, heap free:off
[Sat Jan  1 23:46:02 GMT 2022] Memory: 3526712K/4193848K available (12295K kernel code, 2545K rwdata, 7564K rodata,
2408K init, 3684K bss, 346912K reserved, 0K cma-reserved)
[Sat Jan  1 23:46:02 GMT 2022] random: get_random_u64 called from __kmem_cache_create+0x2a/0x4d0 with crng_init=0
[Sat Jan  1 23:46:02 GMT 2022] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=4, Nodes=1
[Sat Jan  1 23:46:02 GMT 2022] Kernel/User page tables isolation: enabled
[Sat Jan  1 23:46:02 GMT 2022] ftrace: allocating 36444 entries in 143 pages
[Sat Jan  1 23:46:02 GMT 2022] ftrace: allocated 143 pages with 5 groups
[Sat Jan  1 23:46:02 GMT 2022] rcu: Hierarchical RCU implementation.
[Sat Jan  1 23:46:02 GMT 2022] rcu:     RCU restricting CPUs from NR_CPUS=8192 to nr_cpu_ids=4.
[Sat Jan  1 23:46:02 GMT 2022]  Rude variant of Tasks RCU enabled.
[Sat Jan  1 23:46:02 GMT 2022]  Tracing variant of Tasks RCU enabled.
[Sat Jan  1 23:46:02 GMT 2022] rcu: RCU calculated value of scheduler-enlistment delay is 25 jiffies.
[Sat Jan  1 23:46:02 GMT 2022] rcu: Adjusting geometry for rcu_fanout_leaf=16, nr_cpu_ids=4
[Sat Jan  1 23:46:02 GMT 2022] NR_IRQS: 524544, nr_irqs: 456, preallocated irqs: 16
[Sat Jan  1 23:46:02 GMT 2022] random: crng done (trusting CPU's manufacturer)
[Sat Jan  1 23:46:02 GMT 2022] Console: colour VGA+ 80x25
[Sat Jan  1 23:46:02 GMT 2022] printk: console [tty0] enabled
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Core revision 20200925
```

```
[Sat Jan  1 23:46:02 GMT 2022] APIC: Switch to symmetric I/O mode setup
[Sat Jan  1 23:46:02 GMT 2022] x2apic enabled
[Sat Jan  1 23:46:02 GMT 2022] Switched APIC routing to physical x2apic.
[Sat Jan  1 23:46:02 GMT 2022] ..TIMER: vector=0x30 apic1=0 pin1=2 apic2=-1 pin2=-1
[Sat Jan  1 23:46:02 GMT 2022] clocksource: tsc-early: mask: 0xffffffffffffffff max_cycles: 0x396d4bf570c,
max_idle_ns: 881590425443 ns
[Sat Jan  1 23:46:02 GMT 2022] Calibrating delay loop (skipped) preset value.. 3983.99 BogoMIPS (lpj=7967988)
[Sat Jan  1 23:46:02 GMT 2022] pid_max: default: 32768 minimum: 301
[Sat Jan  1 23:46:02 GMT 2022] LSM: Security Framework initializing
[Sat Jan  1 23:46:02 GMT 2022] Yama: disabled by default; enable with sysctl kernel.yama.*
[Sat Jan  1 23:46:02 GMT 2022] AppArmor: AppArmor initialized
[Sat Jan  1 23:46:02 GMT 2022] TOMOYO Linux initialized
[Sat Jan  1 23:46:02 GMT 2022] Mount-cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] Mountpoint-cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] Last level iTLB entries: 4KB 64, 2MB 8, 4MB 8
[Sat Jan  1 23:46:02 GMT 2022] Last level dTLB entries: 4KB 64, 2MB 0, 4MB 0, 1GB 4
[Sat Jan  1 23:46:02 GMT 2022] Spectre V1 : Mitigation: usercopy/swapgs barriers and __user pointer sanitization
[Sat Jan  1 23:46:02 GMT 2022] Spectre V2 : Mitigation: Full generic retpoline
[Sat Jan  1 23:46:02 GMT 2022] Spectre V2 : Spectre v2 / SpectreRSB mitigation: Filling RSB on context switch
[Sat Jan  1 23:46:02 GMT 2022] Speculative Store Bypass: Vulnerable
[Sat Jan  1 23:46:02 GMT 2022] SRBDS: Unknown: Dependent on hypervisor status
[Sat Jan  1 23:46:02 GMT 2022] MDS: Mitigation: Clear CPU buffers
[Sat Jan  1 23:46:02 GMT 2022] Freeing SMP alternatives memory: 32K
[Sat Jan  1 23:46:02 GMT 2022] smpboot: CPU0: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz (family: 0x6, model: 0x8e,
stepping: 0xa)
[Sat Jan  1 23:46:02 GMT 2022] Performance Events: unsupported p6 CPU model 142 no PMU driver, software events only.
[Sat Jan  1 23:46:02 GMT 2022] rcu: Hierarchical SRCU implementation.
[Sat Jan  1 23:46:02 GMT 2022] NMI watchdog: Perf NMI watchdog permanently disabled
[Sat Jan  1 23:46:02 GMT 2022] smp: Bringing up secondary CPUs ...
[Sat Jan  1 23:46:02 GMT 2022] x86: Booting SMP configuration:
[Sat Jan  1 23:46:02 GMT 2022] .... node  #0, CPUs:      #1
[Sat Jan  1 23:46:02 GMT 2022] kvm-clock: cpu 1, msr 2a0b7041, secondary cpu clock
[Sat Jan  1 23:46:02 GMT 2022]  #2
[Sat Jan  1 23:46:02 GMT 2022] kvm-clock: cpu 2, msr 2a0b7081, secondary cpu clock
[Sat Jan  1 23:46:02 GMT 2022]  #3
[Sat Jan  1 23:46:02 GMT 2022] kvm-clock: cpu 3, msr 2a0b70c1, secondary cpu clock
[Sat Jan  1 23:46:02 GMT 2022] smp: Brought up 1 node, 4 CPUs
[Sat Jan  1 23:46:02 GMT 2022] smpboot: Max logical packages: 1
[Sat Jan  1 23:46:02 GMT 2022] smpboot: Total of 4 processors activated (15935.97 BogoMIPS)
[Sat Jan  1 23:46:02 GMT 2022] node 0 deferred pages initialised in 4ms
[Sat Jan  1 23:46:02 GMT 2022] devtmpfs: initialized
[Sat Jan  1 23:46:02 GMT 2022] x86/mm: Memory block size: 128MB
[Sat Jan  1 23:46:02 GMT 2022] clocksource: jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns:
7645041785100000 ns
[Sat Jan  1 23:46:02 GMT 2022] futex hash table entries: 1024 (order: 4, 65536 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] pinctrl core: initialized pinctrl subsystem
[Sat Jan  1 23:46:02 GMT 2022] NET: Registered protocol family 16
[Sat Jan  1 23:46:02 GMT 2022] audit: initializing netlink subsys (disabled)
[Sat Jan  1 23:46:02 GMT 2022] audit: type=2000 audit(1641080769.850:1): state=initialized audit_enabled=0 res=1
[Sat Jan  1 23:46:02 GMT 2022] thermal_sys: Registered thermal governor 'fair_share'
[Sat Jan  1 23:46:02 GMT 2022] thermal_sys: Registered thermal governor 'bang_bang'
[Sat Jan  1 23:46:02 GMT 2022] thermal_sys: Registered thermal governor 'step_wise'
[Sat Jan  1 23:46:02 GMT 2022] thermal_sys: Registered thermal governor 'user_space'
[Sat Jan  1 23:46:02 GMT 2022] thermal_sys: Registered thermal governor 'power_allocator'
[Sat Jan  1 23:46:02 GMT 2022] cpuidle: using governor ladder
[Sat Jan  1 23:46:02 GMT 2022] cpuidle: using governor menu
[Sat Jan  1 23:46:02 GMT 2022] ACPI: bus type PCI registered
[Sat Jan  1 23:46:02 GMT 2022] acpiphp: ACPI Hot Plug PCI Controller Driver version: 0.5
[Sat Jan  1 23:46:02 GMT 2022] PCI: Using configuration type 1 for base access
[Sat Jan  1 23:46:02 GMT 2022] Kprobes globally optimized
[Sat Jan  1 23:46:02 GMT 2022] HugeTLB registered 2.00 MiB page size, pre-allocated 0 pages
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Added _OSI(Module Device)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Added _OSI(Processor Device)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Added _OSI(3.0 _SCP Extensions)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Added _OSI(Processor Aggregator Device)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Added _OSI(Linux-Dell-Video)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Added _OSI(Linux-Lenovo-NV-HDMI-Audio)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Added _OSI(Linux-HPI-Hybrid-Graphics)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: 2 ACPI AML tables successfully acquired and loaded
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Interpreter enabled
[Sat Jan  1 23:46:02 GMT 2022] ACPI: (supports S0 S5)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Using IOAPIC for interrupt routing
[Sat Jan  1 23:46:02 GMT 2022] PCI: Using host bridge windows from ACPI; if necessary, use "pci=nocrs" and report a
bug
[Sat Jan  1 23:46:02 GMT 2022] ACPI: Enabled 2 GPEs in block 00 to 07
[Sat Jan  1 23:46:02 GMT 2022] ACPI: PCI Root Bridge [PCI0] (domain 0000 [bus 00-ff])
```

```
[Sat Jan  1 23:46:02 GMT 2022] acpi PNP0A03:00: _OSC: OS supports [ASPM ClockPM Segments MSI HPX-Type3]
[Sat Jan  1 23:46:02 GMT 2022] acpi PNP0A03:00: _OSC: not requesting OS control; OS requires [ExtendedConfig ASPM
ClockPM MSI]
[Sat Jan  1 23:46:02 GMT 2022] acpi PNP0A03:00: fail to add MMCONFIG information, can't access extended PCI
configuration space under this bridge.
[Sat Jan  1 23:46:02 GMT 2022] PCI host bridge to bus 0000:00
[Sat Jan  1 23:46:02 GMT 2022] pci_bus 0000:00: root bus resource [io  0x0000-0x0cf7 window]
[Sat Jan  1 23:46:02 GMT 2022] pci_bus 0000:00: root bus resource [io  0x0d00-0xffff window]
[Sat Jan  1 23:46:02 GMT 2022] pci_bus 0000:00: root bus resource [mem 0x000a0000-0x000bffff window]
[Sat Jan  1 23:46:02 GMT 2022] pci_bus 0000:00: root bus resource [mem 0xe0000000-0xfdffffff window]
[Sat Jan  1 23:46:02 GMT 2022] pci_bus 0000:00: root bus resource [bus 00-ff]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:00.0: [8086:1237] type 00 class 0x060000
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:01.0: [8086:7000] type 00 class 0x060100
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:01.1: [8086:7111] type 00 class 0x01018a
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:01.1: reg 0x20: [io  0xd000-0xd00f]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x10: [io  0x01f0-0x01f7]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x14: [io  0x03f6]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x18: [io  0x0170-0x0177]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x1c: [io  0x0376]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:02.0: [15ad:0405] type 00 class 0x030000
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:02.0: reg 0x10: [io  0xd010-0xd01f]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:02.0: reg 0x14: [mem 0xe0000000-0xe7ffffff pref]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:02.0: reg 0x18: [mem 0xf0000000-0xf01fffff]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:03.0: [8086:100e] type 00 class 0x020000
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:03.0: reg 0x10: [mem 0xf0200000-0xf021ffff]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:03.0: reg 0x18: [io  0xd020-0xd027]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:04.0: [80ee:cafe] type 00 class 0x088000
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:04.0: reg 0x10: [io  0xd040-0xd05f]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:04.0: reg 0x14: [mem 0xf0400000-0xf07fffff]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:04.0: reg 0x18: [mem 0xf0800000-0xf0803fff pref]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:05.0: [8086:2415] type 00 class 0x040100
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:05.0: reg 0x10: [io  0xd100-0xd1ff]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:05.0: reg 0x14: [io  0xd200-0xd23f]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:06.0: [106b:003f] type 00 class 0x0c0310
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:06.0: reg 0x10: [mem 0xf0804000-0xf0804fff]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:07.0: [8086:7113] type 00 class 0x068000
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:07.0: quirk: [io  0x4000-0x403f] claimed by PIIX4 ACPI
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:07.0: quirk: [io  0x4100-0x410f] claimed by PIIX4 SMB
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:0d.0: [8086:2829] type 00 class 0x010601
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:0d.0: reg 0x10: [io  0xd240-0xd247]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:0d.0: reg 0x14: [io  0xd248-0xd24b]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:0d.0: reg 0x18: [io  0xd250-0xd257]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:0d.0: reg 0x1c: [io  0xd258-0xd25b]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:0d.0: reg 0x20: [io  0xd260-0xd26f]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:0d.0: reg 0x24: [mem 0xf0806000-0xf0807fff]
[Sat Jan  1 23:46:02 GMT 2022] ACPI: PCI Interrupt Link [LNKA] (IRQs 5 9 10 *11)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: PCI Interrupt Link [LNKB] (IRQs 5 9 *10 11)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: PCI Interrupt Link [LNKC] (IRQs 5 *9 10 11)
[Sat Jan  1 23:46:02 GMT 2022] ACPI: PCI Interrupt Link [LNKD] (IRQs 5 9 10 *11)
[Sat Jan  1 23:46:02 GMT 2022] iommu: Default domain type: Translated
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:02.0: vgaarb: setting as boot VGA device
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:02.0: vgaarb: VGA device added: decodes=io+mem,owns=io+mem,locks=none
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:02.0: vgaarb: bridge control possible
[Sat Jan  1 23:46:02 GMT 2022] vgaarb: loaded
[Sat Jan  1 23:46:02 GMT 2022] EDAC MC: Ver: 3.0.0
[Sat Jan  1 23:46:02 GMT 2022] NetLabel: Initializing
[Sat Jan  1 23:46:02 GMT 2022] NetLabel:  domain hash size = 128
[Sat Jan  1 23:46:02 GMT 2022] NetLabel:  protocols = UNLABELED CIPSOv4 CALIPSO
[Sat Jan  1 23:46:02 GMT 2022] NetLabel:  unlabeled traffic allowed by default
[Sat Jan  1 23:46:02 GMT 2022] PCI: Using ACPI for IRQ routing
[Sat Jan  1 23:46:02 GMT 2022] PCI: pci_cache_line_size set to 64 bytes
[Sat Jan  1 23:46:02 GMT 2022] e820: reserve RAM buffer [mem 0x0009fc00-0x0009ffff]
[Sat Jan  1 23:46:02 GMT 2022] e820: reserve RAM buffer [mem 0xdfff0000-0xdfffffff]
[Sat Jan  1 23:46:02 GMT 2022] clocksource: Switched to clocksource kvm-clock
[Sat Jan  1 23:46:02 GMT 2022] VFS: Disk quotas dquot_6.6.0
[Sat Jan  1 23:46:02 GMT 2022] VFS: Dquot-cache hash table entries: 512 (order 0, 4096 bytes)
[Sat Jan  1 23:46:02 GMT 2022] AppArmor: AppArmor Filesystem Enabled
[Sat Jan  1 23:46:02 GMT 2022] pnp: PnP ACPI init
[Sat Jan  1 23:46:02 GMT 2022] pnp 00:00: Plug and Play ACPI device, IDs PNP0303 (active)
[Sat Jan  1 23:46:02 GMT 2022] pnp 00:01: Plug and Play ACPI device, IDs PNP0f03 (active)
[Sat Jan  1 23:46:02 GMT 2022] pnp: PnP ACPI: found 2 devices
[Sat Jan  1 23:46:02 GMT 2022] clocksource: acpi_pm: mask: 0xffffff max_cycles: 0xffffff, max_idle_ns: 2085701024 ns
[Sat Jan  1 23:46:02 GMT 2022] NET: Registered protocol family 2
[Sat Jan  1 23:46:02 GMT 2022] IP idents hash table entries: 65536 (order: 7, 524288 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] tcp_listen_portaddr_hash hash table entries: 2048 (order: 3, 32768 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] TCP established hash table entries: 32768 (order: 6, 262144 bytes, linear)
```

```
[Sat Jan  1 23:46:02 GMT 2022] TCP bind hash table entries: 32768 (order: 7, 524288 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] TCP: Hash tables configured (established 32768 bind 32768)
[Sat Jan  1 23:46:02 GMT 2022] UDP hash table entries: 2048 (order: 4, 65536 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] UDP-Lite hash table entries: 2048 (order: 4, 65536 bytes, linear)
[Sat Jan  1 23:46:02 GMT 2022] NET: Registered protocol family 1
[Sat Jan  1 23:46:02 GMT 2022] NET: Registered protocol family 44
[Sat Jan  1 23:46:02 GMT 2022] pci_bus 0000:00: resource 4 [io  0x0000-0x0cf7 window]
[Sat Jan  1 23:46:02 GMT 2022] pci_bus 0000:00: resource 5 [io  0x0d00-0xffff window]
[Sat Jan  1 23:46:02 GMT 2022] pci_bus 0000:00: resource 6 [mem 0x000a0000-0x000bffff window]
[Sat Jan  1 23:46:02 GMT 2022] pci_bus 0000:00: resource 7 [mem 0xe0000000-0xfdffffff window]
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:00.0: Limiting direct PCI/PCI transfers
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:01.0: Activating ISA DMA hang workarounds
[Sat Jan  1 23:46:02 GMT 2022] pci 0000:00:02.0: Video device with shadowed ROM at [mem 0x000c0000-0x000dffff]
[Sat Jan  1 23:46:02 GMT 2022] PCI: CLS 0 bytes, default 64
[Sat Jan  1 23:46:02 GMT 2022] Trying to unpack rootfs image as initramfs...
[Sat Jan  1 23:46:03 GMT 2022] Freeing initrd memory: 41552K
[Sat Jan  1 23:46:03 GMT 2022] PCI-DMA: Using software bounce buffering for IO (SWIOTLB)
[Sat Jan  1 23:46:03 GMT 2022] software IO TLB: mapped [mem 0x00000000d3000000-0x00000000d7000000] (64MB)
[Sat Jan  1 23:46:03 GMT 2022] clocksource: tsc: mask: 0xffffffffffffffff max_cycles: 0x396d4bf570c, max_idle_ns:
881590425443 ns
[Sat Jan  1 23:46:03 GMT 2022] clocksource: Switched to clocksource tsc
[Sat Jan  1 23:46:03 GMT 2022] platform rtc_cmos: registered platform RTC device (no PNP device found)
[Sat Jan  1 23:46:03 GMT 2022] Initialise system trusted keyrings
[Sat Jan  1 23:46:03 GMT 2022] Key type blacklist registered
[Sat Jan  1 23:46:03 GMT 2022] workingset: timestamp_bits=36 max_order=20 bucket_order=0
[Sat Jan  1 23:46:03 GMT 2022] zbud: loaded
[Sat Jan  1 23:46:03 GMT 2022] integrity: Platform Keyring initialized
[Sat Jan  1 23:46:03 GMT 2022] Key type asymmetric registered
[Sat Jan  1 23:46:03 GMT 2022] Asymmetric key parser 'x509' registered
[Sat Jan  1 23:46:03 GMT 2022] Block layer SCSI generic (bsg) driver version 0.4 loaded (major 251)
[Sat Jan  1 23:46:03 GMT 2022] io scheduler mq-deadline registered
[Sat Jan  1 23:46:03 GMT 2022] shpchp: Standard Hot Plug PCI Controller Driver version: 0.4
[Sat Jan  1 23:46:03 GMT 2022] intel_idle: Please enable MWAIT in BIOS SETUP
[Sat Jan  1 23:46:03 GMT 2022] Serial: 8250/16550 driver, 4 ports, IRQ sharing enabled
[Sat Jan  1 23:46:03 GMT 2022] Linux agpgart interface v0.103
[Sat Jan  1 23:46:03 GMT 2022] AMD-Vi: AMD IOMMUv2 functionality not available on this system - This is not a bug.
[Sat Jan  1 23:46:03 GMT 2022] i8042: PNP: PS/2 Controller [PNP0303:PS2K,PNP0f03:PS2M] at 0x60,0x64 irq 1,12
[Sat Jan  1 23:46:03 GMT 2022] serio: i8042 KBD port at 0x60,0x64 irq 1
[Sat Jan  1 23:46:03 GMT 2022] serio: i8042 AUX port at 0x60,0x64 irq 12
[Sat Jan  1 23:46:03 GMT 2022] mousedev: PS/2 mouse device common for all mice
[Sat Jan  1 23:46:03 GMT 2022] input: AT Translated Set 2 keyboard as /devices/platform/i8042/serio0/input/input0
[Sat Jan  1 23:46:03 GMT 2022] rtc_cmos rtc_cmos: registered as rtc0
[Sat Jan  1 23:46:03 GMT 2022] rtc_cmos rtc_cmos: setting system clock to 2022-01-01T23:46:03 UTC (1641080763)
[Sat Jan  1 23:46:03 GMT 2022] rtc_cmos rtc_cmos: alarms up to one day, 114 bytes nvram
[Sat Jan  1 23:46:03 GMT 2022] intel_pstate: CPU model not supported
[Sat Jan  1 23:46:03 GMT 2022] ledtrig-cpu: registered to indicate activity on CPUs
[Sat Jan  1 23:46:03 GMT 2022] NET: Registered protocol family 10
[Sat Jan  1 23:46:03 GMT 2022] Segment Routing with IPv6
[Sat Jan  1 23:46:03 GMT 2022] mip6: Mobile IPv6
[Sat Jan  1 23:46:03 GMT 2022] NET: Registered protocol family 17
[Sat Jan  1 23:46:03 GMT 2022] mpls_gso: MPLS GSO support
[Sat Jan  1 23:46:03 GMT 2022] IPI shorthand broadcast: enabled
[Sat Jan  1 23:46:03 GMT 2022] sched_clock: Marking stable (1233609906, 16004508)->(1275262612, -25648198)
[Sat Jan  1 23:46:03 GMT 2022] registered taskstats version 1
[Sat Jan  1 23:46:03 GMT 2022] Loading compiled-in X.509 certificates
[Sat Jan  1 23:46:03 GMT 2022] Loaded X.509 cert 'Debian Secure Boot CA: 6ccece7e4c6c0d1f6149f3dd27dfcc5cbb419ea1'
[Sat Jan  1 23:46:03 GMT 2022] Loaded X.509 cert 'Debian Secure Boot Signer 2021 - linux:
4b6ef5abca669825178e052c84667ccbc0531f8c'
[Sat Jan  1 23:46:03 GMT 2022] zswap: loaded using pool lzo/zbud
[Sat Jan  1 23:46:03 GMT 2022] Key type ._fscrypt registered
[Sat Jan  1 23:46:03 GMT 2022] Key type .fscrypt registered
[Sat Jan  1 23:46:03 GMT 2022] Key type fscrypt-provisioning registered
[Sat Jan  1 23:46:03 GMT 2022] AppArmor: AppArmor sha1 policy hashing enabled
[Sat Jan  1 23:46:03 GMT 2022] Freeing unused kernel image (initmem) memory: 2408K
[Sat Jan  1 23:46:03 GMT 2022] Write protecting the kernel read-only data: 22528k
[Sat Jan  1 23:46:03 GMT 2022] Freeing unused kernel image (text/rodata gap) memory: 2040K
[Sat Jan  1 23:46:03 GMT 2022] Freeing unused kernel image (rodata/data gap) memory: 628K
[Sat Jan  1 23:46:03 GMT 2022] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[Sat Jan  1 23:46:03 GMT 2022] x86/mm: Checking user space page tables
[Sat Jan  1 23:46:03 GMT 2022] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[Sat Jan  1 23:46:03 GMT 2022] Run /init as init process
[Sat Jan  1 23:46:03 GMT 2022]   with arguments:
[Sat Jan  1 23:46:03 GMT 2022]     /init
[Sat Jan  1 23:46:03 GMT 2022]   with environment:
[Sat Jan  1 23:46:03 GMT 2022]     HOME=/
[Sat Jan  1 23:46:03 GMT 2022]     TERM=linux
```

```
[Sat Jan  1 23:46:03 GMT 2022]     BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64
[Sat Jan  1 23:46:03 GMT 2022]     crashkernel=384M-:128M
[Sat Jan  1 23:46:03 GMT 2022] input: Power Button as /devices/LNXSYSTM:00/LNXPWRBN:00/input/input2
[Sat Jan  1 23:46:03 GMT 2022] piix4_smbus 0000:00:07.0: SMBus Host Controller at 0x4100, revision 0
[Sat Jan  1 23:46:03 GMT 2022] ACPI: Video Device [GFX0] (multi-head: yes  rom: no  post: no)
[Sat Jan  1 23:46:03 GMT 2022] input: Video Bus as
/devices/LNXSYSTM:00/LNXSYBUS:00/PNP0A03:00/LNXVIDEO:00/input/input3
[Sat Jan  1 23:46:03 GMT 2022] battery: ACPI: Battery Slot [BAT0] (battery present)
[Sat Jan  1 23:46:03 GMT 2022] SCSI subsystem initialized
[Sat Jan  1 23:46:03 GMT 2022] e1000: Intel(R) PRO/1000 Network Driver
[Sat Jan  1 23:46:03 GMT 2022] e1000: Copyright (c) 1999-2006 Intel Corporation.
[Sat Jan  1 23:46:03 GMT 2022] ACPI: Power Button [PWRF]
[Sat Jan  1 23:46:03 GMT 2022] input: Sleep Button as /devices/LNXSYSTM:00/LNXSLPBN:00/input/input5
[Sat Jan  1 23:46:03 GMT 2022] ACPI: Sleep Button [SLPF]
[Sat Jan  1 23:46:03 GMT 2022] libata version 3.00 loaded.
[Sat Jan  1 23:46:03 GMT 2022] ACPI: bus type USB registered
[Sat Jan  1 23:46:03 GMT 2022] usbcore: registered new interface driver usbfs
[Sat Jan  1 23:46:03 GMT 2022] usbcore: registered new interface driver hub
[Sat Jan  1 23:46:03 GMT 2022] usbcore: registered new device driver usb
[Sat Jan  1 23:46:03 GMT 2022] ata_piix 0000:00:01.1: version 2.13
[Sat Jan  1 23:46:03 GMT 2022] scsi host0: ata_piix
[Sat Jan  1 23:46:03 GMT 2022] ahci 0000:00:0d.0: version 3.0
[Sat Jan  1 23:46:03 GMT 2022] ahci 0000:00:0d.0: SSS flag set, parallel bus scan disabled
[Sat Jan  1 23:46:03 GMT 2022] ahci 0000:00:0d.0: AHCI 0001.0100 32 slots 1 ports 3 Gbps 0x1 impl SATA mode
[Sat Jan  1 23:46:03 GMT 2022] ahci 0000:00:0d.0: flags: 64bit ncq stag only ccc
[Sat Jan  1 23:46:03 GMT 2022] scsi host2: ahci
[Sat Jan  1 23:46:03 GMT 2022] ata3: SATA max UDMA/133 abar m8192@0xf0806000 port 0xf0806100 irq 21
[Sat Jan  1 23:46:03 GMT 2022] scsi host1: ata_piix
[Sat Jan  1 23:46:03 GMT 2022] ata1: PATA max UDMA/33 cmd 0x1f0 ctl 0x3f6 bmdma 0xd000 irq 14
[Sat Jan  1 23:46:03 GMT 2022] ata2: PATA max UDMA/33 cmd 0x170 ctl 0x376 bmdma 0xd008 irq 15
[Sat Jan  1 23:46:03 GMT 2022] ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
[Sat Jan  1 23:46:03 GMT 2022] ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
[Sat Jan  1 23:46:03 GMT 2022] ehci-pci: EHCI PCI platform driver
[Sat Jan  1 23:46:03 GMT 2022] ohci-pci: OHCI PCI platform driver
[Sat Jan  1 23:46:03 GMT 2022] ohci-pci 0000:00:06.0: OHCI PCI host controller
[Sat Jan  1 23:46:03 GMT 2022] ohci-pci 0000:00:06.0: new USB bus registered, assigned bus number 1
[Sat Jan  1 23:46:03 GMT 2022] ohci-pci 0000:00:06.0: irq 22, io mem 0xf0804000
[Sat Jan  1 23:46:03 GMT 2022] [drm] DMA map mode: Caching DMA mappings.
[Sat Jan  1 23:46:03 GMT 2022] [drm] Capabilities:
[Sat Jan  1 23:46:03 GMT 2022] [drm]   Cursor.
[Sat Jan  1 23:46:03 GMT 2022] [drm]   Cursor bypass 2.
[Sat Jan  1 23:46:03 GMT 2022] [drm]   Alpha cursor.
[Sat Jan  1 23:46:03 GMT 2022] [drm]   3D.
[Sat Jan  1 23:46:03 GMT 2022] [drm]   Extended Fifo.
[Sat Jan  1 23:46:03 GMT 2022] [drm]   Pitchlock.
[Sat Jan  1 23:46:03 GMT 2022] [drm]   Irq mask.
[Sat Jan  1 23:46:03 GMT 2022] [drm]   GMR.
[Sat Jan  1 23:46:03 GMT 2022] [drm]   Traces.
[Sat Jan  1 23:46:03 GMT 2022] [drm]   GMR2.
[Sat Jan  1 23:46:03 GMT 2022] [drm]   Screen Object 2.
[Sat Jan  1 23:46:03 GMT 2022] [drm] Max GMR ids is 8192
[Sat Jan  1 23:46:03 GMT 2022] [drm] Max number of GMR pages is 1048576
[Sat Jan  1 23:46:03 GMT 2022] [drm] Max dedicated hypervisor surface memory is 393216 kiB
[Sat Jan  1 23:46:03 GMT 2022] [drm] Maximum display memory size is 131072 kiB
[Sat Jan  1 23:46:03 GMT 2022] [drm] VRAM at 0xe0000000 size is 131072 kiB
[Sat Jan  1 23:46:03 GMT 2022] [drm] MMIO at 0xf0000000 size is 2048 kiB
[Sat Jan  1 23:46:03 GMT 2022] [TTM] Zone  kernel: Available graphics memory: 1946798 KiB
[Sat Jan  1 23:46:03 GMT 2022] [TTM] Initializing pool allocator
[Sat Jan  1 23:46:03 GMT 2022] [TTM] Initializing DMA pool allocator
[Sat Jan  1 23:46:03 GMT 2022] [drm] Screen Objects Display Unit initialized
[Sat Jan  1 23:46:03 GMT 2022] [drm] width 720
[Sat Jan  1 23:46:03 GMT 2022] [drm] height 400
[Sat Jan  1 23:46:03 GMT 2022] [drm] bpp 32
[Sat Jan  1 23:46:03 GMT 2022] [drm] Fifo max 0x00200000 min 0x00001000 cap 0x00000355
[Sat Jan  1 23:46:03 GMT 2022] [drm] Atomic: yes.
[Sat Jan  1 23:46:03 GMT 2022] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.
[Sat Jan  1 23:46:03 GMT 2022] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.
[Sat Jan  1 23:46:03 GMT 2022] fbcon: svgadrmfb (fb0) is primary device
[Sat Jan  1 23:46:03 GMT 2022] Console: switching to colour frame buffer device 100x37
[Sat Jan  1 23:46:03 GMT 2022] [drm] Initialized vmwgfx 2.18.0 20200114 for 0000:00:02.0 on minor 0
[Sat Jan  1 23:46:03 GMT 2022] usb usb1: New USB device found, idVendor=1d6b, idProduct=0001, bcdDevice= 5.10
[Sat Jan  1 23:46:03 GMT 2022] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[Sat Jan  1 23:46:03 GMT 2022] usb usb1: Product: OHCI PCI host controller
[Sat Jan  1 23:46:03 GMT 2022] usb usb1: Manufacturer: Linux 5.10.0-10-amd64 ohci_hcd
[Sat Jan  1 23:46:03 GMT 2022] usb usb1: SerialNumber: 0000:00:06.0
[Sat Jan  1 23:46:03 GMT 2022] hub 1-0:1.0: USB hub found
```

```
[Sat Jan  1 23:46:03 GMT 2022] hub 1-0:1.0: 12 ports detected
[Sat Jan  1 23:46:03 GMT 2022] ata2.00: ATAPI: VBOX CD-ROM, 1.0, max UDMA/133
[Sat Jan  1 23:46:03 GMT 2022] input: ImExPS/2 Generic Explorer Mouse as /devices/platform/i8042/serio1/input/input4
[Sat Jan  1 23:46:03 GMT 2022] ata3: SATA link up 3.0 Gbps (SStatus 123 SControl 300)
[Sat Jan  1 23:46:03 GMT 2022] ata3.00: ATA-6: VBOX HARDDISK, 1.0, max UDMA/133
[Sat Jan  1 23:46:03 GMT 2022] ata3.00: 209715200 sectors, multi 128: LBA48 NCQ (depth 32)
[Sat Jan  1 23:46:03 GMT 2022] ata3.00: configured for UDMA/133
[Sat Jan  1 23:46:03 GMT 2022] scsi 2:0:0:0: Direct-Access     ATA      VBOX HARDDISK    1.0  PQ: 0 ANSI: 5
[Sat Jan  1 23:46:03 GMT 2022] scsi 1:0:0:0: CD-ROM            VBOX     CD-ROM           1.0  PQ: 0 ANSI: 5
[Sat Jan  1 23:46:03 GMT 2022] sd 2:0:0:0: [sda] 209715200 512-byte logical blocks: (107 GB/100 GiB)
[Sat Jan  1 23:46:03 GMT 2022] sd 2:0:0:0: [sda] Write Protect is off
[Sat Jan  1 23:46:03 GMT 2022] sd 2:0:0:0: [sda] Mode Sense: 00 3a 00 00
[Sat Jan  1 23:46:03 GMT 2022] sd 2:0:0:0: [sda] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
[Sat Jan  1 23:46:03 GMT 2022]  sda: sda1 sda2 < sda5 >
[Sat Jan  1 23:46:03 GMT 2022] e1000 0000:00:03.0 eth0: (PCI:33MHz:32-bit) 08:00:27:26:5a:6b
[Sat Jan  1 23:46:03 GMT 2022] e1000 0000:00:03.0 eth0: Intel(R) PRO/1000 Network Connection
[Sat Jan  1 23:46:03 GMT 2022] e1000 0000:00:03.0 enp0s3: renamed from eth0
[Sat Jan  1 23:46:03 GMT 2022] sd 2:0:0:0: [sda] Attached SCSI disk
[Sat Jan  1 23:46:03 GMT 2022] sr 1:0:0:0: [sr0] scsi3-mmc drive: 32x/32x xa/form2 tray
[Sat Jan  1 23:46:03 GMT 2022] cdrom: Uniform CD-ROM driver Revision: 3.20
[Sat Jan  1 23:46:03 GMT 2022] usb 1-1: new full-speed USB device number 2 using ohci-pci
[Sat Jan  1 23:46:03 GMT 2022] sr 1:0:0:0: Attached scsi CD-ROM sr0
[Sat Jan  1 23:46:04 GMT 2022] usb 1-1: New USB device found, idVendor=80ee, idProduct=0021, bcdDevice= 1.00
[Sat Jan  1 23:46:04 GMT 2022] usb 1-1: New USB device strings: Mfr=1, Product=3, SerialNumber=0
[Sat Jan  1 23:46:04 GMT 2022] usb 1-1: Product: USB Tablet
[Sat Jan  1 23:46:04 GMT 2022] usb 1-1: Manufacturer: VirtualBox
[Sat Jan  1 23:46:04 GMT 2022] hid: raw HID events driver (C) Jiri Kosina
[Sat Jan  1 23:46:04 GMT 2022] usbcore: registered new interface driver usbhid
[Sat Jan  1 23:46:04 GMT 2022] usbhid: USB HID core driver
[Sat Jan  1 23:46:04 GMT 2022] input: VirtualBox USB Tablet as /devices/pci0000:00/0000:00:06.0/usb1/1-1/1-
1:1.0/0003:80EE:0021.0001/input/input6
[Sat Jan  1 23:46:04 GMT 2022] hid-generic 0003:80EE:0021.0001: input,hidraw0: USB HID v1.10 Mouse [VirtualBox USB
Tablet] on usb-0000:00:06.0-1/input0
[Sat Jan  1 23:46:04 GMT 2022] PM: Image not found (code -22)
[Sat Jan  1 23:46:04 GMT 2022] EXT4-fs (sda1): mounted filesystem with ordered data mode. Opts: (null)
[Sat Jan  1 23:46:04 GMT 2022] Not activating Mandatory Access Control as /sbin/tomoyo-init does not exist.
[Sat Jan  1 23:46:04 GMT 2022] systemd[1]: Inserted module 'autofs4'
[Sat Jan  1 23:46:04 GMT 2022] systemd[1]: systemd 247.3-6 running in system mode. (+PAM +AUDIT +SELINUX +IMA
+APPARMOR +SMACK +SYSVINIT +UTMP +LIBCRYPTSETUP +GCRYPT +GNUTLS +ACL +XZ +LZ4 +ZSTD +SECCOMP +BLKID +ELFUTILS +KMOD
+IDN2 -IDN +PCRE2 default-hierarchy=unified)
[Sat Jan  1 23:46:04 GMT 2022] systemd[1]: Detected virtualization oracle.
[Sat Jan  1 23:46:04 GMT 2022] systemd[1]: Detected architecture x86-64.
[Sat Jan  1 23:46:04 GMT 2022] systemd[1]: Set hostname to <coredump>.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: /lib/systemd/system/plymouth-start.service:16: Unit configured to use
KillMode=none. This is unsafe, as it disables systemd's process lifecycle management for the service. Please update
your service to use a safer KillMode=, such as 'mixed' or 'control-group'. Support for KillMode=none is deprecated and
will eventually be removed.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Queued start job for default target Graphical Interface.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Created slice system-getty.slice.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Created slice system-modprobe.slice.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Created slice User and Session Slice.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Started Forward Password Requests to Wall Directory Watch.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Set up automount Arbitrary Executable File Formats File System Automount
Point.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Reached target User and Group Name Lookups.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Reached target Remote File Systems.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Reached target Slices.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Reached target System Time Set.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Reached target System Time Synchronized.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Listening on Syslog Socket.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Listening on fsck to fsckd communication Socket.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Listening on initctl Compatibility Named Pipe.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Listening on Journal Audit Socket.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Listening on Journal Socket (/dev/log).
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Listening on Journal Socket.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Listening on udev Control Socket.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Listening on udev Kernel Socket.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounting Huge Pages File System...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounting POSIX Message Queue File System...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounting Kernel Debug File System...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounting Kernel Trace File System...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Set the console keyboard layout...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Create list of static device nodes for the current kernel...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Load Kernel Module configfs...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Load Kernel Module drm...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Load Kernel Module fuse...
```

```
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Condition check resulted in Set Up Additional Binary Formats being skipped.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Condition check resulted in File System Check on Root Device being skipped.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Journal Service...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Load Kernel Modules...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Remount Root and Kernel File Systems...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Coldplug All udev Devices...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounted Huge Pages File System.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounted POSIX Message Queue File System.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounted Kernel Debug File System.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounted Kernel Trace File System.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Finished Create list of static device nodes for the current kernel.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: modprobe@configfs.service: Succeeded.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Finished Load Kernel Module configfs.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: modprobe@drm.service: Succeeded.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Finished Load Kernel Module drm.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounting Kernel Configuration File System...
[Sat Jan  1 23:46:05 GMT 2022] fuse: init (API version 7.32)
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: modprobe@fuse.service: Succeeded.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Finished Load Kernel Module fuse.
[Sat Jan  1 23:46:05 GMT 2022] EXT4-fs (sda1): re-mounted. Opts: errors=remount-ro
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounting FUSE Control File System...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Finished Remount Root and Kernel File Systems.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Condition check resulted in Rebuild Hardware Database being skipped.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Condition check resulted in Platform Persistent Storage Archival being
skipped.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Load/Save Random Seed...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Create System Users...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounted Kernel Configuration File System.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Mounted FUSE Control File System.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Finished Load Kernel Modules.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Apply Kernel Variables...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Finished Apply Kernel Variables.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Finished Create System Users.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Create Static Device Nodes in /dev...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Finished Load/Save Random Seed.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Condition check resulted in First Boot Complete being skipped.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Finished Create Static Device Nodes in /dev.
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Starting Rule-based Manager for Device Events and Files...
[Sat Jan  1 23:46:05 GMT 2022] systemd[1]: Started Journal Service.
[Sat Jan  1 23:46:05 GMT 2022] systemd-journald[242]: Received client request to flush runtime journal.
[Sat Jan  1 23:46:05 GMT 2022] audit: type=1400 audit(1641080765.624:2): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="nvidia_modprobe" pid=282 comm="apparmor_parser"
[Sat Jan  1 23:46:05 GMT 2022] audit: type=1400 audit(1641080765.624:3): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="nvidia_modprobe//kmod" pid=282 comm="apparmor_parser"
[Sat Jan  1 23:46:05 GMT 2022] audit: type=1400 audit(1641080765.628:4): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="/usr/bin/man" pid=279 comm="apparmor_parser"
[Sat Jan  1 23:46:05 GMT 2022] audit: type=1400 audit(1641080765.628:5): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="man_filter" pid=279 comm="apparmor_parser"
[Sat Jan  1 23:46:05 GMT 2022] audit: type=1400 audit(1641080765.628:6): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="man_groff" pid=279 comm="apparmor_parser"
[Sat Jan  1 23:46:05 GMT 2022] audit: type=1400 audit(1641080765.628:7): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-senddoc" pid=280 comm="apparmor_parser"
[Sat Jan  1 23:46:05 GMT 2022] audit: type=1400 audit(1641080765.636:8): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="lsb_release" pid=283 comm="apparmor_parser"
[Sat Jan  1 23:46:05 GMT 2022] audit: type=1400 audit(1641080765.636:9): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-oopslash" pid=285 comm="apparmor_parser"
[Sat Jan  1 23:46:05 GMT 2022] audit: type=1400 audit(1641080765.636:10): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-xpdfimport" pid=287 comm="apparmor_parser"
[Sat Jan  1 23:46:05 GMT 2022] ACPI: AC Adapter [AC] (on-line)
[Sat Jan  1 23:46:05 GMT 2022] vboxguest: loading out-of-tree module taints kernel.
[Sat Jan  1 23:46:05 GMT 2022] vboxguest: module verification failed: signature and/or required key missing - tainting
kernel
[Sat Jan  1 23:46:05 GMT 2022] sd 2:0:0:0: Attached scsi generic sg0 type 0
[Sat Jan  1 23:46:05 GMT 2022] sr 1:0:0:0: Attached scsi generic sg1 type 5
[Sat Jan  1 23:46:05 GMT 2022] input: PC Speaker as /devices/platform/pcspkr/input/input7
[Sat Jan  1 23:46:05 GMT 2022] vgdrvHeartbeatInit: Setting up heartbeat to trigger every 2000 milliseconds
[Sat Jan  1 23:46:05 GMT 2022] input: Unspecified device as /devices/pci0000:00/0000:00:04.0/input/input8
[Sat Jan  1 23:46:05 GMT 2022] vboxguest: Successfully loaded version 6.1.30 r148432
[Sat Jan  1 23:46:05 GMT 2022] vboxguest: misc device minor 61, IRQ 20, I/O port d040, MMIO at 00000000f0400000 (size
0x400000)
[Sat Jan  1 23:46:05 GMT 2022] vboxguest: Successfully loaded version 6.1.30 r148432 (interface 0x00010004)
[Sat Jan  1 23:46:05 GMT 2022] Adding 998396k swap on /dev/sda5.  Priority:-2 extents:1 across:998396k FS
[Sat Jan  1 23:46:05 GMT 2022] RAPL PMU: API unit is 2^-32 Joules, 0 fixed counters, 10737418240 ms ovfl timer
[Sat Jan  1 23:46:05 GMT 2022] cryptd: max_cpu_qlen set to 1000
[Sat Jan  1 23:46:05 GMT 2022] AVX2 version of gcm_enc/dec engaged.
[Sat Jan  1 23:46:05 GMT 2022] AES CTR mode by8 optimization enabled
```

```
[Sat Jan  1 23:46:05 GMT 2022] intel_pmc_core intel_pmc_core.0:  initialized
[Sat Jan  1 23:46:05 GMT 2022] snd_intel8x0 0000:00:05.0: allow list rate for 1028:0177 is 48000
[Sat Jan  1 23:46:07 GMT 2022] e1000: enp0s3 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[Sat Jan  1 23:46:07 GMT 2022] IPv6: ADDRCONF(NETDEV_CHANGE): enp0s3: link becomes ready
[Sat Jan  1 23:46:10 GMT 2022] vboxvideo: loading version 6.1.30 r148432
[Sat Jan  1 23:46:10 GMT 2022] 23:46:11.223560 main     VBoxService 6.1.30 r148432 (verbosity: 0) linux.amd64 (Nov 22
2021 16:16:32) release log
                               23:46:11.223563 main     Log opened 2022-01-01T23:46:11.223557000Z
[Sat Jan  1 23:46:10 GMT 2022] 23:46:11.223624 main     OS Product: Linux
[Sat Jan  1 23:46:10 GMT 2022] 23:46:11.223644 main     OS Release: 5.10.0-10-amd64
[Sat Jan  1 23:46:10 GMT 2022] 23:46:11.223659 main     OS Version: #1 SMP Debian 5.10.84-1 (2021-12-08)
[Sat Jan  1 23:46:10 GMT 2022] 23:46:11.223674 main     Executable: /opt/VBoxGuestAdditions-6.1.30/sbin/VBoxService
                               23:46:11.223675 main     Process ID: 749
                               23:46:11.223675 main     Package type: LINUX_64BITS_GENERIC
[Sat Jan  1 23:46:10 GMT 2022] 23:46:11.225123 main     6.1.30 r148432 started. Verbose level = 0
[Sat Jan  1 23:46:10 GMT 2022] 23:46:11.225707 main     vbglR3GuestCtrlDetectPeekGetCancelSupport: Supported (#1)
[Sat Jan  1 23:46:13 GMT 2022] rfkill: input handler disabled
[Sat Jan  1 23:46:19 GMT 2022] rfkill: input handler enabled
[Sat Jan  1 23:46:20 GMT 2022] rfkill: input handler disabled
[Sun Jan  2 00:19:32 GMT 2022] BUG: kernel NULL pointer dereference, address: 0000000000000000
[Sun Jan  2 00:19:32 GMT 2022] #PF: supervisor write access in kernel mode
[Sun Jan  2 00:19:32 GMT 2022] #PF: error_code(0x0002) - not-present page
[Sun Jan  2 00:19:32 GMT 2022] PGD 0 P4D 0
[Sun Jan  2 00:19:32 GMT 2022] Oops: 0002 [#1] SMP PTI
[Sun Jan  2 00:19:32 GMT 2022] CPU: 2 PID: 3926 Comm: insmod Kdump: loaded Tainted: G           OE     5.10.0-10-amd64
#1 Debian 5.10.84-1
[Sun Jan  2 00:19:32 GMT 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sun Jan  2 00:19:32 GMT 2022] RIP: 0010:init_module+0x5/0x20 [mod_a]
[Sun Jan  2 00:19:32 GMT 2022] Code: Unable to access opcode bytes at RIP 0xffffffffc063bfdb.
[Sun Jan  2 00:19:32 GMT 2022] RSP: 0018:ffff9a2744617df8 EFLAGS: 00010246
[Sun Jan  2 00:19:32 GMT 2022] RAX: 0000000000000000 RBX: 0000000000000000 RCX: 0000000000000000
[Sun Jan  2 00:19:32 GMT 2022] RDX: 0000000000000cc0 RSI: ffffffffa05355c3 RDI: ffffffffc063c000
[Sun Jan  2 00:19:32 GMT 2022] RBP: ffffffffc063c000 R08: 0000000000000010 R09: ffff8a5b7bbf4110
[Sun Jan  2 00:19:32 GMT 2022] R10: ffff8a5b58731280 R11: 0000000000000000 R12: ffff8a5b7bbf4110
[Sun Jan  2 00:19:32 GMT 2022] R13: ffff9a2744617e90 R14: 0000000000000003 R15: 0000000000000000
[Sun Jan  2 00:19:32 GMT 2022] FS:  00007f9477b73540(0000) GS:ffff8a5c5bd00000(0000) knlGS:0000000000000000
[Sun Jan  2 00:19:32 GMT 2022] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[Sun Jan  2 00:19:32 GMT 2022] CR2: ffffffffc063bfdb CR3: 0000000014fb0001 CR4: 00000000000706e0
[Sun Jan  2 00:19:32 GMT 2022] Call Trace:
[Sun Jan  2 00:19:32 GMT 2022]  do_one_initcall+0x44/0x1d0
[Sun Jan  2 00:19:32 GMT 2022]  ? do_init_module+0x23/0x260
[Sun Jan  2 00:19:32 GMT 2022]  ? kmem_cache_alloc_trace+0xf5/0x200
[Sun Jan  2 00:19:32 GMT 2022]  do_init_module+0x5c/0x260
[Sun Jan  2 00:19:32 GMT 2022]  __do_sys_finit_module+0xb1/0x110
[Sun Jan  2 00:19:32 GMT 2022]  do_syscall_64+0x33/0x80
[Sun Jan  2 00:19:32 GMT 2022]  entry_SYSCALL_64_after_hwframe+0x44/0xa9
[Sun Jan  2 00:19:32 GMT 2022] RIP: 0033:0x7f9477c949b9
[Sun Jan  2 00:19:32 GMT 2022] Code: 00 c3 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 44 00 00 48 89 f8 48 89 f7 48 89 d6 48
89 ca 4d 89 c2 4d 89 c8 4c 8b 4c 24 08 0f 05 <48> 3d 01 f0 ff ff 73 01 c3 48 8b 0d a7 54 0c 00 f7 d8 64 89 01 48
[Sun Jan  2 00:19:32 GMT 2022] RSP: 002b:00007fffd058eb98 EFLAGS: 00000246 ORIG_RAX: 0000000000000139
[Sun Jan  2 00:19:32 GMT 2022] RAX: ffffffffffffffda RBX: 000056091e443790 RCX: 00007f9477c949b9
[Sun Jan  2 00:19:32 GMT 2022] RDX: 0000000000000000 RSI: 000056091e072260 RDI: 0000000000000003
[Sun Jan  2 00:19:32 GMT 2022] RBP: 0000000000000000 R08: 0000000000000000 R09: 00007f9477d5e640
[Sun Jan  2 00:19:32 GMT 2022] R10: 0000000000000003 R11: 0000000000000246 R12: 000056091e072260
[Sun Jan  2 00:19:32 GMT 2022] R13: 0000000000000000 R14: 000056091e443760 R15: 0000000000000000
[Sun Jan  2 00:19:32 GMT 2022] Modules linked in: mod_a(OE+) vboxvideo(OE) intel_rapl_msr intel_rapl_common
intel_pmc_core_pltdrv intel_pmc_core ghash_clmulni_intel rfkill aesni_intel libaes crypto_simd cryptd glue_helper rapl
snd_intel8x0 snd_ac97_codec ac97_bus snd_pcm joydev snd_timer serio_raw snd pcspkr sg vboxguest(OE) soundcore evdev ac
msr fuse configfs ip_tables x_tables autofs4 ext4 crc16 mbcache jbd2 crc32c_generic hid_generic usbhid hid sr_mod
sd_mod cdrom t10_pi crc_t10dif crct10dif_generic ata_generic vmwgfx ttm drm_kms_helper ohci_pci ehci_pci ohci_hcd cec
ehci_hcd crct10dif_pclmul crct10dif_common ahci ata_piix crc32_pclmul libahci libata drm usbcore crc32c_intel e1000
scsi_mod psmouse video battery usb_common i2c_piix4 button
[Sun Jan  2 00:19:32 GMT 2022] CR2: 0000000000000000
```

3. We also get an exception stack trace from the tool where we have more information without and with source code references:

```
crash> bt
PID: 3926   TASK: ffff8a5b4430af80  CPU: 2   COMMAND: "insmod"
 #0 [ffff9a2744617bc8] machine_kexec at ffffffffa046436b
 #1 [ffff9a2744617c20] __crash_kexec at ffffffffa053aaad
 #2 [ffff9a2744617ce8] crash_kexec at ffffffffa053bbe5
 #3 [ffff9a2744617cf8] oops_end at ffffffffa042da9b
 #4 [ffff9a2744617d18] exc_page_fault at ffffffffa0cb6c98
 #5 [ffff9a2744617d40] asm_exc_page_fault at ffffffffa0e00ade
    [exception RIP: init_module+5]
    RIP: ffffffffc063c005  RSP: ffff9a2744617df8  RFLAGS: 00010246
    RAX: 0000000000000000  RBX: 0000000000000000  RCX: 0000000000000000
    RDX: 00000000000000cc0  RSI: ffffffffa05355c3  RDI: ffffffffc063c000
    RBP: ffffffffc063c000   R8: 0000000000000010   R9: ffff8a5b7bbf4110
    R10: ffff8a5b58731280  R11: 0000000000000000  R12: ffff8a5b7bbf4110
    R13: ffff9a2744617e90  R14: 0000000000000003  R15: 0000000000000000
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
 #6 [ffff9a2744617df8] do_one_initcall at ffffffffa0403874
 #7 [ffff9a2744617e60] do_init_module at ffffffffa05355fc
 #8 [ffff9a2744617e80] __do_sys_finit_module at ffffffffa0538281
 #9 [ffff9a2744617f40] do_syscall_64 at ffffffffa0cb3883
#10 [ffff9a2744617f50] entry_SYSCALL_64_after_hwframe at ffffffffa0e0008c
    RIP: 00007f9477c949b9  RSP: 00007fffd058eb98  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 000056091e443790  RCX: 00007f9477c949b9
    RDX: 0000000000000000  RSI: 000056091e072260  RDI: 0000000000000003
    RBP: 0000000000000000   R8: 0000000000000000   R9: 00007f9477d5e640
    R10: 0000000000000003  R11: 0000000000000246  R12: 000056091e072260
    R13: 0000000000000000  R14: 000056091e443760  R15: 0000000000000000
    ORIG_RAX: 0000000000000139  CS: 0033  SS: 002b

crash> bt -l
PID: 3926   TASK: ffff8a5b4430af80  CPU: 2   COMMAND: "insmod"
 #0 [ffff9a2744617bc8] machine_kexec at ffffffffa046436b
    debian/build/build_amd64_none_amd64/include/linux/ftrace.h: 788
 #1 [ffff9a2744617c20] __crash_kexec at ffffffffa053aaad
    debian/build/build_amd64_none_amd64/kernel/kexec_core.c: 963
 #2 [ffff9a2744617ce8] crash_kexec at ffffffffa053bbe5
    debian/build/build_amd64_none_amd64/arch/x86/include/asm/atomic.h: 41
 #3 [ffff9a2744617cf8] oops_end at ffffffffa042da9b
    debian/build/build_amd64_none_amd64/arch/x86/kernel/dumpstack.c: 359
 #4 [ffff9a2744617d18] exc_page_fault at ffffffffa0cb6c98
    debian/build/build_amd64_none_amd64/arch/x86/include/asm/paravirt.h: 658
 #5 [ffff9a2744617d40] asm_exc_page_fault at ffffffffa0e00ade
    /build/linux-3cXDux/linux-5.10.84/arch/x86/include/asm/idtentry.h: 571
    [exception RIP: init_module+5]
    RIP: ffffffffc063c005  RSP: ffff9a2744617df8  RFLAGS: 00010246
    RAX: 0000000000000000  RBX: 0000000000000000  RCX: 0000000000000000
    RDX: 00000000000000cc0  RSI: ffffffffa05355c3  RDI: ffffffffc063c000
    RBP: ffffffffc063c000   R8: 0000000000000010   R9: ffff8a5b7bbf4110
    R10: ffff8a5b58731280  R11: 0000000000000000  R12: ffff8a5b7bbf4110
    R13: ffff9a2744617e90  R14: 0000000000000003  R15: 0000000000000000
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
 #6 [ffff9a2744617df8] do_one_initcall at ffffffffa0403874
    debian/build/build_amd64_none_amd64/init/main.c: 1214
 #7 [ffff9a2744617e60] do_init_module at ffffffffa05355fc
    debian/build/build_amd64_none_amd64/kernel/module.c: 3725
 #8 [ffff9a2744617e80] __do_sys_finit_module at ffffffffa0538281
```

```
    debian/build/build_amd64_none_amd64/kernel/module.c: 4200
 #9 [ffff9a2744617f40] do_syscall_64 at ffffffffa0cb3883
    debian/build/build_amd64_none_amd64/arch/x86/entry/common.c: 46
#10 [ffff9a2744617f50] entry_SYSCALL_64_after_hwframe at ffffffffa0e0008c
    /build/linux-3cXDux/linux-5.10.84/arch/x86/entry/entry_64.S: 127
    RIP: 00007f9477c949b9  RSP: 00007fffd058eb98  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 000056091e443790  RCX: 00007f9477c949b9
    RDX: 0000000000000000  RSI: 000056091e072260  RDI: 0000000000000003
    RBP: 0000000000000000   R8: 0000000000000000   R9: 00007f9477d5e640
    R10: 0000000000000003  R11: 0000000000000246  R12: 000056091e072260
    R13: 0000000000000000  R14: 000056091e443760  R15: 0000000000000000
    ORIG_RAX: 0000000000000139  CS: 0033  SS: 002b
```

4. If we disassemble the problem RIP address or function, we confirm NULL pointer dereference (we also see that the code was optimized as we don't see our module function calls from *init_module* that led to the exception):

```
crash> dis ffffffffc063c005
0xffffffffc063c005 <init_module+5>:     movl   $0x1,0x0

crash> dis init_module+5
0xffffffffc063c005 <init_module+5>:     movl   $0x1,0x0

crash> dis init_module
0xffffffffc063c000 <init_module>:       nopl   0x0(%rax,%rax,1) [FTRACE NOP]
0xffffffffc063c005 <init_module+5>:     movl   $0x1,0x0
0xffffffffc063c010 <init_module+16>:    xor    %eax,%eax
0xffffffffc063c012 <init_module+18>:    ret
0xffffffffc063c013 <init_module+19>:    data16 nopw %cs:0x0(%rax,%rax,1)
0xffffffffc063c01e <init_module+30>:    xchg   %ax,%ax
```

5. Now we dump raws stack region around the stack pointer to see exception processing execution residue:

```
crash> rd -SS -64 ffff9a2744617d08 50
ffff9a2744617d08:  ffff9a2744617d48 0000000000000000
ffff9a2744617d18:  exc_page_fault+120 0000000000000000
ffff9a2744617d28:  0000000000000000 0000000000000000
ffff9a2744617d38:  0000000000000000 asm_exc_page_fault+30
ffff9a2744617d48:  0000000000000000 0000000000000003
ffff9a2744617d58:  ffff9a2744617e90 [ffff8a5b7bbf4110:kmalloc-16]
ffff9a2744617d68:  init_module      0000000000000000
ffff9a2744617d78:  0000000000000000 [ffff8a5b58731280:kmalloc-64]
ffff9a2744617d88:  [ffff8a5b7bbf4110:kmalloc-16] 0000000000000010
ffff9a2744617d98:  0000000000000000 0000000000000000
ffff9a2744617da8:  0000000000000cc0 do_init_module+35
ffff9a2744617db8:  init_module      ffffffffffffffff
ffff9a2744617dc8:  init_module+5    0000000000000010
ffff9a2744617dd8:  0000000000010246 ffff9a2744617df8
ffff9a2744617de8:  0000000000000018 init_module
ffff9a2744617df8:  do_one_initcall+68 do_init_module+35
ffff9a2744617e08:  0000000000000cc0 kmem_cache_alloc_trace+245
ffff9a2744617e18:  [ffff8a5b7bbf4110:kmalloc-16] 0000000000000000
ffff9a2744617e28:  3a0c1c3bc650cd00 0000000000000000
ffff9a2744617e38:  __this_module    3a0c1c3bc650cd00
ffff9a2744617e48:  0000000000000000 __this_module
ffff9a2744617e58:  [ffff8a5b7bbf4110:kmalloc-16] do_init_module+92
ffff9a2744617e68:  000056091e072260 0000000000000000
ffff9a2744617e78:  ffff9a2744617e90 __do_sys_finit_module+177
ffff9a2744617e88:  ffff9a27446a1000 ffff9a27446a111f
```

6. Search for the address ffffffffc063c005 in kernel space:

```
crash> search ffffffffc063c005
ffff8a5b6a0721e0: ffffffffc063c005
ffff8a5b79412ca8: ffffffffc063c005
ffff8a5b79412dc8: ffffffffc063c005
ffff9a2744617ca8: ffffffffc063c005
ffff9a2744617dc8: ffffffffc063c005
ffffffffa22721e0: ffffffffc063c005
```

Note: We see that the address was also found in the raw stack region we inspected in step #5.

7. We can also search for strings; for example, search for “bad” as a value and string:

```
crash> search "bad"
ffff8a5b49b60ce8: bad
ffff8a5b764aa148: bad
ffff8a5b764aa1f0: bad
ffff8a5b76550a28: bad
ffff8a5b76550ad0: bad
ffff8a5b947958c8: bad
ffff8a5b94795970: bad
ffff8a5c406a4500: bad
ffff8a5c40909de8: bad
ffff8a5c40a42eb0: bad
ffff8a5c5c199420: bad
ffff8a5c5c203d60: bad
ffff8a5c5cca8920: bad
ffff8a5c5d023f60: bad
ffff8a5c5d0ec960: bad
ffff8a5c5d1952e0: bad
ffff8a5c5fbc94a0: bad
ffffe9e1c0399420: bad
ffffe9e1c0403d60: bad
ffffe9e1c0ea8920: bad
ffffe9e1c1223f60: bad
ffffe9e1c12ec960: bad
ffffe9e1c13952e0: bad
ffffe9e1c45c94a0: bad
ffffffff81d60ce8: bad

crash> rd ffff8a5b76550a28
ffff8a5b76550a28:  0000000000000bad                    ........

crash> search -c "bad"
ffff8a5b43f5563b: bados.....................................J\......
ffff8a5b568726a3: bad-1.0:amd64.list.........................libproxy1-p
ffff8a5b56872d25: bad1.0-0:amd64.list.......................speech-dispat
ffff8a5b56872e65: bad:amd64.list............................debian-archiv
ffff8a5b56aa657b: bados.....................................J\......
ffff8a5b56b1cb7e: bad.......................................J\........
ffff8a5b56b277be: bad.......................................J\........
ffff8a5b56b45abe: bad.......................................J\........
ffff8a5b690012a5: bad_vsyscall..............................
ffff8a5b69003eac: bad_irq.....irq_vectors...................
ffff8a5b69082137: bad_sector................................P....
ffff8a5b692b7476: bad stack (exploit attempt?)......seccomp tried to chang
ffff8a5b692b88c0: bad-spec.event=0x00,umask=0x81.topdown-retiring.event=0x
ffff8a5b692bb1e9: bad frame in %s frame:%p ip:%lx sp:%lx orax:%lx..0..c in
ffff8a5b692c2ac4: bad microcode data file size.........3microcode: Error:
```

```
ffff8a5b692c594e: bad cpu %d...6.... node %*s#%d, CPUs:  ..c%*s..c%*s#%d.s
ffff8a5b692c68e3: bad signature [%c%c%c%c]!.....3MPTABLE: bad table versio
ffff8a5b692c690b: bad table version (%d)!!......3MPTABLE: null local APIC
ffff8a5b692caae4: bad lookup idx: idx=%u num=%u ip=%pB.........4WARNING: W
ffff8a5b692cab24: bad lookup value: idx=%u num=%u start=%u stop=%u ip=%pB.
ffff8a5b692cc0d0: bad address %p..ioremap on RAM at %pa - %pa...4pmd %p !=
ffff8a5b692d38ba: bad: scheduling from the idle thread!..........4Unable t
ffff8a5b692dde32: bad vermagic.intree.staging.license.GPL v2.GPL and addit
ffff8a5b692df25b: bad section index %d.........kexec_file: kernel loader d
ffff8a5b692df56f: bad section index %d.....Loading segment %d: buf=0x%p bu
ffff8a5b692e4ee1: bad taint, not creating trace events....4Failed to enabl
ffff8a5b692edcae: bad rc=0x%x from %ps()..handle tail call...4ref_ctr goin
ffff8a5b692ef6dd: bad_val.,size=%luk.,nr_inodes=%lu.,mode=%03ho.,uid=%u.,g
ffff8a5b692f21dd: bad pte.mm/memory.c.include/linux/swapops.h.include/asm-
ffff8a5b692f2962: bad pgd %p(%016lx)...3%s:%d: bad pud %p(%016lx)...3%s:%d
ffff8a5b692f297f: bad pud %p(%016lx)...3%s:%d: bad pmd %p(%016lx)..&anon_v
ffff8a5b692f299c: bad pmd %p(%016lx)..&anon_vma->rwsem.mm/rmap.c.anon_vma.
ffff8a5b692f2b42: bad address (%p).......3Trying to vfree() nonexistent vm
```

Exercise K3

- **Goal:** Learn how to recognize problems with kernel threads, identify their owner module, and follow call chains
- **Patterns:** Origin Module; NULL Pointer (Code); Hidden Call
- \ALCDA-Dumps\Exercise-K3-x64-GDB.pdf

Exercise K3 (x64, GDB)

Goal: Learn how to recognize problems with kernel threads, identify their owner module, and follow call chains.

Patterns: Origin Module; NULL Pointer (Code); Hidden Call.

1. Load a core dump *dump.202206251922* from the x64/K3 directory and the matching *vmlinux-5.10.0-10-amd64* file from the x64/KSym directory:

```
~/ALCDA2/x64/K3$ crash dump.202206251922 ../KSym/vmlinux-5.10.0-10-amd64

crash 8.0.0++
Copyright (C) 2002-2021  Red Hat, Inc.
Copyright (C) 2004, 2005, 2006, 2010  IBM Corporation
Copyright (C) 1999-2006  Hewlett-Packard Co
Copyright (C) 2005, 2006, 2011, 2012  Fujitsu Limited
Copyright (C) 2006, 2007  VA Linux Systems Japan K.K.
Copyright (C) 2005, 2011, 2020-2021  NEC Corporation
Copyright (C) 1999, 2002, 2007  Silicon Graphics, Inc.
Copyright (C) 1999, 2000, 2001, 2002  Mission Critical Linux, Inc.
Copyright (C) 2015, 2021  VMware, Inc.
This program is free software, covered by the GNU General Public License,
and you are welcome to change it and/or distribute copies of it under
certain conditions.  Enter "help copying" to see the conditions.
This program has absolutely no warranty.  Enter "help warranty" for details.

GNU gdb (GDB) 10.2
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-pc-linux-gnu".
Type "show configuration" for configuration details.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...

      KERNEL: ../KSym/vmlinux-5.10.0-10-amd64  [TAINTED]
    DUMPFILE: dump.202206251922  [PARTIAL DUMP]
        CPUS: 4
        DATE: Sat Jun 25 19:22:31 BST 2022
      UPTIME: 00:02:38
LOAD AVERAGE: 0.52, 0.29, 0.11
       TASKS: 465
    NODENAME: coredump
     RELEASE: 5.10.0-10-amd64
     VERSION: #1 SMP Debian 5.10.84-1 (2021-12-08)
     MACHINE: x86_64  (1992 Mhz)
      MEMORY: 4 GB
       PANIC: "Oops: 0010 [#1] SMP PTI" (check log for details)
         PID: 2189
     COMMAND: "mod_b thread"
        TASK: ffff8facda610000  [THREAD_INFO: ffff8facda610000]
         CPU: 1
```

```
      STATE: TASK_RUNNING (PANIC)

crash>
```

2. We follow the suggestion to check the log for details, and at the end, we find the bug description, crash RIP, the stack pointer, and the stack trace:

```
crash> log -T
[Sat Jun 25 19:19:53 BST 2022] Linux version 5.10.0-10-amd64 (debian-kernel@lists.debian.org) (gcc-10 (Debian 10.2.1-
6) 10.2.1 20210110, GNU ld (GNU Binutils for Debian) 2.35.2) #1 SMP Debian 5.10.84-1 (2021-12-08)
[Sat Jun 25 19:19:53 BST 2022] Command line: BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64 root=UUID=9cc5ee1e-5533-4a0b-
a88f-903bf52d812d ro quiet crashkernel=384M-:128M
[Sat Jun 25 19:19:53 BST 2022] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[Sat Jun 25 19:19:53 BST 2022] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[Sat Jun 25 19:19:53 BST 2022] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[Sat Jun 25 19:19:53 BST 2022] x86/fpu: xstate_offset[2]:  576, xstate_sizes[2]:  256
[Sat Jun 25 19:19:53 BST 2022] x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using 'standard'
format.
[Sat Jun 25 19:19:53 BST 2022] BIOS-provided physical RAM map:
[Sat Jun 25 19:19:53 BST 2022] BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
[Sat Jun 25 19:19:53 BST 2022] BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved
[Sat Jun 25 19:19:53 BST 2022] BIOS-e820: [mem 0x00000000000f0000-0x00000000000fffff] reserved
[Sat Jun 25 19:19:53 BST 2022] BIOS-e820: [mem 0x0000000000100000-0x00000000dffeffff] usable
[Sat Jun 25 19:19:53 BST 2022] BIOS-e820: [mem 0x00000000dfff0000-0x00000000dfffffff] ACPI data
[Sat Jun 25 19:19:53 BST 2022] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec00fff] reserved
[Sat Jun 25 19:19:53 BST 2022] BIOS-e820: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
[Sat Jun 25 19:19:53 BST 2022] BIOS-e820: [mem 0x00000000fffc0000-0x00000000ffffffff] reserved
[Sat Jun 25 19:19:53 BST 2022] BIOS-e820: [mem 0x0000000100000000-0x000000011fffffff] usable
[Sat Jun 25 19:19:53 BST 2022] NX (Execute Disable) protection: active
[Sat Jun 25 19:19:53 BST 2022] SMBIOS 2.5 present.
[Sat Jun 25 19:19:53 BST 2022] DMI: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:19:53 BST 2022] Hypervisor detected: KVM
[Sat Jun 25 19:19:53 BST 2022] kvm-clock: Using msrs 4b564d01 and 4b564d00
[Sat Jun 25 19:19:53 BST 2022] kvm-clock: cpu 0, msr 648b7001, primary cpu clock
[Sat Jun 25 19:19:53 BST 2022] kvm-clock: using sched offset of 5788114847 cycles
[Sat Jun 25 19:19:53 BST 2022] clocksource: kvm-clock: mask: 0xffffffffffffffff max_cycles: 0x1cd42e4dffb,
max_idle_ns: 881590591483 ns
[Sat Jun 25 19:19:53 BST 2022] tsc: Detected 1992.006 MHz processor
[Sat Jun 25 19:19:53 BST 2022] e820: update [mem 0x00000000-0x00000fff] usable ==> reserved
[Sat Jun 25 19:19:53 BST 2022] e820: remove [mem 0x000a0000-0x000fffff] usable
[Sat Jun 25 19:19:53 BST 2022] last_pfn = 0x120000 max_arch_pfn = 0x400000000
[Sat Jun 25 19:19:53 BST 2022] MTRR default type: uncachable
[Sat Jun 25 19:19:53 BST 2022] MTRR variable ranges disabled:
[Sat Jun 25 19:19:53 BST 2022] Disabled
[Sat Jun 25 19:19:53 BST 2022] x86/PAT: MTRRs disabled, skipping PAT initialization too.
[Sat Jun 25 19:19:53 BST 2022] CPU MTRRs all blank - virtualized system.
[Sat Jun 25 19:19:53 BST 2022] x86/PAT: Configuration [0-7]: WB  WT  UC- UC  WB  WT  UC- UC
[Sat Jun 25 19:19:53 BST 2022] last_pfn = 0xdfff0 max_arch_pfn = 0x400000000
[Sat Jun 25 19:19:53 BST 2022] found SMP MP-table at [mem 0x0009fff0-0x0009ffff]
[Sat Jun 25 19:19:53 BST 2022] kexec: Reserving the low 1M of memory for crashkernel
[Sat Jun 25 19:19:53 BST 2022] RAMDISK: [mem 0x32ec7000-0x3575afff]
[Sat Jun 25 19:19:53 BST 2022] ACPI: Early table checksum verification disabled
[Sat Jun 25 19:19:53 BST 2022] ACPI: RSDP 0x00000000000E0000 000024 (v02 VBOX  )
[Sat Jun 25 19:19:53 BST 2022] ACPI: XSDT 0x00000000DFFF0030 00003C (v01 VBOX   VBOXXSDT 00000001 ASL  00000061)
[Sat Jun 25 19:19:53 BST 2022] ACPI: FACP 0x00000000DFFF00F0 0000F4 (v04 VBOX   VBOXFACP 00000001 ASL  00000061)
[Sat Jun 25 19:19:53 BST 2022] ACPI: DSDT 0x00000000DFFF0480 002325 (v02 VBOX   VBOXBIOS 00000002 INTL 20190509)
[Sat Jun 25 19:19:53 BST 2022] ACPI: FACS 0x00000000DFFF0200 000040
[Sat Jun 25 19:19:53 BST 2022] ACPI: FACS 0x00000000DFFF0200 000040
[Sat Jun 25 19:19:53 BST 2022] ACPI: APIC 0x00000000DFFF0240 00006C (v02 VBOX   VBOXAPIC 00000001 ASL  00000061)
[Sat Jun 25 19:19:53 BST 2022] ACPI: SSDT 0x00000000DFFF02B0 0001CC (v01 VBOX   VBOXCPUT 00000002 INTL 20190509)
[Sat Jun 25 19:19:53 BST 2022] ACPI: Reserving FACP table memory at [mem 0xdfff00f0-0xdfff01e3]
[Sat Jun 25 19:19:53 BST 2022] ACPI: Reserving DSDT table memory at [mem 0xdfff0480-0xdfff27a4]
[Sat Jun 25 19:19:53 BST 2022] ACPI: Reserving FACS table memory at [mem 0xdfff0200-0xdfff023f]
[Sat Jun 25 19:19:53 BST 2022] ACPI: Reserving FACS table memory at [mem 0xdfff0200-0xdfff023f]
[Sat Jun 25 19:19:53 BST 2022] ACPI: Reserving APIC table memory at [mem 0xdfff0240-0xdfff02ab]
[Sat Jun 25 19:19:53 BST 2022] ACPI: Reserving SSDT table memory at [mem 0xdfff02b0-0xdfff047b]
[Sat Jun 25 19:19:53 BST 2022] ACPI: Local APIC address 0xfee00000
[Sat Jun 25 19:19:53 BST 2022] No NUMA configuration found
[Sat Jun 25 19:19:53 BST 2022] Faking a node at [mem 0x0000000000000000-0x000000011fffffff]
[Sat Jun 25 19:19:53 BST 2022] NODE_DATA(0) allocated [mem 0x11ffd2000-0x11fffbfff]
[Sat Jun 25 19:19:53 BST 2022] Reserving 128MB of memory at 3440MB for crashkernel (System RAM: 4095MB)
[Sat Jun 25 19:19:53 BST 2022] Zone ranges:
```

```
[Sat Jun 25 19:19:53 BST 2022]   DMA      [mem 0x0000000000001000-0x0000000000ffffff]
[Sat Jun 25 19:19:53 BST 2022]   DMA32    [mem 0x0000000001000000-0x00000000ffffffff]
[Sat Jun 25 19:19:53 BST 2022]   Normal   [mem 0x0000000100000000-0x000000011fffffff]
[Sat Jun 25 19:19:53 BST 2022]   Device   empty
[Sat Jun 25 19:19:53 BST 2022] Movable zone start for each node
[Sat Jun 25 19:19:53 BST 2022] Early memory node ranges
[Sat Jun 25 19:19:53 BST 2022]   node   0: [mem 0x0000000000001000-0x000000000009efff]
[Sat Jun 25 19:19:53 BST 2022]   node   0: [mem 0x0000000000100000-0x00000000dffeffff]
[Sat Jun 25 19:19:53 BST 2022]   node   0: [mem 0x0000000100000000-0x000000011fffffff]
[Sat Jun 25 19:19:53 BST 2022] Initmem setup node 0 [mem 0x0000000000001000-0x000000011fffffff]
[Sat Jun 25 19:19:53 BST 2022] On node 0 totalpages: 1048462
[Sat Jun 25 19:19:53 BST 2022]   DMA zone: 64 pages used for memmap
[Sat Jun 25 19:19:53 BST 2022]   DMA zone: 158 pages reserved
[Sat Jun 25 19:19:53 BST 2022]   DMA zone: 3998 pages, LIFO batch:0
[Sat Jun 25 19:19:53 BST 2022]   DMA32 zone: 14272 pages used for memmap
[Sat Jun 25 19:19:53 BST 2022]   DMA32 zone: 913392 pages, LIFO batch:63
[Sat Jun 25 19:19:53 BST 2022]   Normal zone: 2048 pages used for memmap
[Sat Jun 25 19:19:53 BST 2022]   Normal zone: 131072 pages, LIFO batch:31
[Sat Jun 25 19:19:53 BST 2022] On node 0, zone DMA: 1 pages in unavailable ranges
[Sat Jun 25 19:19:53 BST 2022] On node 0, zone DMA: 97 pages in unavailable ranges
[Sat Jun 25 19:19:53 BST 2022] On node 0, zone Normal: 16 pages in unavailable ranges
[Sat Jun 25 19:19:53 BST 2022] ACPI: PM-Timer IO Port: 0x4008
[Sat Jun 25 19:19:53 BST 2022] ACPI: Local APIC address 0xfee00000
[Sat Jun 25 19:19:53 BST 2022] IOAPIC[0]: apic_id 4, version 32, address 0xfec00000, GSI 0-23
[Sat Jun 25 19:19:53 BST 2022] ACPI: INT_SRC_OVR (bus 0 bus_irq 0 global_irq 2 dfl dfl)
[Sat Jun 25 19:19:53 BST 2022] ACPI: INT_SRC_OVR (bus 0 bus_irq 9 global_irq 9 low level)
[Sat Jun 25 19:19:53 BST 2022] ACPI: IRQ0 used by override.
[Sat Jun 25 19:19:53 BST 2022] ACPI: IRQ9 used by override.
[Sat Jun 25 19:19:53 BST 2022] Using ACPI (MADT) for SMP configuration information
[Sat Jun 25 19:19:53 BST 2022] smpboot: Allowing 4 CPUs, 0 hotplug CPUs
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0x00000000-0x00000fff]
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0x0009f000-0x0009ffff]
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0x000a0000-0x000effff]
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0x000f0000-0x000fffff]
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xdfff0000-0xdfffffff]
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xe0000000-0xfebfffff]
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xfec00000-0xfec00fff]
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xfec01000-0xfedfffff]
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xfee00000-0xfee00fff]
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xfee01000-0xfffbffff]
[Sat Jun 25 19:19:53 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xfffc0000-0xffffffff]
[Sat Jun 25 19:19:53 BST 2022] [mem 0xe0000000-0xfebfffff] available for PCI devices
[Sat Jun 25 19:19:53 BST 2022] Booting paravirtualized kernel on KVM
[Sat Jun 25 19:19:53 BST 2022] clocksource: refined-jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns:
7645519600211568 ns
[Sat Jun 25 19:19:53 BST 2022] setup_percpu: NR_CPUS:8192 nr_cpumask_bits:4 nr_cpu_ids:4 nr_node_ids:1
[Sat Jun 25 19:19:53 BST 2022] percpu: Embedded 58 pages/cpu s200536 r8192 d28840 u524288
[Sat Jun 25 19:19:53 BST 2022] pcpu-alloc: s200536 r8192 d28840 u524288 alloc=1*2097152
[Sat Jun 25 19:19:53 BST 2022] pcpu-alloc: [0] 0 1 2 3
[Sat Jun 25 19:19:53 BST 2022] kvm-guest: PV spinlocks enabled
[Sat Jun 25 19:19:53 BST 2022] PV qspinlock hash table entries: 256 (order: 0, 4096 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] Built 1 zonelists, mobility grouping on.  Total pages: 1031920
[Sat Jun 25 19:19:53 BST 2022] Policy zone: Normal
[Sat Jun 25 19:19:53 BST 2022] Kernel command line: BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64 root=UUID=9cc5ee1e-5533-
4a0b-a88f-903bf52d812d ro quiet crashkernel=384M-:128M
[Sat Jun 25 19:19:53 BST 2022] Dentry cache hash table entries: 524288 (order: 10, 4194304 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] Inode-cache hash table entries: 262144 (order: 9, 2097152 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] mem auto-init: stack:off, heap alloc:on, heap free:off
[Sat Jun 25 19:19:53 BST 2022] Memory: 3526712K/4193848K available (12295K kernel code, 2545K rwdata, 7564K rodata,
2408K init, 3684K bss, 346912K reserved, 0K cma-reserved)
[Sat Jun 25 19:19:53 BST 2022] random: get_random_u64 called from __kmem_cache_create+0x2a/0x4d0 with crng_init=0
[Sat Jun 25 19:19:53 BST 2022] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=4, Nodes=1
[Sat Jun 25 19:19:53 BST 2022] Kernel/User page tables isolation: enabled
[Sat Jun 25 19:19:53 BST 2022] ftrace: allocating 36444 entries in 143 pages
[Sat Jun 25 19:19:53 BST 2022] ftrace: allocated 143 pages with 5 groups
[Sat Jun 25 19:19:53 BST 2022] rcu: Hierarchical RCU implementation.
[Sat Jun 25 19:19:53 BST 2022] rcu:     RCU restricting CPUs from NR_CPUS=8192 to nr_cpu_ids=4.
[Sat Jun 25 19:19:53 BST 2022]  Rude variant of Tasks RCU enabled.
[Sat Jun 25 19:19:53 BST 2022]  Tracing variant of Tasks RCU enabled.
[Sat Jun 25 19:19:53 BST 2022] rcu: RCU calculated value of scheduler-enlistment delay is 25 jiffies.
[Sat Jun 25 19:19:53 BST 2022] rcu: Adjusting geometry for rcu_fanout_leaf=16, nr_cpu_ids=4
[Sat Jun 25 19:19:53 BST 2022] NR_IRQS: 524544, nr_irqs: 456, preallocated irqs: 16
[Sat Jun 25 19:19:53 BST 2022] random: crng done (trusting CPU's manufacturer)
[Sat Jun 25 19:19:53 BST 2022] Console: colour VGA+ 80x25
[Sat Jun 25 19:19:53 BST 2022] printk: console [tty0] enabled
[Sat Jun 25 19:19:53 BST 2022] ACPI: Core revision 20200925
```

```
[Sat Jun 25 19:19:53 BST 2022] APIC: Switch to symmetric I/O mode setup
[Sat Jun 25 19:19:53 BST 2022] x2apic enabled
[Sat Jun 25 19:19:53 BST 2022] Switched APIC routing to physical x2apic.
[Sat Jun 25 19:19:53 BST 2022] ..TIMER: vector=0x30 apic1=0 pin1=2 apic2=-1 pin2=-1
[Sat Jun 25 19:19:53 BST 2022] clocksource: tsc-early: mask: 0xffffffffffffffff max_cycles: 0x396d5dac02a,
max_idle_ns: 881590811122 ns
[Sat Jun 25 19:19:53 BST 2022] Calibrating delay loop (skipped) preset value.. 3984.01 BogoMIPS (lpj=7968024)
[Sat Jun 25 19:19:53 BST 2022] pid_max: default: 32768 minimum: 301
[Sat Jun 25 19:19:53 BST 2022] LSM: Security Framework initializing
[Sat Jun 25 19:19:53 BST 2022] Yama: disabled by default; enable with sysctl kernel.yama.*
[Sat Jun 25 19:19:53 BST 2022] AppArmor: AppArmor initialized
[Sat Jun 25 19:19:53 BST 2022] TOMOYO Linux initialized
[Sat Jun 25 19:19:53 BST 2022] Mount-cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] Mountpoint-cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] Last level iTLB entries: 4KB 64, 2MB 8, 4MB 8
[Sat Jun 25 19:19:53 BST 2022] Last level dTLB entries: 4KB 64, 2MB 0, 4MB 0, 1GB 4
[Sat Jun 25 19:19:53 BST 2022] Spectre V1 : Mitigation: usercopy/swapgs barriers and __user pointer sanitization
[Sat Jun 25 19:19:53 BST 2022] Spectre V2 : Mitigation: Full generic retpoline
[Sat Jun 25 19:19:53 BST 2022] Spectre V2 : Spectre v2 / SpectreRSB mitigation: Filling RSB on context switch
[Sat Jun 25 19:19:53 BST 2022] Speculative Store Bypass: Vulnerable
[Sat Jun 25 19:19:53 BST 2022] SRBDS: Unknown: Dependent on hypervisor status
[Sat Jun 25 19:19:53 BST 2022] MDS: Mitigation: Clear CPU buffers
[Sat Jun 25 19:19:53 BST 2022] Freeing SMP alternatives memory: 32K
[Sat Jun 25 19:19:53 BST 2022] smpboot: CPU0: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz (family: 0x6, model: 0x8e,
stepping: 0xa)
[Sat Jun 25 19:19:53 BST 2022] Performance Events: unsupported p6 CPU model 142 no PMU driver, software events only.
[Sat Jun 25 19:19:53 BST 2022] rcu: Hierarchical SRCU implementation.
[Sat Jun 25 19:19:53 BST 2022] NMI watchdog: Perf NMI watchdog permanently disabled
[Sat Jun 25 19:19:53 BST 2022] smp: Bringing up secondary CPUs ...
[Sat Jun 25 19:19:53 BST 2022] x86: Booting SMP configuration:
[Sat Jun 25 19:19:53 BST 2022] .... node  #0, CPUs:      #1
[Sat Jun 25 19:19:53 BST 2022] kvm-clock: cpu 1, msr 648b7041, secondary cpu clock
[Sat Jun 25 19:19:53 BST 2022]  #2
[Sat Jun 25 19:19:53 BST 2022] kvm-clock: cpu 2, msr 648b7081, secondary cpu clock
[Sat Jun 25 19:19:53 BST 2022]  #3
[Sat Jun 25 19:19:53 BST 2022] kvm-clock: cpu 3, msr 648b70c1, secondary cpu clock
[Sat Jun 25 19:19:53 BST 2022] smp: Brought up 1 node, 4 CPUs
[Sat Jun 25 19:19:53 BST 2022] smpboot: Max logical packages: 1
[Sat Jun 25 19:19:53 BST 2022] smpboot: Total of 4 processors activated (15936.04 BogoMIPS)
[Sat Jun 25 19:19:53 BST 2022] node 0 deferred pages initialised in 0ms
[Sat Jun 25 19:19:53 BST 2022] devtmpfs: initialized
[Sat Jun 25 19:19:53 BST 2022] x86/mm: Memory block size: 128MB
[Sat Jun 25 19:19:53 BST 2022] clocksource: jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns:
7645041785100000 ns
[Sat Jun 25 19:19:53 BST 2022] futex hash table entries: 1024 (order: 4, 65536 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] pinctrl core: initialized pinctrl subsystem
[Sat Jun 25 19:19:53 BST 2022] NET: Registered protocol family 16
[Sat Jun 25 19:19:53 BST 2022] audit: initializing netlink subsys (disabled)
[Sat Jun 25 19:19:53 BST 2022] audit: type=2000 audit(1656181200.989:1): state=initialized audit_enabled=0 res=1
[Sat Jun 25 19:19:53 BST 2022] thermal_sys: Registered thermal governor 'fair_share'
[Sat Jun 25 19:19:53 BST 2022] thermal_sys: Registered thermal governor 'bang_bang'
[Sat Jun 25 19:19:53 BST 2022] thermal_sys: Registered thermal governor 'step_wise'
[Sat Jun 25 19:19:53 BST 2022] thermal_sys: Registered thermal governor 'user_space'
[Sat Jun 25 19:19:53 BST 2022] thermal_sys: Registered thermal governor 'power_allocator'
[Sat Jun 25 19:19:53 BST 2022] cpuidle: using governor ladder
[Sat Jun 25 19:19:53 BST 2022] cpuidle: using governor menu
[Sat Jun 25 19:19:53 BST 2022] ACPI: bus type PCI registered
[Sat Jun 25 19:19:53 BST 2022] acpiphp: ACPI Hot Plug PCI Controller Driver version: 0.5
[Sat Jun 25 19:19:53 BST 2022] PCI: Using configuration type 1 for base access
[Sat Jun 25 19:19:53 BST 2022] Kprobes globally optimized
[Sat Jun 25 19:19:53 BST 2022] HugeTLB registered 2.00 MiB page size, pre-allocated 0 pages
[Sat Jun 25 19:19:53 BST 2022] ACPI: Added _OSI(Module Device)
[Sat Jun 25 19:19:53 BST 2022] ACPI: Added _OSI(Processor Device)
[Sat Jun 25 19:19:53 BST 2022] ACPI: Added _OSI(3.0 _SCP Extensions)
[Sat Jun 25 19:19:53 BST 2022] ACPI: Added _OSI(Processor Aggregator Device)
[Sat Jun 25 19:19:53 BST 2022] ACPI: Added _OSI(Linux-Dell-Video)
[Sat Jun 25 19:19:53 BST 2022] ACPI: Added _OSI(Linux-Lenovo-NV-HDMI-Audio)
[Sat Jun 25 19:19:53 BST 2022] ACPI: Added _OSI(Linux-HPI-Hybrid-Graphics)
[Sat Jun 25 19:19:53 BST 2022] ACPI: 2 ACPI AML tables successfully acquired and loaded
[Sat Jun 25 19:19:53 BST 2022] ACPI: Interpreter enabled
[Sat Jun 25 19:19:53 BST 2022] ACPI: (supports S0 S5)
[Sat Jun 25 19:19:53 BST 2022] ACPI: Using IOAPIC for interrupt routing
[Sat Jun 25 19:19:53 BST 2022] PCI: Using host bridge windows from ACPI; if necessary, use "pci=nocrs" and report a
bug
[Sat Jun 25 19:19:53 BST 2022] ACPI: Enabled 2 GPEs in block 00 to 07
[Sat Jun 25 19:19:53 BST 2022] ACPI: PCI Root Bridge [PCI0] (domain 0000 [bus 00-ff])
```

```
[Sat Jun 25 19:19:53 BST 2022] acpi PNP0A03:00: _OSC: OS supports [ASPM ClockPM Segments MSI HPX-Type3]
[Sat Jun 25 19:19:53 BST 2022] acpi PNP0A03:00: _OSC: not requesting OS control; OS requires [ExtendedConfig ASPM
ClockPM MSI]
[Sat Jun 25 19:19:53 BST 2022] acpi PNP0A03:00: fail to add MMCONFIG information, can't access extended PCI
configuration space under this bridge.
[Sat Jun 25 19:19:53 BST 2022] PCI host bridge to bus 0000:00
[Sat Jun 25 19:19:53 BST 2022] pci_bus 0000:00: root bus resource [io  0x0000-0x0cf7 window]
[Sat Jun 25 19:19:53 BST 2022] pci_bus 0000:00: root bus resource [io  0x0d00-0xffff window]
[Sat Jun 25 19:19:53 BST 2022] pci_bus 0000:00: root bus resource [mem 0x000a0000-0x000bffff window]
[Sat Jun 25 19:19:53 BST 2022] pci_bus 0000:00: root bus resource [mem 0xe0000000-0xfdffffff window]
[Sat Jun 25 19:19:53 BST 2022] pci_bus 0000:00: root bus resource [bus 00-ff]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:00.0: [8086:1237] type 00 class 0x060000
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:01.0: [8086:7000] type 00 class 0x060100
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:01.1: [8086:7111] type 00 class 0x01018a
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:01.1: reg 0x20: [io  0xd000-0xd00f]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x10: [io  0x01f0-0x01f7]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x14: [io  0x03f6]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x18: [io  0x0170-0x0177]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x1c: [io  0x0376]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:02.0: [15ad:0405] type 00 class 0x030000
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:02.0: reg 0x10: [io  0xd010-0xd01f]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:02.0: reg 0x14: [mem 0xe0000000-0xe7ffffff pref]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:02.0: reg 0x18: [mem 0xf0000000-0xf01fffff]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:03.0: [8086:100e] type 00 class 0x020000
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:03.0: reg 0x10: [mem 0xf0200000-0xf021ffff]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:03.0: reg 0x18: [io  0xd020-0xd027]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:04.0: [80ee:cafe] type 00 class 0x088000
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:04.0: reg 0x10: [io  0xd040-0xd05f]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:04.0: reg 0x14: [mem 0xf0400000-0xf07fffff]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:04.0: reg 0x18: [mem 0xf0800000-0xf0803fff pref]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:05.0: [8086:2415] type 00 class 0x040100
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:05.0: reg 0x10: [io  0xd100-0xd1ff]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:05.0: reg 0x14: [io  0xd200-0xd23f]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:06.0: [106b:003f] type 00 class 0x0c0310
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:06.0: reg 0x10: [mem 0xf0804000-0xf0804fff]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:07.0: [8086:7113] type 00 class 0x068000
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:07.0: quirk: [io  0x4000-0x403f] claimed by PIIX4 ACPI
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:07.0: quirk: [io  0x4100-0x410f] claimed by PIIX4 SMB
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:0d.0: [8086:2829] type 00 class 0x010601
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:0d.0: reg 0x10: [io  0xd240-0xd247]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:0d.0: reg 0x14: [io  0xd248-0xd24b]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:0d.0: reg 0x18: [io  0xd250-0xd257]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:0d.0: reg 0x1c: [io  0xd258-0xd25b]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:0d.0: reg 0x20: [io  0xd260-0xd26f]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:0d.0: reg 0x24: [mem 0xf0806000-0xf0807fff]
[Sat Jun 25 19:19:53 BST 2022] ACPI: PCI Interrupt Link [LNKA] (IRQs 5 9 10 *11)
[Sat Jun 25 19:19:53 BST 2022] ACPI: PCI Interrupt Link [LNKB] (IRQs 5 9 *10 11)
[Sat Jun 25 19:19:53 BST 2022] ACPI: PCI Interrupt Link [LNKC] (IRQs 5 *9 10 11)
[Sat Jun 25 19:19:53 BST 2022] ACPI: PCI Interrupt Link [LNKD] (IRQs 5 9 10 *11)
[Sat Jun 25 19:19:53 BST 2022] iommu: Default domain type: Translated
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:02.0: vgaarb: setting as boot VGA device
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:02.0: vgaarb: VGA device added: decodes=io+mem,owns=io+mem,locks=none
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:02.0: vgaarb: bridge control possible
[Sat Jun 25 19:19:53 BST 2022] vgaarb: loaded
[Sat Jun 25 19:19:53 BST 2022] EDAC MC: Ver: 3.0.0
[Sat Jun 25 19:19:53 BST 2022] NetLabel: Initializing
[Sat Jun 25 19:19:53 BST 2022] NetLabel:  domain hash size = 128
[Sat Jun 25 19:19:53 BST 2022] NetLabel:  protocols = UNLABELED CIPSOv4 CALIPSO
[Sat Jun 25 19:19:53 BST 2022] NetLabel:  unlabeled traffic allowed by default
[Sat Jun 25 19:19:53 BST 2022] PCI: Using ACPI for IRQ routing
[Sat Jun 25 19:19:53 BST 2022] PCI: pci_cache_line_size set to 64 bytes
[Sat Jun 25 19:19:53 BST 2022] e820: reserve RAM buffer [mem 0x0009fc00-0x0009ffff]
[Sat Jun 25 19:19:53 BST 2022] e820: reserve RAM buffer [mem 0xdfff0000-0xdfffffff]
[Sat Jun 25 19:19:53 BST 2022] clocksource: Switched to clocksource kvm-clock
[Sat Jun 25 19:19:53 BST 2022] VFS: Disk quotas dquot_6.6.0
[Sat Jun 25 19:19:53 BST 2022] VFS: Dquot-cache hash table entries: 512 (order 0, 4096 bytes)
[Sat Jun 25 19:19:53 BST 2022] AppArmor: AppArmor Filesystem Enabled
[Sat Jun 25 19:19:53 BST 2022] pnp: PnP ACPI init
[Sat Jun 25 19:19:53 BST 2022] pnp 00:00: Plug and Play ACPI device, IDs PNP0303 (active)
[Sat Jun 25 19:19:53 BST 2022] pnp 00:01: Plug and Play ACPI device, IDs PNP0f03 (active)
[Sat Jun 25 19:19:53 BST 2022] pnp: PnP ACPI: found 2 devices
[Sat Jun 25 19:19:53 BST 2022] clocksource: acpi_pm: mask: 0xffffff max_cycles: 0xffffff, max_idle_ns: 2085701024 ns
[Sat Jun 25 19:19:53 BST 2022] NET: Registered protocol family 2
[Sat Jun 25 19:19:53 BST 2022] IP idents hash table entries: 65536 (order: 7, 524288 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] tcp_listen_portaddr_hash hash table entries: 2048 (order: 3, 32768 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] TCP established hash table entries: 32768 (order: 6, 262144 bytes, linear)
```

```
[Sat Jun 25 19:19:53 BST 2022] TCP bind hash table entries: 32768 (order: 7, 524288 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] TCP: Hash tables configured (established 32768 bind 32768)
[Sat Jun 25 19:19:53 BST 2022] UDP hash table entries: 2048 (order: 4, 65536 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] UDP-Lite hash table entries: 2048 (order: 4, 65536 bytes, linear)
[Sat Jun 25 19:19:53 BST 2022] NET: Registered protocol family 1
[Sat Jun 25 19:19:53 BST 2022] NET: Registered protocol family 44
[Sat Jun 25 19:19:53 BST 2022] pci_bus 0000:00: resource 4 [io  0x0000-0x0cf7 window]
[Sat Jun 25 19:19:53 BST 2022] pci_bus 0000:00: resource 5 [io  0x0d00-0xffff window]
[Sat Jun 25 19:19:53 BST 2022] pci_bus 0000:00: resource 6 [mem 0x000a0000-0x000bffff window]
[Sat Jun 25 19:19:53 BST 2022] pci_bus 0000:00: resource 7 [mem 0xe0000000-0xfdffffff window]
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:00.0: Limiting direct PCI/PCI transfers
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:01.0: Activating ISA DMA hang workarounds
[Sat Jun 25 19:19:53 BST 2022] pci 0000:00:02.0: Video device with shadowed ROM at [mem 0x000c0000-0x000dffff]
[Sat Jun 25 19:19:53 BST 2022] PCI: CLS 0 bytes, default 64
[Sat Jun 25 19:19:53 BST 2022] Trying to unpack rootfs image as initramfs...
[Sat Jun 25 19:19:54 BST 2022] Freeing initrd memory: 41552K
[Sat Jun 25 19:19:54 BST 2022] PCI-DMA: Using software bounce buffering for IO (SWIOTLB)
[Sat Jun 25 19:19:54 BST 2022] software IO TLB: mapped [mem 0x00000000d3000000-0x00000000d7000000] (64MB)
[Sat Jun 25 19:19:54 BST 2022] clocksource: tsc: mask: 0xffffffffffffffff max_cycles: 0x396d5dac02a, max_idle_ns:
881590811122 ns
[Sat Jun 25 19:19:54 BST 2022] clocksource: Switched to clocksource tsc
[Sat Jun 25 19:19:54 BST 2022] platform rtc_cmos: registered platform RTC device (no PNP device found)
[Sat Jun 25 19:19:54 BST 2022] Initialise system trusted keyrings
[Sat Jun 25 19:19:54 BST 2022] Key type blacklist registered
[Sat Jun 25 19:19:54 BST 2022] workingset: timestamp_bits=36 max_order=20 bucket_order=0
[Sat Jun 25 19:19:54 BST 2022] zbud: loaded
[Sat Jun 25 19:19:54 BST 2022] integrity: Platform Keyring initialized
[Sat Jun 25 19:19:54 BST 2022] Key type asymmetric registered
[Sat Jun 25 19:19:54 BST 2022] Asymmetric key parser 'x509' registered
[Sat Jun 25 19:19:54 BST 2022] Block layer SCSI generic (bsg) driver version 0.4 loaded (major 251)
[Sat Jun 25 19:19:54 BST 2022] io scheduler mq-deadline registered
[Sat Jun 25 19:19:54 BST 2022] shpchp: Standard Hot Plug PCI Controller Driver version: 0.4
[Sat Jun 25 19:19:54 BST 2022] intel_idle: Please enable MWAIT in BIOS SETUP
[Sat Jun 25 19:19:54 BST 2022] Serial: 8250/16550 driver, 4 ports, IRQ sharing enabled
[Sat Jun 25 19:19:54 BST 2022] Linux agpgart interface v0.103
[Sat Jun 25 19:19:54 BST 2022] AMD-Vi: AMD IOMMUv2 functionality not available on this system - This is not a bug.
[Sat Jun 25 19:19:54 BST 2022] i8042: PNP: PS/2 Controller [PNP0303:PS2K,PNP0f03:PS2M] at 0x60,0x64 irq 1,12
[Sat Jun 25 19:19:54 BST 2022] serio: i8042 KBD port at 0x60,0x64 irq 1
[Sat Jun 25 19:19:54 BST 2022] serio: i8042 AUX port at 0x60,0x64 irq 12
[Sat Jun 25 19:19:54 BST 2022] mousedev: PS/2 mouse device common for all mice
[Sat Jun 25 19:19:54 BST 2022] input: AT Translated Set 2 keyboard as /devices/platform/i8042/serio0/input/input0
[Sat Jun 25 19:19:54 BST 2022] rtc_cmos rtc_cmos: registered as rtc0
[Sat Jun 25 19:19:54 BST 2022] rtc_cmos rtc_cmos: setting system clock to 2022-06-25T18:19:54 UTC (1656181194)
[Sat Jun 25 19:19:54 BST 2022] rtc_cmos rtc_cmos: alarms up to one day, 114 bytes nvram
[Sat Jun 25 19:19:54 BST 2022] intel_pstate: CPU model not supported
[Sat Jun 25 19:19:54 BST 2022] ledtrig-cpu: registered to indicate activity on CPUs
[Sat Jun 25 19:19:54 BST 2022] NET: Registered protocol family 10
[Sat Jun 25 19:19:54 BST 2022] Segment Routing with IPv6
[Sat Jun 25 19:19:54 BST 2022] mip6: Mobile IPv6
[Sat Jun 25 19:19:54 BST 2022] NET: Registered protocol family 17
[Sat Jun 25 19:19:54 BST 2022] mpls_gso: MPLS GSO support
[Sat Jun 25 19:19:54 BST 2022] IPI shorthand broadcast: enabled
[Sat Jun 25 19:19:54 BST 2022] sched_clock: Marking stable (1456509208, 13756044)->(1470822801, -557549)
[Sat Jun 25 19:19:54 BST 2022] registered taskstats version 1
[Sat Jun 25 19:19:54 BST 2022] Loading compiled-in X.509 certificates
[Sat Jun 25 19:19:54 BST 2022] Loaded X.509 cert 'Debian Secure Boot CA: 6ccece7e4c6c0d1f6149f3dd27dfcc5cbb419ea1'
[Sat Jun 25 19:19:54 BST 2022] Loaded X.509 cert 'Debian Secure Boot Signer 2021 - linux:
4b6ef5abca669825178e052c84667ccbc0531f8c'
[Sat Jun 25 19:19:54 BST 2022] zswap: loaded using pool lzo/zbud
[Sat Jun 25 19:19:54 BST 2022] Key type ._fscrypt registered
[Sat Jun 25 19:19:54 BST 2022] Key type .fscrypt registered
[Sat Jun 25 19:19:54 BST 2022] Key type fscrypt-provisioning registered
[Sat Jun 25 19:19:54 BST 2022] AppArmor: AppArmor sha1 policy hashing enabled
[Sat Jun 25 19:19:54 BST 2022] Freeing unused kernel image (initmem) memory: 2408K
[Sat Jun 25 19:19:54 BST 2022] Write protecting the kernel read-only data: 22528k
[Sat Jun 25 19:19:54 BST 2022] Freeing unused kernel image (text/rodata gap) memory: 2040K
[Sat Jun 25 19:19:54 BST 2022] Freeing unused kernel image (rodata/data gap) memory: 628K
[Sat Jun 25 19:19:54 BST 2022] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[Sat Jun 25 19:19:54 BST 2022] x86/mm: Checking user space page tables
[Sat Jun 25 19:19:54 BST 2022] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[Sat Jun 25 19:19:54 BST 2022] Run /init as init process
[Sat Jun 25 19:19:54 BST 2022]   with arguments:
[Sat Jun 25 19:19:54 BST 2022]     /init
[Sat Jun 25 19:19:54 BST 2022]   with environment:
[Sat Jun 25 19:19:54 BST 2022]     HOME=/
[Sat Jun 25 19:19:54 BST 2022]     TERM=linux
```

```
[Sat Jun 25 19:19:54 BST 2022]     BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64
[Sat Jun 25 19:19:54 BST 2022]     crashkernel=384M-:128M
[Sat Jun 25 19:19:54 BST 2022] input: Power Button as /devices/LNXSYSTM:00/LNXPWRBN:00/input/input3
[Sat Jun 25 19:19:54 BST 2022] ACPI: Video Device [GFX0] (multi-head: yes  rom: no  post: no)
[Sat Jun 25 19:19:54 BST 2022] input: Video Bus as
/devices/LNXSYSTM:00/LNXSYBUS:00/PNP0A03:00/LNXVIDEO:00/input/input4
[Sat Jun 25 19:19:54 BST 2022] battery: ACPI: Battery Slot [BAT0] (battery present)
[Sat Jun 25 19:19:54 BST 2022] e1000: Intel(R) PRO/1000 Network Driver
[Sat Jun 25 19:19:54 BST 2022] e1000: Copyright (c) 1999-2006 Intel Corporation.
[Sat Jun 25 19:19:54 BST 2022] piix4_smbus 0000:00:07.0: SMBus Host Controller at 0x4100, revision 0
[Sat Jun 25 19:19:54 BST 2022] ACPI: Power Button [PWRF]
[Sat Jun 25 19:19:54 BST 2022] input: Sleep Button as /devices/LNXSYSTM:00/LNXSLPBN:00/input/input5
[Sat Jun 25 19:19:54 BST 2022] ACPI: Sleep Button [SLPF]
[Sat Jun 25 19:19:54 BST 2022] ACPI: bus type USB registered
[Sat Jun 25 19:19:54 BST 2022] usbcore: registered new interface driver usbfs
[Sat Jun 25 19:19:54 BST 2022] usbcore: registered new interface driver hub
[Sat Jun 25 19:19:54 BST 2022] usbcore: registered new device driver usb
[Sat Jun 25 19:19:54 BST 2022] ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
[Sat Jun 25 19:19:54 BST 2022] ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
[Sat Jun 25 19:19:54 BST 2022] ehci-pci: EHCI PCI platform driver
[Sat Jun 25 19:19:54 BST 2022] ohci-pci: OHCI PCI platform driver
[Sat Jun 25 19:19:54 BST 2022] ohci-pci 0000:00:06.0: OHCI PCI host controller
[Sat Jun 25 19:19:54 BST 2022] ohci-pci 0000:00:06.0: new USB bus registered, assigned bus number 1
[Sat Jun 25 19:19:54 BST 2022] ohci-pci 0000:00:06.0: irq 22, io mem 0xf0804000
[Sat Jun 25 19:19:54 BST 2022] SCSI subsystem initialized
[Sat Jun 25 19:19:54 BST 2022] libata version 3.00 loaded.
[Sat Jun 25 19:19:54 BST 2022] ata_piix 0000:00:01.1: version 2.13
[Sat Jun 25 19:19:54 BST 2022] scsi host0: ata_piix
[Sat Jun 25 19:19:54 BST 2022] scsi host1: ata_piix
[Sat Jun 25 19:19:54 BST 2022] ata1: PATA max UDMA/33 cmd 0x1f0 ctl 0x3f6 bmdma 0xd000 irq 14
[Sat Jun 25 19:19:54 BST 2022] ata2: PATA max UDMA/33 cmd 0x170 ctl 0x376 bmdma 0xd008 irq 15
[Sat Jun 25 19:19:54 BST 2022] ahci 0000:00:0d.0: version 3.0
[Sat Jun 25 19:19:54 BST 2022] ahci 0000:00:0d.0: SSS flag set, parallel bus scan disabled
[Sat Jun 25 19:19:54 BST 2022] ahci 0000:00:0d.0: AHCI 0001.0100 32 slots 1 ports 3 Gbps 0x1 impl SATA mode
[Sat Jun 25 19:19:54 BST 2022] ahci 0000:00:0d.0: flags: 64bit ncq stag only ccc
[Sat Jun 25 19:19:54 BST 2022] scsi host2: ahci
[Sat Jun 25 19:19:54 BST 2022] ata3: SATA max UDMA/133 abar m8192@0xf0806000 port 0xf0806100 irq 21
[Sat Jun 25 19:19:54 BST 2022] usb usb1: New USB device found, idVendor=1d6b, idProduct=0001, bcdDevice= 5.10
[Sat Jun 25 19:19:54 BST 2022] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[Sat Jun 25 19:19:54 BST 2022] usb usb1: Product: OHCI PCI host controller
[Sat Jun 25 19:19:54 BST 2022] usb usb1: Manufacturer: Linux 5.10.0-10-amd64 ohci_hcd
[Sat Jun 25 19:19:54 BST 2022] usb usb1: SerialNumber: 0000:00:06.0
[Sat Jun 25 19:19:54 BST 2022] hub 1-0:1.0: USB hub found
[Sat Jun 25 19:19:54 BST 2022] hub 1-0:1.0: 12 ports detected
[Sat Jun 25 19:19:54 BST 2022] [drm] DMA map mode: Caching DMA mappings.
[Sat Jun 25 19:19:54 BST 2022] [drm] Capabilities:
[Sat Jun 25 19:19:54 BST 2022] [drm]   Cursor.
[Sat Jun 25 19:19:54 BST 2022] [drm]   Cursor bypass 2.
[Sat Jun 25 19:19:54 BST 2022] [drm]   Alpha cursor.
[Sat Jun 25 19:19:54 BST 2022] [drm]   3D.
[Sat Jun 25 19:19:54 BST 2022] [drm]   Extended Fifo.
[Sat Jun 25 19:19:54 BST 2022] [drm]   Pitchlock.
[Sat Jun 25 19:19:54 BST 2022] [drm]   Irq mask.
[Sat Jun 25 19:19:54 BST 2022] [drm]   GMR.
[Sat Jun 25 19:19:54 BST 2022] [drm]   Traces.
[Sat Jun 25 19:19:54 BST 2022] [drm]   GMR2.
[Sat Jun 25 19:19:54 BST 2022] [drm]   Screen Object 2.
[Sat Jun 25 19:19:54 BST 2022] [drm] Max GMR ids is 8192
[Sat Jun 25 19:19:54 BST 2022] [drm] Max number of GMR pages is 1048576
[Sat Jun 25 19:19:54 BST 2022] [drm] Max dedicated hypervisor surface memory is 393216 kiB
[Sat Jun 25 19:19:54 BST 2022] [drm] Maximum display memory size is 131072 kiB
[Sat Jun 25 19:19:54 BST 2022] [drm] VRAM at 0xe0000000 size is 131072 kiB
[Sat Jun 25 19:19:54 BST 2022] [drm] MMIO at 0xf0000000 size is 2048 kiB
[Sat Jun 25 19:19:54 BST 2022] [TTM] Zone  kernel: Available graphics memory: 1946798 KiB
[Sat Jun 25 19:19:54 BST 2022] [TTM] Initializing pool allocator
[Sat Jun 25 19:19:54 BST 2022] [TTM] Initializing DMA pool allocator
[Sat Jun 25 19:19:54 BST 2022] [drm] Screen Objects Display Unit initialized
[Sat Jun 25 19:19:54 BST 2022] [drm] width 720
[Sat Jun 25 19:19:54 BST 2022] [drm] height 400
[Sat Jun 25 19:19:54 BST 2022] [drm] bpp 32
[Sat Jun 25 19:19:54 BST 2022] [drm] Fifo max 0x00200000 min 0x00001000 cap 0x00000355
[Sat Jun 25 19:19:54 BST 2022] [drm] Atomic: yes.
[Sat Jun 25 19:19:54 BST 2022] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.
[Sat Jun 25 19:19:54 BST 2022] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.
[Sat Jun 25 19:19:54 BST 2022] fbcon: svgadrmfb (fb0) is primary device
[Sat Jun 25 19:19:54 BST 2022] Console: switching to colour frame buffer device 100x37
```

```
[Sat Jun 25 19:19:54 BST 2022] [drm] Initialized vmwgfx 2.18.0 20200114 for 0000:00:02.0 on minor 0
[Sat Jun 25 19:19:55 BST 2022] input: ImExPS/2 Generic Explorer Mouse as /devices/platform/i8042/serio1/input/input6
[Sat Jun 25 19:19:55 BST 2022] ata2.00: ATAPI: VBOX CD-ROM, 1.0, max UDMA/133
[Sat Jun 25 19:19:55 BST 2022] scsi 1:0:0:0: CD-ROM            VBOX     CD-ROM           1.0  PQ: 0 ANSI: 5
[Sat Jun 25 19:19:55 BST 2022] ata3: SATA link up 3.0 Gbps (SStatus 123 SControl 300)
[Sat Jun 25 19:19:55 BST 2022] ata3.00: ATA-6: VBOX HARDDISK, 1.0, max UDMA/133
[Sat Jun 25 19:19:55 BST 2022] ata3.00: 209715200 sectors, multi 128: LBA48 NCQ (depth 32)
[Sat Jun 25 19:19:55 BST 2022] ata3.00: configured for UDMA/133
[Sat Jun 25 19:19:55 BST 2022] scsi 2:0:0:0: Direct-Access     ATA      VBOX HARDDISK    1.0  PQ: 0 ANSI: 5
[Sat Jun 25 19:19:55 BST 2022] sd 2:0:0:0: [sda] 209715200 512-byte logical blocks: (107 GB/100 GiB)
[Sat Jun 25 19:19:55 BST 2022] sd 2:0:0:0: [sda] Write Protect is off
[Sat Jun 25 19:19:55 BST 2022] sd 2:0:0:0: [sda] Mode Sense: 00 3a 00 00
[Sat Jun 25 19:19:55 BST 2022] sd 2:0:0:0: [sda] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
[Sat Jun 25 19:19:55 BST 2022] e1000 0000:00:03.0 eth0: (PCI:33MHz:32-bit) 08:00:27:26:5a:6b
[Sat Jun 25 19:19:55 BST 2022] e1000 0000:00:03.0 eth0: Intel(R) PRO/1000 Network Connection
[Sat Jun 25 19:19:55 BST 2022] e1000 0000:00:03.0 enp0s3: renamed from eth0
[Sat Jun 25 19:19:55 BST 2022] sr 1:0:0:0: [sr0] scsi3-mmc drive: 32x/32x xa/form2 tray
[Sat Jun 25 19:19:55 BST 2022] cdrom: Uniform CD-ROM driver Revision: 3.20
[Sat Jun 25 19:19:55 BST 2022]  sda: sda1 sda2 < sda5 >
[Sat Jun 25 19:19:55 BST 2022] usb 1-1: new full-speed USB device number 2 using ohci-pci
[Sat Jun 25 19:19:55 BST 2022] sd 2:0:0:0: [sda] Attached SCSI disk
[Sat Jun 25 19:19:55 BST 2022] sr 1:0:0:0: Attached scsi CD-ROM sr0
[Sat Jun 25 19:19:55 BST 2022] usb 1-1: New USB device found, idVendor=80ee, idProduct=0021, bcdDevice= 1.00
[Sat Jun 25 19:19:55 BST 2022] usb 1-1: New USB device strings: Mfr=1, Product=3, SerialNumber=0
[Sat Jun 25 19:19:55 BST 2022] usb 1-1: Product: USB Tablet
[Sat Jun 25 19:19:55 BST 2022] usb 1-1: Manufacturer: VirtualBox
[Sat Jun 25 19:19:55 BST 2022] hid: raw HID events driver (C) Jiri Kosina
[Sat Jun 25 19:19:55 BST 2022] usbcore: registered new interface driver usbhid
[Sat Jun 25 19:19:55 BST 2022] usbhid: USB HID core driver
[Sat Jun 25 19:19:55 BST 2022] input: VirtualBox USB Tablet as /devices/pci0000:00/0000:00:06.0/usb1/1-1/1-
1:1.0/0003:80EE:0021.0001/input/input7
[Sat Jun 25 19:19:55 BST 2022] hid-generic 0003:80EE:0021.0001: input,hidraw0: USB HID v1.10 Mouse [VirtualBox USB
Tablet] on usb-0000:00:06.0-1/input0
[Sat Jun 25 19:19:55 BST 2022] PM: Image not found (code -22)
[Sat Jun 25 19:19:56 BST 2022] EXT4-fs (sda1): mounted filesystem with ordered data mode. Opts: (null)
[Sat Jun 25 19:19:56 BST 2022] Not activating Mandatory Access Control as /sbin/tomoyo-init does not exist.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Inserted module 'autofs4'
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: systemd 247.3-6 running in system mode. (+PAM +AUDIT +SELINUX +IMA
+APPARMOR +SMACK +SYSVINIT +UTMP +LIBCRYPTSETUP +GCRYPT +GNUTLS +ACL +XZ +LZ4 +ZSTD +SECCOMP +BLKID +ELFUTILS +KMOD
+IDN2 -IDN +PCRE2 default-hierarchy=unified)
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Detected virtualization oracle.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Detected architecture x86-64.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Set hostname to <coredump>.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: /lib/systemd/system/plymouth-start.service:16: Unit configured to use
KillMode=none. This is unsafe, as it disables systemd's process lifecycle management for the service. Please update
your service to use a safer KillMode=, such as 'mixed' or 'control-group'. Support for KillMode=none is deprecated and
will eventually be removed.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Queued start job for default target Graphical Interface.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Created slice system-getty.slice.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Created slice system-modprobe.slice.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Created slice User and Session Slice.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Started Forward Password Requests to Wall Directory Watch.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Set up automount Arbitrary Executable File Formats File System Automount
Point.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Reached target User and Group Name Lookups.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Reached target Remote File Systems.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Reached target Slices.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Reached target System Time Set.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Reached target System Time Synchronized.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Listening on Syslog Socket.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Listening on fsck to fsckd communication Socket.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Listening on initctl Compatibility Named Pipe.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Listening on Journal Audit Socket.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Listening on Journal Socket (/dev/log).
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Listening on Journal Socket.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Listening on udev Control Socket.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Listening on udev Kernel Socket.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Mounting Huge Pages File System...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Mounting POSIX Message Queue File System...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Mounting Kernel Debug File System...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Mounting Kernel Trace File System...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Starting Set the console keyboard layout...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Starting Create list of static device nodes for the current kernel...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Starting Load Kernel Module configfs...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Starting Load Kernel Module drm...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Starting Load Kernel Module fuse...
```

```
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Condition check resulted in Set Up Additional Binary Formats being skipped.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Condition check resulted in File System Check on Root Device being skipped.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Starting Journal Service...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Starting Load Kernel Modules...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Starting Remount Root and Kernel File Systems...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Starting Coldplug All udev Devices...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Mounted Huge Pages File System.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Mounted POSIX Message Queue File System.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Mounted Kernel Debug File System.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Mounted Kernel Trace File System.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Finished Create list of static device nodes for the current kernel.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: modprobe@configfs.service: Succeeded.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Finished Load Kernel Module configfs.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: modprobe@drm.service: Succeeded.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Finished Load Kernel Module drm.
[Sat Jun 25 19:19:56 BST 2022] fuse: init (API version 7.32)
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Mounting Kernel Configuration File System...
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: modprobe@fuse.service: Succeeded.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Finished Load Kernel Module fuse.
[Sat Jun 25 19:19:56 BST 2022] systemd[1]: Mounting FUSE Control File System...
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Finished Load Kernel Modules.
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Starting Apply Kernel Variables...
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Mounted Kernel Configuration File System.
[Sat Jun 25 19:19:57 BST 2022] EXT4-fs (sda1): re-mounted. Opts: errors=remount-ro
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Finished Remount Root and Kernel File Systems.
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Mounted FUSE Control File System.
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Condition check resulted in Rebuild Hardware Database being skipped.
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Condition check resulted in Platform Persistent Storage Archival being
skipped.
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Starting Load/Save Random Seed...
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Starting Create System Users...
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Finished Apply Kernel Variables.
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Finished Load/Save Random Seed.
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Condition check resulted in First Boot Complete being skipped.
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Finished Create System Users.
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Starting Create Static Device Nodes in /dev...
[Sat Jun 25 19:19:57 BST 2022] systemd[1]: Started Journal Service.
[Sat Jun 25 19:19:57 BST 2022] systemd-journald[243]: Received client request to flush runtime journal.
[Sat Jun 25 19:19:57 BST 2022] systemd-journald[243]: File
/var/log/journal/7a35ae5c9d954e019d1b34858d5e1923/system.journal corrupted or uncleanly shut down, renaming and
replacing.
[Sat Jun 25 19:19:57 BST 2022] audit: type=1400 audit(1656181197.320:2): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="/usr/bin/man" pid=279 comm="apparmor_parser"
[Sat Jun 25 19:19:57 BST 2022] audit: type=1400 audit(1656181197.320:3): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="man_filter" pid=279 comm="apparmor_parser"
[Sat Jun 25 19:19:57 BST 2022] audit: type=1400 audit(1656181197.320:4): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="man_groff" pid=279 comm="apparmor_parser"
[Sat Jun 25 19:19:57 BST 2022] audit: type=1400 audit(1656181197.320:5): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="nvidia_modprobe" pid=282 comm="apparmor_parser"
[Sat Jun 25 19:19:57 BST 2022] audit: type=1400 audit(1656181197.320:6): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="nvidia_modprobe//kmod" pid=282 comm="apparmor_parser"
[Sat Jun 25 19:19:57 BST 2022] audit: type=1400 audit(1656181197.324:7): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-senddoc" pid=280 comm="apparmor_parser"
[Sat Jun 25 19:19:57 BST 2022] audit: type=1400 audit(1656181197.332:8): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="lsb_release" pid=283 comm="apparmor_parser"
[Sat Jun 25 19:19:57 BST 2022] audit: type=1400 audit(1656181197.332:9): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-oopslash" pid=285 comm="apparmor_parser"
[Sat Jun 25 19:19:57 BST 2022] audit: type=1400 audit(1656181197.336:10): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-xpdfimport" pid=287 comm="apparmor_parser"
[Sat Jun 25 19:19:57 BST 2022] ACPI: AC Adapter [AC] (off-line)
[Sat Jun 25 19:19:57 BST 2022] sr 1:0:0:0: Attached scsi generic sg0 type 5
[Sat Jun 25 19:19:57 BST 2022] sd 2:0:0:0: Attached scsi generic sg1 type 0
[Sat Jun 25 19:19:57 BST 2022] vboxguest: loading out-of-tree module taints kernel.
[Sat Jun 25 19:19:57 BST 2022] vboxguest: module verification failed: signature and/or required key missing - tainting
kernel
[Sat Jun 25 19:19:57 BST 2022] input: PC Speaker as /devices/platform/pcspkr/input/input8
[Sat Jun 25 19:19:57 BST 2022] vgdrvHeartbeatInit: Setting up heartbeat to trigger every 2000 milliseconds
[Sat Jun 25 19:19:57 BST 2022] input: Unspecified device as /devices/pci0000:00/0000:00:04.0/input/input9
[Sat Jun 25 19:19:57 BST 2022] vboxguest: Successfully loaded version 6.1.30 r148432
[Sat Jun 25 19:19:57 BST 2022] vboxguest: misc device minor 61, IRQ 20, I/O port d040, MMIO at 00000000f0400000 (size
0x400000)
[Sat Jun 25 19:19:57 BST 2022] vboxguest: Successfully loaded version 6.1.30 r148432 (interface 0x00010004)
[Sat Jun 25 19:19:57 BST 2022] Adding 998396k swap on /dev/sda5.  Priority:-2 extents:1 across:998396k FS
[Sat Jun 25 19:19:57 BST 2022] RAPL PMU: API unit is 2^-32 Joules, 0 fixed counters, 10737418240 ms ovfl timer
[Sat Jun 25 19:19:57 BST 2022] cryptd: max_cpu_qlen set to 1000
[Sat Jun 25 19:19:57 BST 2022] AVX2 version of gcm_enc/dec engaged.
```

```
[Sat Jun 25 19:19:57 BST 2022] AES CTR mode by8 optimization enabled
[Sat Jun 25 19:19:57 BST 2022] snd_intel8x0 0000:00:05.0: allow list rate for 1028:0177 is 48000
[Sat Jun 25 19:19:58 BST 2022] intel_pmc_core intel_pmc_core.0:  initialized
[Sat Jun 25 19:19:58 BST 2022] e1000: enp0s3 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[Sat Jun 25 19:19:58 BST 2022] IPv6: ADDRCONF(NETDEV_CHANGE): enp0s3: link becomes ready
[Sat Jun 25 19:20:06 BST 2022] vboxvideo: loading version 6.1.30 r148432
[Sat Jun 25 19:20:06 BST 2022] 18:20:06.637808 main     VBoxService 6.1.30 r148432 (verbosity: 0) linux.amd64 (Nov 22
2021 16:16:32) release log
                               18:20:06.637812 main     Log opened 2022-06-25T18:20:06.637803000Z
[Sat Jun 25 19:20:06 BST 2022] 18:20:06.637914 main     OS Product: Linux
[Sat Jun 25 19:20:06 BST 2022] 18:20:06.637949 main     OS Release: 5.10.0-10-amd64
[Sat Jun 25 19:20:06 BST 2022] 18:20:06.637975 main     OS Version: #1 SMP Debian 5.10.84-1 (2021-12-08)
[Sat Jun 25 19:20:06 BST 2022] 18:20:06.638001 main     Executable: /opt/VBoxGuestAdditions-6.1.30/sbin/VBoxService
                               18:20:06.638002 main     Process ID: 745
                               18:20:06.638003 main     Package type: LINUX_64BITS_GENERIC
[Sat Jun 25 19:20:06 BST 2022] 18:20:06.640870 main     6.1.30 r148432 started. Verbose level = 0
[Sat Jun 25 19:20:06 BST 2022] 18:20:06.642328 main     vbglR3GuestCtrlDetectPeekGetCancelSupport: Supported (#1)
[Sat Jun 25 19:20:06 BST 2022] vboxsf: g_fHostFeatures=0x8000000f g_fSfFeatures=0x1 g_uSfLastFunction=29
[Sat Jun 25 19:20:06 BST 2022] vboxsf: Successfully loaded version 6.1.30 r148432
[Sat Jun 25 19:20:06 BST 2022] vboxsf: Successfully loaded version 6.1.30 r148432 on 5.10.0-10-amd64
(LINUX_VERSION_CODE=0x50a54)
[Sat Jun 25 19:20:06 BST 2022] 18:20:06.660750 automount vbsvcAutomounterMountIt: Successfully mounted 'shared' on
'/media/sf_shared'
[Sat Jun 25 19:20:12 BST 2022] rfkill: input handler disabled
[Sat Jun 25 19:20:20 BST 2022] systemd-journald[243]: File /var/log/journal/7a35ae5c9d954e019d1b34858d5e1923/user-
1000.journal corrupted or uncleanly shut down, renaming and replacing.
[Sat Jun 25 19:20:21 BST 2022] rfkill: input handler enabled
[Sat Jun 25 19:20:23 BST 2022] rfkill: input handler disabled
[Sat Jun 25 19:22:31 BST 2022] BUG: kernel NULL pointer dereference, address: 0000000000000000
[Sat Jun 25 19:22:31 BST 2022] #PF: supervisor instruction fetch in kernel mode
[Sat Jun 25 19:22:31 BST 2022] #PF: error_code(0x0010) - not-present page
[Sat Jun 25 19:22:31 BST 2022] PGD 0 P4D 0
[Sat Jun 25 19:22:31 BST 2022] Oops: 0010 [#1] SMP PTI
[Sat Jun 25 19:22:31 BST 2022] CPU: 1 PID: 2189 Comm: mod_b thread Kdump: loaded Tainted: G           OE     5.10.0-
10-amd64 #1 Debian 5.10.84-1
[Sat Jun 25 19:22:31 BST 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:22:31 BST 2022] RIP: 0010:0x0
[Sat Jun 25 19:22:31 BST 2022] Code: Unable to access opcode bytes at RIP 0xffffffffffffffd6.
[Sat Jun 25 19:22:31 BST 2022] RSP: 0018:ffffbb1d00b1ff08 EFLAGS: 00010246
[Sat Jun 25 19:22:31 BST 2022] RAX: 0000000000000000 RBX: ffffffffc0a0e000 RCX: 0000000000000000
[Sat Jun 25 19:22:31 BST 2022] RDX: 0000000000000000 RSI: 0000000000000246 RDI: 0000000000000000
[Sat Jun 25 19:22:31 BST 2022] RBP: ffff8facd1db2980 R08: 0000000000000000 R09: 0000000000000000
[Sat Jun 25 19:22:31 BST 2022] R10: 0000000000000000 R11: 0000000000000000 R12: ffff8facf1dabd40
[Sat Jun 25 19:22:31 BST 2022] R13: ffffbb1d00b2fd28 R14: 0000000000000000 R15: ffff8facda610000
[Sat Jun 25 19:22:31 BST 2022] FS:  0000000000000000(0000) GS:ffff8faddbc80000(0000) knlGS:0000000000000000
[Sat Jun 25 19:22:31 BST 2022] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[Sat Jun 25 19:22:31 BST 2022] CR2: ffffffffffffffd6 CR3: 00000000269f4006 CR4: 00000000000706e0
[Sat Jun 25 19:22:31 BST 2022] Call Trace:
[Sat Jun 25 19:22:31 BST 2022]  kthread_f+0x14/0x20 [mod_b]
[Sat Jun 25 19:22:31 BST 2022]  kthread+0x11b/0x140
[Sat Jun 25 19:22:31 BST 2022]  ? __kthread_bind_mask+0x60/0x60
[Sat Jun 25 19:22:31 BST 2022]  ret_from_fork+0x22/0x30
[Sat Jun 25 19:22:31 BST 2022] Modules linked in: mod_b(OE) vboxsf(OE) vboxvideo(OE) rfkill intel_rapl_msr
intel_rapl_common intel_pmc_core_pltdrv intel_pmc_core ghash_clmulni_intel aesni_intel libaes crypto_simd cryptd
glue_helper rapl snd_intel8x0 snd_ac97_codec ac97_bus snd_pcm joydev snd_timer pcspkr evdev serio_raw vboxguest(OE)
snd sg soundcore ac msr fuse configfs ip_tables x_tables autofs4 ext4 crc16 mbcache jbd2 crc32c_generic hid_generic
usbhid hid sr_mod sd_mod cdrom t10_pi crc_t10dif crct10dif_generic ata_generic vmwgfx ttm drm_kms_helper cec drm ahci
libahci ata_piix crct10dif_pclmul crct10dif_common libata crc32_pclmul crc32c_intel psmouse ohci_pci ehci_pci ohci_hcd
ehci_hcd scsi_mod usbcore i2c_piix4 e1000 usb_common battery video button
[Sat Jun 25 19:22:31 BST 2022] CR2: 0000000000000000
```

3. We also get an exception stack trace from the tool where we have more information without and with source code references (we use **-sx** to include offsets):

```
crash> bt -sx
PID: 2189   TASK: ffff8facda610000  CPU: 1   COMMAND: "mod_b thread"
 #0 [ffffbb1d00b1fcd8] machine_kexec+0x1bb at ffffffffb726436b
 #1 [ffffbb1d00b1fd30] __crash_kexec+0x6d at ffffffffb733aaad
 #2 [ffffbb1d00b1fdf8] crash_kexec+0x35 at ffffffffb733bbe5
 #3 [ffffbb1d00b1fe08] oops_end+0x9b at ffffffffb722da9b
 #4 [ffffbb1d00b1fe28] exc_page_fault+0x78 at ffffffffb7ab6c98
 #5 [ffffbb1d00b1fe50] asm_exc_page_fault+0x1e at ffffffffb7c00ade
 #6 [ffffbb1d00b1ff08] kthread_f+0x14 at ffffffffc0a0e014 [mod_b]
 #7 [ffffbb1d00b1ff10] kthread+0x11b at ffffffffb72ac91b
 #8 [ffffbb1d00b1ff50] ret_from_fork+0x22 at ffffffffb7204442
```

4. However, the problem RIP address is 0, so we need to look at the call stack below **asm_exc_page_fault** where we have to follow calls and jumps to arrive at the problem function:

```
crash> dis kthread_f
0xffffffffc0a0e000 <kthread_f>: nopl   0x0(%rax,%rax,1) [FTRACE NOP]
0xffffffffc0a0e005 <kthread_f+5>:       mov    $0x2710,%edi
0xffffffffc0a0e00a <kthread_f+10>:      call   0xffffffffb7317cf0 <msleep>
0xffffffffc0a0e00f <kthread_f+15>:      call   0xffffffffc0a0e060
0xffffffffc0a0e014 <kthread_f+20>:      xor    %eax,%eax
0xffffffffc0a0e016 <kthread_f+22>:      ret
0xffffffffc0a0e017 <kthread_f+23>:      nopw   0x0(%rax,%rax,1)

crash> dis 0xffffffffc0a0e060
0xffffffffc0a0e060 <foo>:       nopl   0x0(%rax,%rax,1) [FTRACE NOP]
0xffffffffc0a0e065 <foo+5>:     jmp    0xffffffffc0a0e070 <bar>
0xffffffffc0a0e06a <foo+10>:    nopw   0x0(%rax,%rax,1)

crash> dis 0xffffffffc0a0e070
0xffffffffc0a0e070 <bar>:       nopl   0x0(%rax,%rax,1) [FTRACE NOP]
0xffffffffc0a0e075 <bar+5>:     xor    %eax,%eax
0xffffffffc0a0e077 <bar+7>:     jmp    0xffffffffb7e01ca0 <__x86_indirect_thunk_rax>
0xffffffffc0a0e07c <bar+12>:    add    %al,(%rax)
0xffffffffc0a0e07e <bar+14>:    add    %al,(%rax)
0xffffffffc0a0e080 <bar+16>:    add    %al,(%rax)
0xffffffffc0a0e082 <bar+18>:    add    %al,(%rax)
0xffffffffc0a0e084 <bar+20>:    add    %al,(%rax)
0xffffffffc0a0e086 <bar+22>:    add    %al,(%rax)
0xffffffffc0a0e088 <bar+24>:    add    %al,(%rax)
0xffffffffc0a0e08a <bar+26>:    add    %al,(%rax)
0xffffffffc0a0e08c <bar+28>:    add    %al,(%rax)
0xffffffffc0a0e08e <bar+30>:    add    %al,(%rax)
0xffffffffc0a0e090 <bar+32>:    add    %al,(%rax)
0xffffffffc0a0e092 <bar+34>:    add    %al,(%rax)
0xffffffffc0a0e094 <bar+36>:    add    %al,(%rax)
0xffffffffc0a0e096 <bar+38>:    add    %al,(%rax)
0xffffffffc0a0e098 <bar+40>:    add    %al,(%rax)
0xffffffffc0a0e09a <bar+42>:    add    %al,(%rax)
0xffffffffc0a0e09c <bar+44>:    add    %al,(%rax)
0xffffffffc0a0e09e <bar+46>:    add    %al,(%rax)
0xffffffffc0a0e0a0 <bar+48>:    add    %al,(%rax)
0xffffffffc0a0e0a2 <bar+50>:    add    %al,(%rax)
0xffffffffc0a0e0a4 <bar+52>:    add    %al,(%rax)
0xffffffffc0a0e0a6 <bar+54>:    add    %al,(%rax)
```

```
0xffffffffc0a0e0a8 <bar+56>:    add    %al,(%rax)
0xffffffffc0a0e0aa <bar+58>:    add    %al,(%rax)
0xffffffffc0a0e0ac <bar+60>:    add    %al,(%rax)
0xffffffffc0a0e0ae <bar+62>:    add    %al,(%rax)
0xffffffffc0a0e0b0 <bar+64>:    add    %al,(%rax)
0xffffffffc0a0e0b2 <bar+66>:    add    %al,(%rax)
0xffffffffc0a0e0b4 <bar+68>:    add    %al,(%rax)
0xffffffffc0a0e0b6 <bar+70>:    add    %al,(%rax)
0xffffffffc0a0e0b8 <bar+72>:    add    %al,(%rax)
0xffffffffc0a0e0ba <bar+74>:    add    %al,(%rax)
0xffffffffc0a0e0bc <bar+76>:    add    %al,(%rax)
0xffffffffc0a0e0be <bar+78>:    add    %al,(%rax)
0xffffffffc0a0e0c0 <bar+80>:    add    %al,(%rax)
0xffffffffc0a0e0c2 <bar+82>:    add    %al,(%rax)
0xffffffffc0a0e0c4 <bar+84>:    add    %al,(%rax)
0xffffffffc0a0e0c6 <bar+86>:    add    %al,(%rax)
0xffffffffc0a0e0c8 <bar+88>:    add    %al,(%rax)
0xffffffffc0a0e0ca <bar+90>:    add    %al,(%rax)
0xffffffffc0a0e0cc <bar+92>:    add    %al,(%rax)
0xffffffffc0a0e0ce <bar+94>:    add    %al,(%rax)
0xffffffffc0a0e0d0 <bar+96>:    add    %al,(%rax)
0xffffffffc0a0e0d2 <bar+98>:    add    %al,(%rax)
0xffffffffc0a0e0d4 <bar+100>:   add    %al,(%rax)
0xffffffffc0a0e0d6 <bar+102>:   add    %al,(%rax)
0xffffffffc0a0e0d8 <bar+104>:   add    %al,(%rax)
0xffffffffc0a0e0da <bar+106>:   add    %al,(%rax)
0xffffffffc0a0e0dc <bar+108>:   add    %al,(%rax)
0xffffffffc0a0e0de <bar+110>:   add    %al,(%rax)
0xffffffffc0a0e0e0 <bar+112>:   add    %al,(%rax)
0xffffffffc0a0e0e2 <bar+114>:   add    %al,(%rax)
0xffffffffc0a0e0e4 <bar+116>:   add    %al,(%rax)
0xffffffffc0a0e0e6 <bar+118>:   add    %al,(%rax)
 -- MORE --  forward: <SPACE>, <ENTER> or j  backward: b or k  quit: qq
```

```
crash> dis 0xffffffffb7e01ca0
0xffffffffb7e01ca0 <__x86_indirect_thunk_rax>:  jmp    0xffffffffb7e01ca5 <__x86_retpoline_rax>
0xffffffffb7e01ca2 <__x86_indirect_thunk_rax+2>:        nopl   (%rax)

crash> dis 0xffffffffb7e01ca5
0xffffffffb7e01ca5 <__x86_retpoline_rax>:       call   0xffffffffb7e01cb1 <__x86_retpoline_rax+12>
0xffffffffb7e01caa <__x86_retpoline_rax+5>:     pause
0xffffffffb7e01cac <__x86_retpoline_rax+7>:     lfence
0xffffffffb7e01caf <__x86_retpoline_rax+10>:    jmp    0xffffffffb7e01caa <__x86_retpoline_rax+5>
0xffffffffb7e01cb1 <__x86_retpoline_rax+12>:    mov    %rax,(%rsp)
0xffffffffb7e01cb5 <__x86_retpoline_rax+16>:    ret
```

5. Since **ret** instruction takes its return address from **(%rsp)** value, the %RSP-8 from message output should point to memory value 0 (we need to subtract 8 bytes from %RSP address because %RSP is incremented before transferring execution to the stored return address):

```
[Sat Jun 25 19:22:31 BST 2022] RSP: 0018:ffffbb1d00b1ff08 EFLAGS: 00010246

crash> rd ffffbb1d00b1ff00
ffffbb1d00b1ff00:  0000000000000000                        ........
```

Exercise K4

- **Goal:** Learn how to identify spiking kernel threads
- **Patterns:** Stack Trace Collection (CPUs); Interrupt Stack; Spiking Thread
- \ALCDA-Dumps\Exercise-K4-x64-GDB.pdf

Exercise K4 (x64, GDB)

Goal: Learn how to identify spiking kernel threads.

Patterns: Stack Trace Collection (CPUs); Spiking Thread.

1. Load a core dump *dump.202206251950* from the x64/K4 directory and the matching *vmlinux-5.10.0-10-amd64* file from the x64/KSym directory:

```
~/ALCDA2/x64/K4$ crash dump.202206251950 ../KSym/vmlinux-5.10.0-10-amd64

crash 8.0.0++
Copyright (C) 2002-2021  Red Hat, Inc.
Copyright (C) 2004, 2005, 2006, 2010  IBM Corporation
Copyright (C) 1999-2006  Hewlett-Packard Co
Copyright (C) 2005, 2006, 2011, 2012  Fujitsu Limited
Copyright (C) 2006, 2007  VA Linux Systems Japan K.K.
Copyright (C) 2005, 2011, 2020-2021  NEC Corporation
Copyright (C) 1999, 2002, 2007  Silicon Graphics, Inc.
Copyright (C) 1999, 2000, 2001, 2002  Mission Critical Linux, Inc.
Copyright (C) 2015, 2021  VMware, Inc.
This program is free software, covered by the GNU General Public License,
and you are welcome to change it and/or distribute copies of it under
certain conditions.  Enter "help copying" to see the conditions.
This program has absolutely no warranty.  Enter "help warranty" for details.

GNU gdb (GDB) 10.2
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-pc-linux-gnu".
Type "show configuration" for configuration details.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...

      KERNEL: ../KSym/vmlinux-5.10.0-10-amd64  [TAINTED]
    DUMPFILE: dump.202206251950  [PARTIAL DUMP]
        CPUS: 4
        DATE: Sat Jun 25 19:50:39 BST 2022
      UPTIME: 00:27:39
LOAD AVERAGE: 5.58, 2.54, 1.03
       TASKS: 460
    NODENAME: coredump
     RELEASE: 5.10.0-10-amd64
     VERSION: #1 SMP Debian 5.10.84-1 (2021-12-08)
     MACHINE: x86_64  (1992 Mhz)
      MEMORY: 4 GB
       PANIC: "Kernel panic - not syncing: sysrq triggered crash"
         PID: 2172
     COMMAND: "bash"
        TASK: ffff9eb669a217c0  [THREAD_INFO: ffff9eb669a217c0]
         CPU: 2
```

```
       STATE: TASK_RUNNING (PANIC)

crash>
```

2. Since this is a manual dump, we check stack traces on all CPUs:

```
crash> bt -a
PID: 2999   TASK: ffff9eb7510e17c0  CPU: 0   COMMAND: "mod_c thread"
 #0 [fffffe000000de50] crash_nmi_callback at ffffffff92e58e43
 #1 [fffffe000000de58] nmi_handle at ffffffff92e2e168
 #2 [fffffe000000dea0] default_do_nmi at ffffffff936b4fe2
 #3 [fffffe000000dec8] exc_nmi at ffffffff936b51ff
 #4 [fffffe000000def0] end_repeat_nmi at ffffffff938014db
    [exception RIP: foo+5]
    RIP: ffffffffc0b10065  RSP: ffffb5dbc3c13f08  RFLAGS: 00000246
    RAX: 0000000000000000  RBX: ffffffffc0b10000  RCX: 0000000000000000
    RDX: 0000000000000000  RSI: 0000000000000246  RDI: 0000000000000000
    RBP: ffff9eb751648280   R8: 0000000000000000   R9: 0000000000000000
    R10: 0000000000000000  R11: 0000000000000000  R12: ffff9eb678078fc0
    R13: ffffb5dbc3ce3d28  R14: 0000000000000000  R15: ffff9eb7510e17c0
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
--- <NMI exception stack> ---
 #5 [ffffb5dbc3c13f08] foo at ffffffffc0b10065 [mod_c]
 #6 [ffffb5dbc3c13f08] kthread_f at ffffffffc0b10014 [mod_c]
 #7 [ffffb5dbc3c13f10] kthread at ffffffff92eac91b
 #8 [ffffb5dbc3c13f50] ret_from_fork at ffffffff92e04442

PID: 0      TASK: ffff9eb74024df00  CPU: 1   COMMAND: "swapper/1"
 #0 [fffffe0000048e50] crash_nmi_callback at ffffffff92e58e43
 #1 [fffffe0000048e58] nmi_handle at ffffffff92e2e168
 #2 [fffffe0000048ea0] default_do_nmi at ffffffff936b4fe2
 #3 [fffffe0000048ec8] exc_nmi at ffffffff936b51ff
 #4 [fffffe0000048ef0] end_repeat_nmi at ffffffff938014db
    [exception RIP: native_safe_halt+14]
    RIP: ffffffff936c3eee  RSP: ffffb5dbc0083ef0  RFLAGS: 00000212
    RAX: ffffffff936c3d90  RBX: 0000000000000001  RCX: ffff9eb75bcb09c0
    RDX: 00000000000dcc1e  RSI: ffffb5dbc0083e88  RDI: 0000018279667269
    RBP: ffff9eb74024df00   R8: 0000000000000001   R9: 00000182493dff0f
    R10: 0000000000000006  R11: 000000000001d400  R12: 0000000000000000
    R13: 0000000000000000  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
--- <NMI exception stack> ---
 #5 [ffffb5dbc0083ef0] native_safe_halt at ffffffff936c3eee
 #6 [ffffb5dbc0083ef0] default_idle at ffffffff936c3d9a
 #7 [ffffb5dbc0083ef8] default_idle_call at ffffffff936c4008
 #8 [ffffb5dbc0083f00] do_idle at ffffffff92ec17a8
 #9 [ffffb5dbc0083f40] cpu_startup_entry at ffffffff92ec19c9
#10 [ffffb5dbc0083f50] secondary_startup_64_no_verify at ffffffff92e000f5

PID: 2172   TASK: ffff9eb669a217c0  CPU: 2   COMMAND: "bash"
 #0 [ffffb5dbc28a3cd0] machine_kexec at ffffffff92e6436b
 #1 [ffffb5dbc28a3d28] __crash_kexec at ffffffff92f3aaad
 #2 [ffffb5dbc28a3df0] panic at ffffffff9367f24d
 #3 [ffffb5dbc28a3e70] sysrq_handle_crash at ffffffff933ca426
 #4 [ffffb5dbc28a3e78] __handle_sysrq.cold at ffffffff936a44c3
 #5 [ffffb5dbc28a3ea8] write_sysrq_trigger at ffffffff933cad34
 #6 [ffffb5dbc28a3eb8] proc_reg_write at ffffffff93164501
 #7 [ffffb5dbc28a3ed0] vfs_write at ffffffff930c1f40
 #8 [ffffb5dbc28a3f08] ksys_write at ffffffff930c23cf
 #9 [ffffb5dbc28a3f40] do_syscall_64 at ffffffff936b3883
```

```
#10 [ffffb5dbc28a3f50] entry_SYSCALL_64_after_hwframe at ffffffff9380008c
    RIP: 00007f4ab1536f33  RSP: 00007ffe545645e8  RFLAGS: 00000246
    RAX: ffffffffffffffda  RBX: 0000000000000002  RCX: 00007f4ab1536f33
    RDX: 0000000000000002  RSI: 0000560032d7a560  RDI: 0000000000000001
    RBP: 0000560032d7a560   R8: 000000000000000a   R9: 0000000000000001
    R10: 0000560032d7b5d0  R11: 0000000000000246  R12: 0000000000000002
    R13: 00007f4ab16076a0  R14: 0000000000000002  R15: 00007f4ab16078a0
    ORIG_RAX: 0000000000000001  CS: 0033  SS: 002b

PID: 0      TASK: ffff9eb74026af80  CPU: 3   COMMAND: "swapper/3"
 #0 [fffffe00000bee50] crash_nmi_callback at ffffffff92e58e43
 #1 [fffffe00000bee58] nmi_handle at ffffffff92e2e168
 #2 [fffffe00000beea0] default_do_nmi at ffffffff936b4fe2
 #3 [fffffe00000beec8] exc_nmi at ffffffff936b51ff
 #4 [fffffe00000beef0] end_repeat_nmi at ffffffff938014db
    [exception RIP: native_safe_halt+14]
    RIP: ffffffff936c3eee  RSP: ffffb5dbc0093ef0  RFLAGS: 00000212
    RAX: ffffffff936c3d90  RBX: 0000000000000003  RCX: ffff9eb75bdb09c0
    RDX: 000000000012d26e  RSI: ffffb5dbc0093e88  RDI: 0000018279667269
    RBP: ffff9eb74026af80   R8: 0000000000000001   R9: 00000182493898ef
    R10: 0000000000000000  R11: 0000000000000000  R12: 0000000000000000
    R13: 0000000000000000  R14: 0000000000000000  R15: 0000000000000000
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
--- <NMI exception stack> ---
 #5 [ffffb5dbc0093ef0] native_safe_halt at ffffffff936c3eee
 #6 [ffffb5dbc0093ef0] default_idle at ffffffff936c3d9a
 #7 [ffffb5dbc0093ef8] default_idle_call at ffffffff936c4008
 #8 [ffffb5dbc0093f00] do_idle at ffffffff92ec17a8
 #9 [ffffb5dbc0093f40] cpu_startup_entry at ffffffff92ec19c9
#10 [ffffb5dbc0093f50] secondary_startup_64_no_verify at ffffffff92e000f5
```

Note: We see **PID 2999** was interrupted on CPU 0. **PID: 2172** stack trace is manual dump generation command. The rest of the PIDs are idle threads.

3. We can check the process tree and CPU consumption:

```
crash> ps -p 2999
PID: 0      TASK: ffffffff94413940  CPU: 0   COMMAND: "swapper/0"
 PID: 2      TASK: ffff9eb7401f0000  CPU: 1   COMMAND: "kthreadd"
  PID: 2999   TASK: ffff9eb7510e17c0  CPU: 0   COMMAND: "mod_c thread"

crash> ps -t 2999
PID: 2999   TASK: ffff9eb7510e17c0  CPU: 0   COMMAND: "mod_c thread"
    RUN TIME: 00:03:28
  START TIME: 1451195628414
       UTIME: 0
       STIME: 198424000000
```

4. Just **ps** command also shows running PIDs:

```
crash> ps
   PID    PPID  CPU       TASK        ST  %MEM     VSZ    RSS  COMM
      0      0   0  ffffffff94413940  RU   0.0       0      0  [swapper/0]
>     0      0   1  ffff9eb74024df00  RU   0.0       0      0  [swapper/1]
      0      0   2  ffff9eb740268000  RU   0.0       0      0  [swapper/2]
>     0      0   3  ffff9eb74026af80  RU   0.0       0      0  [swapper/3]
      1      0   3  ffff9eb7401f5f00  IN   0.2  164292  10396  systemd
      2      0   1  ffff9eb7401f0000  IN   0.0       0      0  [kthreadd]
```

```
    3       2   0  ffff9eb7401f2f80  ID   0.0       0      0  [rcu_gp]
    4       2   0  ffff9eb7401f4740  ID   0.0       0      0  [rcu_par_gp]
    6       2   0  ffff9eb74022df00  ID   0.0       0      0  [kworker/0:0H]
    9       2   0  ffff9eb74022c740  ID   0.0       0      0  [mm_percpu_wq]
   10       2   0  ffff9eb7402297c0  IN   0.0       0      0  [rcu_tasks_rude_]
   11       2   0  ffff9eb740248000  IN   0.0       0      0  [rcu_tasks_trace]
   12       2   0  ffff9eb74024af80  IN   0.0       0      0  [ksoftirqd/0]
   13       2   3  ffff9eb74024c740  ID   0.0       0      0  [rcu_sched]
   14       2   0  ffff9eb7402497c0  RU   0.0       0      0  [migration/0]
   15       2   0  ffff9eb74026c740  IN   0.0       0      0  [cpuhp/0]
   16       2   1  ffff9eb7402697c0  IN   0.0       0      0  [cpuhp/1]
   17       2   1  ffff9eb74026df00  IN   0.0       0      0  [migration/1]
   18       2   1  ffff9eb74028df00  IN   0.0       0      0  [ksoftirqd/1]
   19       2   1  ffff9eb740288000  ID   0.0       0      0  [kworker/1:0]
   20       2   1  ffff9eb74028af80  ID   0.0       0      0  [kworker/1:0H]
   21       2   2  ffff9eb74028c740  IN   0.0       0      0  [cpuhp/2]
   22       2   2  ffff9eb7402897c0  IN   0.0       0      0  [migration/2]
   23       2   2  ffff9eb7402b5f00  IN   0.0       0      0  [ksoftirqd/2]
   24       2   2  ffff9eb7402b0000  ID   0.0       0      0  [kworker/2:0]
   25       2   2  ffff9eb7402b2f80  ID   0.0       0      0  [kworker/2:0H]
   26       2   3  ffff9eb7402b4740  IN   0.0       0      0  [cpuhp/3]
   27       2   3  ffff9eb7402b17c0  IN   0.0       0      0  [migration/3]
   28       2   3  ffff9eb7402e5f00  IN   0.0       0      0  [ksoftirqd/3]
   30       2   3  ffff9eb7402e2f80  ID   0.0       0      0  [kworker/3:0H]
   34       2   3  ffff9eb7403417c0  IN   0.0       0      0  [kdevtmpfs]
   35       2   2  ffff9eb740345f00  ID   0.0       0      0  [netns]
   36       2   3  ffff9eb740340000  IN   0.0       0      0  [kauditd]
   37       2   3  ffff9eb740342f80  IN   0.0       0      0  [khungtaskd]
   38       2   2  ffff9eb75bd597c0  IN   0.0       0      0  [oom_reaper]
   39       2   3  ffff9eb75bd5df00  ID   0.0       0      0  [writeback]
   40       2   3  ffff9eb75bd58000  IN   0.0       0      0  [kcompactd0]
   41       2   3  ffff9eb75bd5af80  IN   0.0       0      0  [ksmd]
   42       2   2  ffff9eb75bd5c740  IN   0.0       0      0  [khugepaged]
   61       2   0  ffff9eb7403a8000  ID   0.0       0      0  [kintegrityd]
   62       2   2  ffff9eb7403aaf80  ID   0.0       0      0  [kblockd]
   63       2   2  ffff9eb740830000  ID   0.0       0      0  [blkcg_punt_bio]
   64       2   3  ffff9eb740832f80  ID   0.0       0      0  [edac-poller]
   65       2   2  ffff9eb740834740  ID   0.0       0      0  [devfreq_wq]
   67       2   0  ffff9eb740835f00  RU   0.0       0      0  [kworker/0:1H]
   70       2   2  ffff9eb75bd7af80  IN   0.0       0      0  [kswapd0]
   71       2   3  ffff9eb75bd7df00  ID   0.0       0      0  [kthrotld]
   72       2   2  ffff9eb75bd797c0  ID   0.0       0      0  [acpi_thermal_pm]
   73       2   3  ffff9eb7584d17c0  ID   0.0       0      0  [ipv6_addrconf]
   82       2   2  ffff9eb7584d4740  ID   0.0       0      0  [kstrp]
   85       2   2  ffff9eb758524740  ID   0.0       0      0  [zswap-shrink]
   86       2   0  ffff9eb7585217c0  ID   0.0       0      0  [kworker/u9:0]
  109       2   3  ffff9eb7585aaf80  ID   0.0       0      0  [kworker/3:1H]
  137       2   2  ffff9eb7584d0000  ID   0.0       0      0  [kworker/2:1H]
  140       2   1  ffff9eb7585a8000  ID   0.0       0      0  [ata_sff]
  141       2   2  ffff9eb7585ac740  IN   0.0       0      0  [scsi_eh_0]
  142       2   1  ffff9eb7586297c0  ID   0.0       0      0  [scsi_tmf_0]
  143       2   1  ffff9eb75862c740  IN   0.0       0      0  [scsi_eh_1]
  144       2   0  ffff9eb75862af80  ID   0.0       0      0  [scsi_tmf_1]
  145       2   1  ffff9eb751ad17c0  IN   0.0       0      0  [scsi_eh_2]
  146       2   1  ffff9eb751ad5f00  ID   0.0       0      0  [scsi_tmf_2]
  148       2   1  ffff9eb751ad2f80  IN   0.0       0      0  [irq/18-vmwgfx]
  149       2   1  ffff9eb751ad4740  ID   0.0       0      0  [ttm_swap]
  150       2   2  ffff9eb751be17c0  IN   0.0       0      0  [card0-crtc0]
  151       2   2  ffff9eb751be5f00  IN   0.0       0      0  [card0-crtc1]
  152       2   2  ffff9eb751be0000  IN   0.0       0      0  [card0-crtc2]
```

```
  153      2   2  ffff9eb751be2f80  IN   0.0       0      0  [card0-crtc3]
  154      2   2  ffff9eb751be4740  IN   0.0       0      0  [card0-crtc4]
  155      2   2  ffff9eb751bfaf80  IN   0.0       0      0  [card0-crtc5]
  156      2   2  ffff9eb751bfc740  IN   0.0       0      0  [card0-crtc6]
  157      2   2  ffff9eb751bf97c0  IN   0.0       0      0  [card0-crtc7]
  165      2   1  ffff9eb75862df00  ID   0.0       0      0  [kworker/1:1H]
  203      2   2  ffff9eb7585cdf00  IN   0.0       0      0  [jbd2/sda1-8]
  204      2   3  ffff9eb7585c97c0  ID   0.0       0      0  [ext4-rsv-conver]
  244      1   2  ffff9eb751478000  IN   0.4   50460  19848  systemd-journal
  264      1   2  ffff9eb7587597c0  IN   0.1   23416   6584  systemd-udevd
  326      2   0  ffff9eb7507397c0  ID   0.0       0      0  [iprt-VBoxWQueue]
  429      1   1  ffff9eb74031df00  IN   0.2  236304   7464  accounts-daemon
  439      1   1  ffff9eb75bdf17c0  IN   0.1    7272   3960  avahi-daemon
  441      1   3  ffff9eb75bdf2f80  IN   0.1    6684   2724  cron
  443      1   2  ffff9eb75bdf0000  IN   0.1    9728   6016  dbus-daemon
  452      1   1  ffff9eb7524a97c0  IN   0.2  236304   7464  gmain
  457      1   2  ffff9eb7524f2f80  IN   0.4  254472  16936  NetworkManager
  470      1   0  ffff9eb74031af80  IN   0.2  235884  10296  polkitd
  471      1   3  ffff9eb7403197c0  IN   0.1  220740   4636  rsyslogd
  478      1   0  ffff9eb74462c740  IN   0.1  232780   6320  switcheroo-cont
  480      1   2  ffff9eb7454e8000  IN   0.2  235884  10296  gmain
  481      1   3  ffff9eb7446297c0  IN   0.2   22088   7388  systemd-logind
  483      1   2  ffff9eb74462df00  IN   0.3  393696  12496  udisksd
  492      1   1  ffff9eb75bdf4740  IN   0.1   14560   6596  wpa_supplicant
  497      1   2  ffff9eb7524aaf80  IN   0.1  220740   4636  in:imuxsock
  498      1   1  ffff9eb7585adf00  IN   0.1  220740   4636  in:imklog
  499      1   2  ffff9eb7510e2f80  IN   0.1  220740   4636  rs:main Q:Reg
  500    439   2  ffff9eb7514797c0  IN   0.0    7092   1348  avahi-daemon
  503      1   0  ffff9eb7454eaf80  IN   0.3  393696  12496  gmain
  505      1   0  ffff9eb7454edf00  IN   0.1  232780   6320  gmain
  520      2   1  ffff9eb7510e5f00  ID   0.0       0      0  [cryptd]
  523      1   0  ffff9eb7510e4740  IN   0.2  235884  10296  gdbus
  524      1   1  ffff9eb751512f80  IN   0.2  236304   7464  gdbus
  525      1   1  ffff9eb7515117c0  IN   0.1  232780   6320  gdbus
  526      1   1  ffff9eb7402e4740  IN   0.3  393696  12496  gdbus
  531      1   3  ffff9eb7524f17c0  IN   0.2  314792  11144  ModemManager
  532      1   1  ffff9eb740318000  IN   0.4  254472  16936  gmain
  536      1   2  ffff9eb7403adf00  IN   0.4  254472  16936  gdbus
  560      1   2  ffff9eb7510e0000  IN   0.3  393696  12496  probing-thread
  577      1   2  ffff9eb75073df00  IN   0.2  314792  11144  gmain
  581      1   2  ffff9eb743f62f80  IN   0.2  314792  11144  gdbus
  590      1   2  ffff9eb743f60000  IN   0.5  118724  25884  unattended-upgr
  618      1   0  ffff9eb74039af80  IN   0.3  393696  12496  cleanup
  656      1   0  ffff9eb758520000  IN   0.5  118724  25884  gmain
  752      1   2  ffff9eb75875c740  IN   0.1  293648   3916  VBoxService
  753      1   2  ffff9eb7524f4740  IN   0.2  239632  10724  gdm3
  754      1   2  ffff9eb7413bc740  IN   0.1  293648   3916  RTThrdPP
  755      1   0  ffff9eb7524f0000  IN   0.1  293648   3916  control
  756      1   1  ffff9eb75bdf5f00  IN   0.1  293648   3916  timesync
  757      1   2  ffff9eb75293c740  IN   0.1  293648   3916  vminfo
  758      1   1  ffff9eb7529397c0  IN   0.1  293648   3916  cpuhotplug
  759      1   2  ffff9eb75293af80  IN   0.1  293648   3916  memballoon
  760      1   1  ffff9eb75293df00  IN   0.1  293648   3916  vmstats
  761      1   3  ffff9eb74462af80  IN   0.1  293648   3916  automount
  763      1   3  ffff9eb744628000  IN   0.2  239632  10724  gmain
  764      1   2  ffff9eb758628000  IN   0.2  239632  10724  gdbus
  799      1   1  ffff9eb743f617c0  IN   0.1  153692   3332  rtkit-daemon
  801      1   2  ffff9eb743f65f00  IN   0.1  153692   3332  rtkit-daemon
  802      1   2  ffff9eb758758000  IN   0.1  153692   3332  rtkit-daemon
  892      1   3  ffff9eb64992af80  IN   0.2  247200   9032  upowerd
```

```
 896      1   2  ffff9eb64992c740  IN   0.2  247200   9032  gmain
 897      1   3  ffff9eb647b24740  IN   0.2  247200   9032  gdbus
 963      1   1  ffff9eb740398000  IN   0.9  364544  44188  packagekitd
 967      1   3  ffff9eb655368000  IN   0.9  364544  44188  gmain
 968      1   2  ffff9eb65536af80  IN   0.9  364544  44188  gdbus
1096      1   1  ffff9eb65dc74740  IN   0.3  242976  13248  colord
1098      1   3  ffff9eb65dc697c0  IN   0.3  242976  13248  gmain
1101      1   3  ffff9eb65a6617c0  IN   0.3  242976  13248  gdbus
1225      2   1  ffff9eb651c72f80  ID   0.0       0      0  [kworker/u8:0]
1249    753   2  ffff9eb6550ac740  IN   0.2  166628  10064  gdm-session-wor
1250    753   0  ffff9eb75824df00  IN   0.2  166628  10064  gmain
1251    753   1  ffff9eb75824af80  IN   0.2  166628  10064  gdbus
1254      1   3  ffff9eb6550adf00  IN   0.2   15744   9396  systemd
1255   1254   3  ffff9eb6550a8000  IN   0.1  167148   4684  (sd-pam)
1266      2   2  ffff9eb65dd797c0  ID   0.0       0      0  [kworker/2:1]
1275   1254   1  ffff9eb65dd7c740  IN   0.1   90576   5736  pipewire
1276   1254   3  ffff9eb751bf8000  IN   0.6 1418256  28824  pulseaudio
1278   1254   0  ffff9eb65a4b0000  IN   0.1   90576   5736  pipewire
1279   1254   3  ffff9eb7403ac740  IN   0.5  509392  25492  tracker-miner-f
1282   1254   3  ffff9eb65a4faf80  IN   0.1    8944   5492  dbus-daemon
1283   1254   1  ffff9eb65a4b2f80  IN   0.5  509392  25492  gmain
1285      1   3  ffff9eb651c717c0  IN   0.2  237216   7484  gnome-keyring-d
1286      1   0  ffff9eb7402e0000  IN   0.2  237216   7484  gmain
1287      1   3  ffff9eb751ad0000  IN   0.2  237216   7484  gdbus
1289   1254   2  ffff9eb65a464740  IN   0.5  509392  25492  gdbus
1293   1254   1  ffff9eb65a462f80  IN   0.5  509392  25492  dconf worker
1302   1254   1  ffff9eb7524a8000  IN   0.2  236896   9568  gvfsd
1303   1254   3  ffff9eb75bd7c740  IN   0.2  236896   9568  gmain
1304   1254   3  ffff9eb751515f00  IN   0.2  236896   9568  gdbus
1308   1254   1  ffff9eb7584d2f80  IN   0.2  379924   8564  gvfsd-fuse
1309   1249   2  ffff9eb65a4b4740  IN   0.1  158836   5724  gdm-wayland-ses
1312   1254   0  ffff9eb751510000  IN   0.2  379924   8564  gvfsd-fuse
1313   1254   1  ffff9eb649982f80  IN   0.2  379924   8564  gvfsd-fuse
1314   1254   0  ffff9eb6499817c0  IN   0.2  379924   8564  gmain
1315   1249   1  ffff9eb7524adf00  IN   0.1  158836   5724  gmain
1316   1249   1  ffff9eb6445c17c0  IN   0.1  158836   5724  gdbus
1317   1254   1  ffff9eb649985f00  IN   0.2  379924   8564  gdbus
1318   1309   1  ffff9eb6445c5f00  IN   0.3  297996  15868  gnome-session-b
1320   1254   2  ffff9eb6445c4740  IN   0.3  496516  13132  gvfs-udisks2-vo
1324   1254   2  ffff9eb65a4617c0  IN   0.2  379924   8564  gvfs-fuse-sub
1325   1275   1  ffff9eb651c82f80  IN   0.1   85300   6476  pipewire-media-
1329   1254   2  ffff9eb64469af80  IN   0.3  496516  13132  gmain
1331   1254   0  ffff9eb655140000  IN   0.3  496516  13132  gdbus
1332   1275   0  ffff9eb745984740  IN   0.1   85300   6476  pipewire-media-
1341   1254   2  ffff9eb647b20000  IN   0.3  496516  13132  dconf worker
1349   1254   0  ffff9eb65a404740  IN   0.1  235108   6952  gvfs-gphoto2-vo
1353   1254   2  ffff9eb655144740  IN   0.1  235108   6952  gmain
1355   1254   1  ffff9eb644772f80  IN   0.1  235108   6952  gdbus
1356   1254   1  ffff9eb745982f80  IN   0.2  311556   7880  gvfs-afc-volume
1357   1254   1  ffff9eb6551417c0  IN   0.2  311556   7880  gvfs-afc-volume
1358   1254   2  ffff9eb65866df00  IN   0.2  311556   7880  gmain
1360   1254   3  ffff9eb6499297c0  IN   0.2  311556   7880  gdbus
1365   1254   0  ffff9eb65a59c740  IN   0.1  233064   6480  gvfs-goa-volume
1372   1254   3  ffff9eb6585b5f00  IN   0.1  233064   6480  gmain
1373   1254   0  ffff9eb65513c740  IN   0.1  233064   6480  gdbus
1376   1254   1  ffff9eb649aa5f00  IN   0.9  550100  40728  goa-daemon
1377   1309   3  ffff9eb649aa17c0  IN   0.3  297996  15868  gmain
1378   1309   0  ffff9eb7413baf80  IN   0.3  297996  15868  gdbus
1379   1309   0  ffff9eb7413b8000  IN   0.3  297996  15868  dconf worker
1380   1254   0  ffff9eb65a5997c0  RU   0.6 1418256  28824  alsa-sink-Intel
```

```
1381   1254   2  ffff9eb7401f17c0  IN   0.1   88176   5260  gnome-session-c
1382   1254   2  ffff9eb65536c740  IN   0.1    5964   4132  ssh-agent
1383   1254   2  ffff9eb65a4017c0  IN   0.1   88176   5260  gmain
1384   1254   1  ffff9eb64989c740  IN   0.4  519724  17156  gnome-session-b
1387   1254   2  ffff9eb7585caf80  IN   0.9  550100  40728  gmain
1389   1254   3  ffff9eb649a9df00  IN   0.9  550100  40728  gdbus
1390   1254   2  ffff9eb7408317c0  IN   0.9  550100  40728  dconf worker
1393   1254   1  ffff9eb6551c0000  IN   0.2  385520   9356  goa-identity-se
1394   1254   0  ffff9eb7585a97c0  IN   0.4  519724  17156  gmain
1395   1254   3  ffff9eb65a5a4740  IN   0.4  519724  17156  gdbus
1397   1254   3  ffff9eb651d25f00  IN   0.2  385520   9356  gmain
1399   1254   1  ffff9eb649a9c740  IN   0.2  385520   9356  gdbus
1403   1254   1  ffff9eb745968000  IN   0.4  519724  17156  dconf worker
1405   1254   0  ffff9eb74022af80  IN   0.2  232872   8344  gvfs-mtp-volume
1407   1254   2  ffff9eb65a665f00  IN   0.2  232872   8344  gmain
1412   1254   1  ffff9eb649a997c0  IN   0.2  232872   8344  gdbus
1414      1   0  ffff9eb651c80000  IN   0.2  237216   7484  timer
1421   1384   2  ffff9eb668e0af80  IN   0.1  307284   6544  at-spi-bus-laun
1423   1254   1  ffff9eb668e60000  IN   5.4 5099268 252452  gnome-shell
1425   1384   3  ffff9eb649a9af80  IN   0.1  307284   6544  gmain
1426   1384   2  ffff9eb65a405f00  IN   0.1  307284   6544  dconf worker
1428   1384   0  ffff9eb668eedf00  IN   0.1  307284   6544  gdbus
1429   1421   2  ffff9eb668ee8000  IN   0.1    8040   4364  dbus-daemon
1430   1254   2  ffff9eb668e65f00  IN   0.5  509392  25492  pool-tracker-mi
1432   1254   2  ffff9eb668e0df00  IN   5.4 5099268 252452  gmain
1434   1254   3  ffff9eb668e08000  IN   5.4 5099268 252452  gdbus
1435   1254   3  ffff9eb668e097c0  IN   5.4 5099268 252452  dconf worker
1443   1254   2  ffff9eb64993c740  IN   0.6 1418256  28824  alsa-source-Int
1444   1254   3  ffff9eb651c74740  IN   5.4 5099268 252452  llvmpipe-0
1445   1254   1  ffff9eb65535c740  IN   5.4 5099268 252452  llvmpipe-1
1446   1254   2  ffff9eb668f95f00  IN   5.4 5099268 252452  llvmpipe-2
1447   1254   3  ffff9eb668f90000  IN   5.4 5099268 252452  llvmpipe-3
1448   1254   3  ffff9eb668f92f80  IN   5.4 5099268 252452  gnome-shell
1449   1254   2  ffff9eb668f94740  IN   5.4 5099268 252452  gnome-shell
1450   1254   1  ffff9eb668f917c0  IN   5.4 5099268 252452  gnome-shell
1451   1254   3  ffff9eb668f9c740  IN   5.4 5099268 252452  gnome-shell
1452   1254   0  ffff9eb668f997c0  IN   5.4 5099268 252452  gnome-s:disk$0
1453   1254   1  ffff9eb668f9df00  IN   5.4 5099268 252452  gnome-s:disk$1
1454   1254   3  ffff9eb668f98000  IN   5.4 5099268 252452  gnome-s:disk$2
1455   1254   3  ffff9eb668f9af80  IN   5.4 5099268 252452  gnome-s:disk$3
1456   1254   3  ffff9eb649930000  IN   5.4 5099268 252452  JS Helper
1457   1254   1  ffff9eb649934740  IN   5.4 5099268 252452  JS Helper
1458   1254   2  ffff9eb65a4a8000  IN   5.4 5099268 252452  JS Helper
1459   1254   1  ffff9eb669a20000  IN   5.4 5099268 252452  JS Helper
1460   1423   1  ffff9eb669a22f80  IN   1.0 1045216  45820  Xwayland
1481   1254   2  ffff9eb66b4cc740  IN   0.1  232788   5952  xdg-permission-
1482   1254   0  ffff9eb66b47df00  IN   0.1  232788   5952  gmain
1484   1254   0  ffff9eb66b4c97c0  IN   0.1  232788   5952  gdbus
1486   1254   2  ffff9eb66b4cdf00  IN   0.5  581412  22908  gnome-shell-cal
1487   1254   2  ffff9eb66b4c5f00  IN   0.5  581412  22908  gmain
1489   1254   0  ffff9eb66b4c8000  IN   0.5  581412  22908  gdbus
1490   1254   1  ffff9eb66b51c740  IN   0.5  581412  22908  dconf worker
1491   1254   1  ffff9eb66b5197c0  IN   0.5  581412  22908  gnome-shell-cal
1492   1254   1  ffff9eb66b5217c0  IN   0.5  392820  25840  evolution-sourc
1493   1254   2  ffff9eb66b51df00  IN   0.5  392820  25840  gmain
1494   1254   1  ffff9eb66b518000  IN   0.5  392820  25840  dconf worker
1495   1254   0  ffff9eb66b525f00  IN   0.5  392820  25840  gdbus
1499   1254   1  ffff9eb66b57df00  IN   0.2  155880   7656  dconf-service
1501   1254   0  ffff9eb66b578000  IN   0.5  581412  22908  pool-gnome-shel
1502   1254   1  ffff9eb66b520000  IN   0.7  849188  35104  evolution-calen
```

```
1503   1254   2  ffff9eb66b522f80  IN   0.2  155880   7656  gmain
1504   1254   1  ffff9eb66b524740  IN   0.2  155880   7656  gdbus
1505   1254   2  ffff9eb66b57af80  IN   0.7  849188  35104  gmain
1506   1254   0  ffff9eb66b5797c0  IN   0.7  849188  35104  gdbus
1511   1254   0  ffff9eb66b65df00  IN   0.7  849188  35104  dconf worker
1512   1254   2  ffff9eb66b658000  IN   0.7  849188  35104  evolution-calen
1513   1254   0  ffff9eb66b65af80  IN   0.7  849188  35104  pool-evolution-
1515   1254   0  ffff9eb66b47c740  IN   0.7  849188  35104  pool-evolution-
1517   1254   0  ffff9eb66b47af80  IN   0.7  849188  35104  pool-evolution-
1518   1254   0  ffff9eb66b6a4740  IN   0.7  849188  35104  evolution-calen
1519   1254   3  ffff9eb644774740  IN   0.7  668124  30948  evolution-addre
1520   1254   1  ffff9eb66b6a17c0  IN   0.7  668124  30948  gmain
1521   1254   0  ffff9eb66b6a5f00  IN   0.7  668124  30948  gdbus
1523   1254   1  ffff9eb66b6a2f80  IN   0.7  668124  30948  dconf worker
1524   1254   0  ffff9eb66b6e8000  IN   0.7  668124  30948  evolution-addre
1527   1254   2  ffff9eb66b65c740  IN   0.2  165668   9228  at-spi2-registr
1528   1254   3  ffff9eb66b6ec740  IN   0.6 2735516  28000  gjs
1529   1254   1  ffff9eb66b7bdf00  IN   0.2  165668   9228  gmain
1530   1254   3  ffff9eb66b7b8000  IN   0.2  165668   9228  gdbus
1533   1254   3  ffff9eb66b7bc740  IN   0.2  306852   8632  gsd-a11y-settin
1535   1254   1  ffff9eb66b7b97c0  IN   0.5  598252  25300  gsd-color
1536   1254   1  ffff9eb66b6597c0  IN   0.3  376132  16132  gsd-datetime
1537   1254   3  ffff9eb670058000  IN   0.2  308860   8008  gsd-housekeepin
1538   1254   3  ffff9eb67005af80  IN   0.5  341904  22120  gsd-keyboard
1539   1254   3  ffff9eb67005c740  IN   0.6  865732  29844  gsd-media-keys
1540   1254   3  ffff9eb6700597c0  IN   0.6  643240  28304  gsd-power
1542   1254   1  ffff9eb67005df00  IN   0.3  320192  12944  gsd-print-notif
1543   1254   1  ffff9eb67013af80  IN   0.1  454268   6536  gsd-rfkill
1544   1254   3  ffff9eb66b7cc740  IN   0.2  306852   8632  gmain
1546   1254   1  ffff9eb66b7c97c0  IN   0.6 2735516  28000  JS Helper
1547   1254   1  ffff9eb67019df00  IN   0.6 2735516  28000  JS Helper
1548   1254   3  ffff9eb670198000  IN   0.6 2735516  28000  JS Helper
1549   1254   0  ffff9eb67019af80  IN   0.6 2735516  28000  JS Helper
1550   1254   3  ffff9eb67019c740  IN   0.2  308860   8008  gmain
1552   1254   0  ffff9eb67013c740  IN   0.1  232700   5984  gsd-screensaver
1553   1254   3  ffff9eb670234740  IN   0.2  306852   8632  gdbus
1554   1254   0  ffff9eb6702317c0  IN   0.2  308860   8008  gdbus
1555   1254   1  ffff9eb6701397c0  IN   0.2  462324  10500  gsd-sharing
1556   1254   3  ffff9eb670235f00  IN   0.2  306852   8632  dconf worker
1559   1254   2  ffff9eb66b6edf00  IN   0.7  668124  30948  pool-evolution-
1560   1254   0  ffff9eb6702f5f00  IN   0.1  454268   6536  gmain
1561   1254   1  ffff9eb6702fc740  IN   0.1  232700   5984  gmain
1563   1254   0  ffff9eb67013df00  IN   0.2  459984  10012  gsd-smartcard
1564   1254   3  ffff9eb6702fdf00  IN   0.3  319496  12488  gsd-sound
1565   1254   3  ffff9eb670232f80  IN   0.2  308860   8008  dconf worker
1566   1254   0  ffff9eb6702f8000  IN   0.2  455828   9152  gsd-usb-protect
1568   1254   2  ffff9eb67180df00  IN   0.5  342328  22448  gsd-wacom
1569   1254   0  ffff9eb66b5caf80  IN   0.1  232700   5984  gdbus
1570   1254   3  ffff9eb671848000  IN   0.2  459984  10012  gmain
1571   1254   1  ffff9eb671808000  IN   0.3  320192  12944  gmain
1572   1254   3  ffff9eb6702f0000  IN   0.1  454268   6536  gdbus
1573   1384   2  ffff9eb66b7caf80  IN   1.7  857432  78400  gnome-software
1577   1254   2  ffff9eb6702f2f80  IN   0.2  462324  10500  gmain
1578   1254   1  ffff9eb67180af80  IN   0.3  320192  12944  gdbus
1579   1254   0  ffff9eb67184c740  IN   0.2  459984  10012  gdbus
1582   1254   1  ffff9eb6702f17c0  IN   0.2  462324  10500  dconf worker
1583   1384   3  ffff9eb6718e4740  IN   1.5  660920  70156  evolution-alarm
1584   1254   0  ffff9eb6718497c0  IN   0.2  455828   9152  gmain
1585   1254   1  ffff9eb67180c740  IN   0.3  376132  16132  gmain
1586   1384   3  ffff9eb6718e17c0  IN   0.1  231792   6820  gsd-disk-utilit
```

```
1588   1254   1  ffff9eb6718097c0  IN   0.3  376132  16132  gdbus
1589   1254   1  ffff9eb671a0af80  IN   0.5  342328  22448  gmain
1591   1254   1  ffff9eb671a117c0  IN   0.3  319496  12488  gmain
1592   1254   2  ffff9eb671a097c0  IN   0.5  342328  22448  dconf worker
1593   1254   0  ffff9eb67184df00  IN   0.2  455828   9152  gdbus
1595   1254   3  ffff9eb671a5af80  IN   0.2  462324  10500  gdbus
1596   1254   1  ffff9eb671a5c740  IN   0.5  342328  22448  gdbus
1598   1254   0  ffff9eb671ab17c0  IN   0.5  598252  25300  gmain
1600   1254   0  ffff9eb671ab5f00  IN   0.6  643240  28304  gmain
1602   1384   3  ffff9eb671a14740  IN   0.1  231792   6820  gmain
1604   1254   1  ffff9eb671ab2f80  IN   0.5  598252  25300  dconf worker
1606   1254   2  ffff9eb671b40000  IN   0.6  643240  28304  dconf worker
1607   1254   2  ffff9eb671a597c0  IN   0.5  341904  22120  gmain
1609   1254   0  ffff9eb671b42f80  IN   0.5  598252  25300  gdbus
1610   1254   2  ffff9eb671b44740  IN   0.6  643240  28304  gdbus
1612   1254   3  ffff9eb671a15f00  IN   0.6 2735516  28000  gmain
1620   1254   0  ffff9eb671b92f80  IN   0.2  455828   9152  dconf worker
1623   1254   1  ffff9eb671b88000  IN   0.3  376132  16132  dconf worker
1625   1254   2  ffff9eb671b95f00  IN   0.2  459984  10012  dconf worker
1628   1254   2  ffff9eb671bdc740  IN   0.5  341904  22120  dconf worker
1629   1254   0  ffff9eb671b90000  IN   0.2  459984  10012  pool-gsd-smartc
1630   1254   0  ffff9eb671afaf80  IN   0.6 2735516  28000  gdbus
1633   1254   1  ffff9eb671b8c740  IN   0.4  344808  17200  gsd-printer
1634   1384   0  ffff9eb671bddf00  IN   0.1  231792   6820  gdbus
1635   1254   0  ffff9eb671af97c0  IN   0.3  319496  12488  gdbus
1636   1254   0  ffff9eb671bd8000  IN   0.5  341904  22120  gdbus
1644   1254   1  ffff9eb66b7baf80  IN   0.0   19888   1248  VBoxClient
1646   1254   3  ffff9eb6760bc740  IN   0.3  319496  12488  dconf worker
1648   1644   0  ffff9eb671a0df00  IN   0.1  152024   4316  VBoxClient
1651   1254   2  ffff9eb671bdaf80  IN   0.6  865732  29844  gmain
1654   1644   0  ffff9eb6760d5f00  IN   0.1  152024   4316  RTThrdPP
1655   1254   0  ffff9eb676165f00  IN   0.6  865732  29844  dconf worker
1657   1254   2  ffff9eb670138000  IN   0.6  865732  29844  gdbus
1662   1254   0  ffff9eb6761e8000  IN   0.0   19888   1208  VBoxClient
1663   1662   2  ffff9eb6761eaf80  IN   0.1  152124   3228  VBoxClient
1668   1644   0  ffff9eb67624df00  IN   0.1  152024   4316  SHCLX11
1672   1254   0  ffff9eb67624af80  IN   0.0   19888   1240  VBoxClient
1673   1672   0  ffff9eb67624c740  IN   0.1  152640   3424  VBoxClient
1678   1254   0  ffff9eb6762d97c0  IN   0.0   19888   1252  VBoxClient
1679   1678   0  ffff9eb6762ddf00  IN   0.1   85904   2488  VBoxDRMClient
1684   1254   2  ffff9eb6761f17c0  IN   0.4  344808  17200  gmain
1685   1254   0  ffff9eb6761f4740  IN   0.4  344808  17200  gdbus
1688   1384   2  ffff9eb6760b97c0  IN   1.7  857432  78400  gmain
1694   1384   2  ffff9eb671af8000  IN   1.7  857432  78400  gdbus
1703   1384   0  ffff9eb6762bc740  IN   1.7  857432  78400  dconf worker
1706   1384   2  ffff9eb6702faf80  IN   1.5  660920  70156  gmain
1708   1384   2  ffff9eb6718e2f80  IN   1.5  660920  70156  dconf worker
1709   1384   1  ffff9eb6760d17c0  IN   1.5  660920  70156  gdbus
1713   1423   0  ffff9eb6762e8000  IN   1.0 1045216  45820  llvmpipe-0
1714   1423   1  ffff9eb6762eaf80  IN   1.0 1045216  45820  llvmpipe-1
1715   1423   3  ffff9eb6762ec740  IN   1.0 1045216  45820  llvmpipe-2
1716   1423   0  ffff9eb6762e97c0  IN   1.0 1045216  45820  llvmpipe-3
1717   1423   1  ffff9eb66b5c8000  IN   1.0 1045216  45820  Xwayland
1718   1423   0  ffff9eb6762daf80  IN   1.0 1045216  45820  Xwayland
1719   1423   0  ffff9eb6762dc740  IN   1.0 1045216  45820  Xwayland
1720   1423   3  ffff9eb6762d8000  IN   1.0 1045216  45820  Xwayland
1721   1423   1  ffff9eb671a10000  IN   1.0 1045216  45820  Xwaylan:disk$0
1722   1423   0  ffff9eb6761f0000  IN   1.0 1045216  45820  Xwaylan:disk$1
1723   1423   0  ffff9eb66b6e97c0  IN   1.0 1045216  45820  Xwaylan:disk$2
1724   1423   3  ffff9eb66b6eaf80  IN   1.0 1045216  45820  Xwaylan:disk$3
```

```
1727   1384   2  ffff9eb6718e0000  IN   1.5  660920  70156  evolution-alarm
1728   1423   1  ffff9eb6718e5f00  IN   0.3  458576  13268  ibus-daemon
1729   1662   2  ffff9eb676160000  IN   0.1  152124   3228  RTThrdPP
1730   1662   2  ffff9eb6760d0000  IN   0.1  152124   3228  X11 events
1731   1254   3  ffff9eb6762b97c0  IN   1.3 1366624  60760  gsd-xsettings
1732   1672   0  ffff9eb676164740  IN   0.1  152640   3424  RTThrdPP
1733   1672   0  ffff9eb676162f80  IN   0.1  152640   3424  dndHGCM
1734   1672   1  ffff9eb671b417c0  IN   0.1  152640   3424  dndX11
1735   1423   0  ffff9eb671a08000  IN   0.3  458576  13268  gmain
1736   1423   1  ffff9eb649a98000  IN   0.3  458576  13268  gdbus
1738      1   3  ffff9eb6760d2f80  IN   0.7  381744  31540  fwupd
1744   1728   1  ffff9eb66b432f80  IN   0.2  233724   7352  ibus-dconf
1745   1728   1  ffff9eb66b4317c0  IN   0.6  346624  27180  ibus-extension-
1751   1728   0  ffff9eb65a59af80  IN   0.2  233724   7352  gmain
1752   1254   1  ffff9eb65a59df00  IN   1.2 1218628  58068  ibus-x11
1754   1728   1  ffff9eb6760bdf00  IN   0.2  233724   7352  gdbus
1756   1254   3  ffff9eb668f497c0  IN   0.2  233576   7168  ibus-portal
1759   1254   0  ffff9eb651c84740  IN   0.2  233576   7168  gmain
1762   1728   1  ffff9eb649b4c740  IN   0.2  233724   7352  dconf worker
1763   1728   2  ffff9eb649b4df00  IN   0.6  346624  27180  gmain
1765   1254   3  ffff9eb669a24740  IN   0.2  233576   7168  gdbus
1770   1728   1  ffff9eb669a25f00  IN   0.6  346624  27180  gdbus
1771   1728   2  ffff9eb649b4af80  IN   0.6  346624  27180  dconf worker
1772      1   1  ffff9eb66b4c17c0  IN   0.7  381744  31540  gmain
1775   1254   0  ffff9eb6762edf00  IN   1.3 1366624  60760  llvmpipe-0
1776   1254   2  ffff9eb66b4caf80  IN   1.3 1366624  60760  llvmpipe-1
1777   1254   0  ffff9eb6553597c0  IN   1.3 1366624  60760  llvmpipe-2
1778   1254   1  ffff9eb655358000  IN   1.3 1366624  60760  llvmpipe-3
1779   1254   2  ffff9eb65535df00  IN   1.3 1366624  60760  gsd-xsettings
1780   1254   0  ffff9eb65535af80  IN   1.3 1366624  60760  gsd-xsettings
1781   1254   3  ffff9eb671b8df00  IN   1.3 1366624  60760  gsd-xsettings
1782   1254   1  ffff9eb6762497c0  IN   1.3 1366624  60760  gsd-xsettings
1783   1254   2  ffff9eb65866af80  IN   1.3 1366624  60760  gsd-xse:disk$0
1784   1254   0  ffff9eb6586697c0  IN   1.3 1366624  60760  gsd-xse:disk$1
1785   1254   3  ffff9eb658668000  IN   1.3 1366624  60760  gsd-xse:disk$2
1786   1254   1  ffff9eb65a5a2f80  IN   1.3 1366624  60760  gsd-xse:disk$3
1788   1254   1  ffff9eb65a5a5f00  IN   1.3 1366624  60760  gmain
1789   1254   0  ffff9eb65a5a17c0  IN   1.3 1366624  60760  gdbus
1790   1728   1  ffff9eb65a465f00  IN   0.2  159900   7204  ibus-engine-sim
1791   1254   0  ffff9eb65a5a0000  IN   1.3 1366624  60760  dconf worker
1792   1728   2  ffff9eb668f4df00  IN   0.2  159900   7204  gmain
1793   1728   1  ffff9eb668f4c740  IN   0.2  159900   7204  gdbus
1796      1   0  ffff9eb7454ec740  IN   0.7  381744  31540  libusb_event
1797      1   3  ffff9eb649ba2f80  IN   0.7  381744  31540  GUsbEventThread
1798   1384   1  ffff9eb649938000  IN   1.5  660920  70156  evolution-alarm
1815   1254   0  ffff9eb6551397c0  IN   1.2 1218628  58068  llvmpipe-0
1816   1254   2  ffff9eb655138000  IN   1.2 1218628  58068  llvmpipe-1
1817   1254   3  ffff9eb65513af80  IN   1.2 1218628  58068  llvmpipe-2
1818   1254   1  ffff9eb6498997c0  IN   1.2 1218628  58068  llvmpipe-3
1819   1254   0  ffff9eb64989af80  IN   1.2 1218628  58068  ibus-x11
1820   1254   3  ffff9eb6551c2f80  IN   1.2 1218628  58068  ibus-x11
1821   1254   0  ffff9eb6551c4740  IN   1.2 1218628  58068  ibus-x11
1822   1254   1  ffff9eb6551c17c0  IN   1.2 1218628  58068  ibus-x11
1824   1254   2  ffff9eb758525f00  IN   1.2 1218628  58068  ibus-x1:disk$0
1825   1254   0  ffff9eb758522f80  IN   1.2 1218628  58068  ibus-x1:disk$1
1826   1254   1  ffff9eb74596af80  IN   1.2 1218628  58068  ibus-x1:disk$2
1827   1254   2  ffff9eb74039c740  IN   1.2 1218628  58068  ibus-x1:disk$3
1829   1254   2  ffff9eb75147af80  IN   1.2 1218628  58068  gmain
1830   1254   0  ffff9eb75147c740  IN   1.2 1218628  58068  gdbus
1831      1   2  ffff9eb649ba4740  IN   0.7  381744  31540  gdbus
```

```
   2124   1384   0  ffff9eb668e0c740  IN   1.7  857432  78400  pool-org.gnome.
   2125   1384   1  ffff9eb644722f80  IN   1.7  857432  78400  pool-org.gnome.
   2126   1384   0  ffff9eb6760d4740  IN   1.7  857432  78400  pool-org.gnome.
   2127   1384   1  ffff9eb647b22f80  IN   1.7  857432  78400  pool-org.gnome.
   2134      2   1  ffff9eb668eec740  ID   0.0       0      0  [kworker/1:1]
   2143   1254   3  ffff9eb65a400000  IN   5.4 5099268 252452  pool-gnome-shel
   2144   1254   0  ffff9eb65a402f80  IN   5.4 5099268 252452  pool-gnome-shel
   2145   1254   1  ffff9eb751514740  IN   5.4 5099268 252452  pool-gnome-shel
   2146   1254   1  ffff9eb644720000  IN   5.4 5099268 252452  pool-gnome-shel
   2156   1254   1  ffff9eb647b25f00  IN   0.9  400984  44224  gnome-terminal-
   2157   1254   2  ffff9eb671b8af80  IN   0.9  400984  44224  gmain
   2159   1254   0  ffff9eb671b897c0  IN   0.9  400984  44224  gdbus
   2160   1254   0  ffff9eb758248000  IN   0.9  400984  44224  dconf worker
   2161   2156   0  ffff9eb75824c740  IN   0.1    8116   4804  bash
   2165   2161   3  ffff9eb6760b8000  IN   0.1   10792   5296  sudo
   2166   2165   0  ffff9eb7582497c0  IN   0.1   10028   4808  su
   2167   2166   1  ffff9eb6760baf80  IN   0.1    8104   4904  bash
   2170   2167   2  ffff9eb65a598000  IN   0.2   16600   8368  mc
>  2172   2170   2  ffff9eb669a217c0  RU   0.1    7100   3840  bash
   2182      2   3  ffff9eb6585b0000  ID   0.0       0      0  [kworker/3:0]
   2184      2   0  ffff9eb6761617c0  RU   0.0       0      0  [kworker/0:2]
   2205   1254   1  ffff9eb65dd7af80  IN   0.1  159328   6152  gvfsd-metadata
   2206   1254   1  ffff9eb6585b2f80  IN   0.1  159328   6152  gmain
   2207   1254   1  ffff9eb6585b4740  IN   0.1  159328   6152  gdbus
   2441      2   0  ffff9eb7413b97c0  RU   0.0       0      0  [kworker/0:1]
   2443      2   3  ffff9eb7413bdf00  ID   0.0       0      0  [kworker/3:2]
   2448      2   2  ffff9eb740228000  RU   0.0       0      0  [kworker/u8:1]
   2597   1254   3  ffff9eb651d20000  IN   0.6  643240  28304  threaded-ml
   2606      2   0  ffff9eb651d22f80  ID   0.0       0      0  [kworker/0:0]
   2624      2   2  ffff9eb6761edf00  ID   0.0       0      0  [kworker/u8:2]
>  2999      2   0  ffff9eb7510e17c0  RU   0.0       0      0  [mod_c thread]
   3004      2   2  ffff9eb649aa0000  ID   0.0       0      0  [kworker/u8:3]
   3008   1254   2  ffff9eb649980000  IN   5.4 5099268 252452  pool-gnome-shel
   3009   1254   1  ffff9eb65dc6af80  IN   5.4 5099268 252452  threaded-ml
```

5. We can also see our problem thread in the log because the watchdog reported it:

```
crash> log -T
[Sat Jun 25 19:23:00 BST 2022] Linux version 5.10.0-10-amd64 (debian-kernel@lists.debian.org) (gcc-10 (Debian 10.2.1-
6) 10.2.1 20210110, GNU ld (GNU Binutils for Debian) 2.35.2) #1 SMP Debian 5.10.84-1 (2021-12-08)
[Sat Jun 25 19:23:00 BST 2022] Command line: BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64 root=UUID=9cc5ee1e-5533-4a0b-
a88f-903bf52d812d ro quiet crashkernel=384M-:128M
[Sat Jun 25 19:23:00 BST 2022] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[Sat Jun 25 19:23:00 BST 2022] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[Sat Jun 25 19:23:00 BST 2022] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[Sat Jun 25 19:23:00 BST 2022] x86/fpu: xstate_offset[2]:  576, xstate_sizes[2]:  256
[Sat Jun 25 19:23:00 BST 2022] x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using 'standard'
format.
[Sat Jun 25 19:23:00 BST 2022] BIOS-provided physical RAM map:
[Sat Jun 25 19:23:00 BST 2022] BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
[Sat Jun 25 19:23:00 BST 2022] BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved
[Sat Jun 25 19:23:00 BST 2022] BIOS-e820: [mem 0x00000000000f0000-0x00000000000fffff] reserved
[Sat Jun 25 19:23:00 BST 2022] BIOS-e820: [mem 0x0000000000100000-0x00000000dffeffff] usable
[Sat Jun 25 19:23:00 BST 2022] BIOS-e820: [mem 0x00000000dfff0000-0x00000000dfffffff] ACPI data
[Sat Jun 25 19:23:00 BST 2022] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec00fff] reserved
[Sat Jun 25 19:23:00 BST 2022] BIOS-e820: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
[Sat Jun 25 19:23:00 BST 2022] BIOS-e820: [mem 0x00000000fffc0000-0x00000000ffffffff] reserved
[Sat Jun 25 19:23:00 BST 2022] BIOS-e820: [mem 0x0000000100000000-0x000000011fffffff] usable
[Sat Jun 25 19:23:00 BST 2022] NX (Execute Disable) protection: active
[Sat Jun 25 19:23:00 BST 2022] SMBIOS 2.5 present.
[Sat Jun 25 19:23:00 BST 2022] DMI: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:23:00 BST 2022] Hypervisor detected: KVM
[Sat Jun 25 19:23:00 BST 2022] kvm-clock: Using msrs 4b564d01 and 4b564d00
[Sat Jun 25 19:23:00 BST 2022] kvm-clock: cpu 0, msr 3dab7001, primary cpu clock
[Sat Jun 25 19:23:00 BST 2022] kvm-clock: using sched offset of 9691889742 cycles
```

```
[Sat Jun 25 19:23:00 BST 2022] clocksource: kvm-clock: mask: 0xffffffffffffffff max_cycles: 0x1cd42e4dffb,
max_idle_ns: 881590591483 ns
[Sat Jun 25 19:23:00 BST 2022] tsc: Detected 1992.006 MHz processor
[Sat Jun 25 19:23:00 BST 2022] e820: update [mem 0x00000000-0x00000fff] usable ==> reserved
[Sat Jun 25 19:23:00 BST 2022] e820: remove [mem 0x000a0000-0x000fffff] usable
[Sat Jun 25 19:23:00 BST 2022] last_pfn = 0x120000 max_arch_pfn = 0x400000000
[Sat Jun 25 19:23:00 BST 2022] MTRR default type: uncachable
[Sat Jun 25 19:23:00 BST 2022] MTRR variable ranges disabled:
[Sat Jun 25 19:23:00 BST 2022] Disabled
[Sat Jun 25 19:23:00 BST 2022] x86/PAT: MTRRs disabled, skipping PAT initialization too.
[Sat Jun 25 19:23:00 BST 2022] CPU MTRRs all blank - virtualized system.
[Sat Jun 25 19:23:00 BST 2022] x86/PAT: Configuration [0-7]: WB  WT  UC- UC  WB  WT  UC- UC
[Sat Jun 25 19:23:00 BST 2022] last_pfn = 0xdfff0 max_arch_pfn = 0x400000000
[Sat Jun 25 19:23:00 BST 2022] found SMP MP-table at [mem 0x0009fff0-0x0009ffff]
[Sat Jun 25 19:23:00 BST 2022] kexec: Reserving the low 1M of memory for crashkernel
[Sat Jun 25 19:23:00 BST 2022] RAMDISK: [mem 0x32ec7000-0x3575afff]
[Sat Jun 25 19:23:00 BST 2022] ACPI: Early table checksum verification disabled
[Sat Jun 25 19:23:00 BST 2022] ACPI: RSDP 0x00000000000E0000 000024 (v02 VBOX  )
[Sat Jun 25 19:23:00 BST 2022] ACPI: XSDT 0x00000000DFFF0030 00003C (v01 VBOX   VBOXXSDT 00000001 ASL  00000061)
[Sat Jun 25 19:23:00 BST 2022] ACPI: FACP 0x00000000DFFF00F0 0000F4 (v04 VBOX   VBOXFACP 00000001 ASL  00000061)
[Sat Jun 25 19:23:00 BST 2022] ACPI: DSDT 0x00000000DFFF0480 002325 (v02 VBOX   VBOXBIOS 00000002 INTL 20190509)
[Sat Jun 25 19:23:00 BST 2022] ACPI: FACS 0x00000000DFFF0200 000040
[Sat Jun 25 19:23:00 BST 2022] ACPI: FACS 0x00000000DFFF0200 000040
[Sat Jun 25 19:23:00 BST 2022] ACPI: APIC 0x00000000DFFF0240 00006C (v02 VBOX   VBOXAPIC 00000001 ASL  00000061)
[Sat Jun 25 19:23:00 BST 2022] ACPI: SSDT 0x00000000DFFF02B0 0001CC (v01 VBOX   VBOXCPUT 00000002 INTL 20190509)
[Sat Jun 25 19:23:00 BST 2022] ACPI: Reserving FACP table memory at [mem 0xdfff00f0-0xdfff01e3]
[Sat Jun 25 19:23:00 BST 2022] ACPI: Reserving DSDT table memory at [mem 0xdfff0480-0xdfff27a4]
[Sat Jun 25 19:23:00 BST 2022] ACPI: Reserving FACS table memory at [mem 0xdfff0200-0xdfff023f]
[Sat Jun 25 19:23:00 BST 2022] ACPI: Reserving FACS table memory at [mem 0xdfff0200-0xdfff023f]
[Sat Jun 25 19:23:00 BST 2022] ACPI: Reserving APIC table memory at [mem 0xdfff0240-0xdfff02ab]
[Sat Jun 25 19:23:00 BST 2022] ACPI: Reserving SSDT table memory at [mem 0xdfff02b0-0xdfff047b]
[Sat Jun 25 19:23:00 BST 2022] ACPI: Local APIC address 0xfee00000
[Sat Jun 25 19:23:00 BST 2022] No NUMA configuration found
[Sat Jun 25 19:23:00 BST 2022] Faking a node at [mem 0x0000000000000000-0x000000011fffffff]
[Sat Jun 25 19:23:00 BST 2022] NODE_DATA(0) allocated [mem 0x11ffd2000-0x11fffbfff]
[Sat Jun 25 19:23:00 BST 2022] Reserving 128MB of memory at 3440MB for crashkernel (System RAM: 4095MB)
[Sat Jun 25 19:23:00 BST 2022] Zone ranges:
[Sat Jun 25 19:23:00 BST 2022]   DMA      [mem 0x0000000000001000-0x0000000000ffffff]
[Sat Jun 25 19:23:00 BST 2022]   DMA32    [mem 0x0000000001000000-0x00000000ffffffff]
[Sat Jun 25 19:23:00 BST 2022]   Normal   [mem 0x0000000100000000-0x000000011fffffff]
[Sat Jun 25 19:23:00 BST 2022]   Device   empty
[Sat Jun 25 19:23:00 BST 2022] Movable zone start for each node
[Sat Jun 25 19:23:00 BST 2022] Early memory node ranges
[Sat Jun 25 19:23:00 BST 2022]   node   0: [mem 0x0000000000001000-0x000000000009efff]
[Sat Jun 25 19:23:00 BST 2022]   node   0: [mem 0x0000000000100000-0x00000000dffeffff]
[Sat Jun 25 19:23:00 BST 2022]   node   0: [mem 0x0000000100000000-0x000000011fffffff]
[Sat Jun 25 19:23:00 BST 2022] Initmem setup node 0 [mem 0x0000000000001000-0x000000011fffffff]
[Sat Jun 25 19:23:00 BST 2022] On node 0 totalpages: 1048462
[Sat Jun 25 19:23:00 BST 2022]   DMA zone: 64 pages used for memmap
[Sat Jun 25 19:23:00 BST 2022]   DMA zone: 158 pages reserved
[Sat Jun 25 19:23:00 BST 2022]   DMA zone: 3998 pages, LIFO batch:0
[Sat Jun 25 19:23:00 BST 2022]   DMA32 zone: 14272 pages used for memmap
[Sat Jun 25 19:23:00 BST 2022]   DMA32 zone: 913392 pages, LIFO batch:63
[Sat Jun 25 19:23:00 BST 2022]   Normal zone: 2048 pages used for memmap
[Sat Jun 25 19:23:00 BST 2022]   Normal zone: 131072 pages, LIFO batch:31
[Sat Jun 25 19:23:00 BST 2022] On node 0, zone DMA: 1 pages in unavailable ranges
[Sat Jun 25 19:23:00 BST 2022] On node 0, zone DMA: 97 pages in unavailable ranges
[Sat Jun 25 19:23:00 BST 2022] On node 0, zone Normal: 16 pages in unavailable ranges
[Sat Jun 25 19:23:00 BST 2022] ACPI: PM-Timer IO Port: 0x4008
[Sat Jun 25 19:23:00 BST 2022] ACPI: Local APIC address 0xfee00000
[Sat Jun 25 19:23:00 BST 2022] IOAPIC[0]: apic_id 4, version 32, address 0xfec00000, GSI 0-23
[Sat Jun 25 19:23:00 BST 2022] ACPI: INT_SRC_OVR (bus 0 bus_irq 0 global_irq 2 dfl dfl)
[Sat Jun 25 19:23:00 BST 2022] ACPI: INT_SRC_OVR (bus 0 bus_irq 9 global_irq 9 low level)
[Sat Jun 25 19:23:00 BST 2022] ACPI: IRQ0 used by override.
[Sat Jun 25 19:23:00 BST 2022] ACPI: IRQ9 used by override.
[Sat Jun 25 19:23:00 BST 2022] Using ACPI (MADT) for SMP configuration information
[Sat Jun 25 19:23:00 BST 2022] smpboot: Allowing 4 CPUs, 0 hotplug CPUs
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0x00000000-0x00000fff]
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0x0009f000-0x0009ffff]
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0x000a0000-0x000effff]
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0x000f0000-0x000fffff]
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xdfff0000-0xdfffffff]
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xe0000000-0xfebfffff]
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xfec00000-0xfec00fff]
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xfec01000-0xfedfffff]
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xfee00000-0xfee00fff]
```

```
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xfee01000-0xfffbffff]
[Sat Jun 25 19:23:00 BST 2022] PM: hibernation: Registered nosave memory: [mem 0xfffc0000-0xffffffff]
[Sat Jun 25 19:23:00 BST 2022] [mem 0xe0000000-0xfebfffff] available for PCI devices
[Sat Jun 25 19:23:00 BST 2022] Booting paravirtualized kernel on KVM
[Sat Jun 25 19:23:00 BST 2022] clocksource: refined-jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns:
7645519600211568 ns
[Sat Jun 25 19:23:00 BST 2022] setup_percpu: NR_CPUS:8192 nr_cpumask_bits:4 nr_cpu_ids:4 nr_node_ids:1
[Sat Jun 25 19:23:00 BST 2022] percpu: Embedded 58 pages/cpu s200536 r8192 d28840 u524288
[Sat Jun 25 19:23:00 BST 2022] pcpu-alloc: s200536 r8192 d28840 u524288 alloc=1*2097152
[Sat Jun 25 19:23:00 BST 2022] pcpu-alloc: [0] 0 1 2 3
[Sat Jun 25 19:23:00 BST 2022] kvm-guest: PV spinlocks enabled
[Sat Jun 25 19:23:00 BST 2022] PV qspinlock hash table entries: 256 (order: 0, 4096 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] Built 1 zonelists, mobility grouping on.  Total pages: 1031920
[Sat Jun 25 19:23:00 BST 2022] Policy zone: Normal
[Sat Jun 25 19:23:00 BST 2022] Kernel command line: BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64 root=UUID=9cc5ee1e-5533-
4a0b-a88f-903bf52d812d ro quiet crashkernel=384M-:128M
[Sat Jun 25 19:23:00 BST 2022] Dentry cache hash table entries: 524288 (order: 10, 4194304 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] Inode-cache hash table entries: 262144 (order: 9, 2097152 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] mem auto-init: stack:off, heap alloc:on, heap free:off
[Sat Jun 25 19:23:00 BST 2022] Memory: 3526712K/4193848K available (12295K kernel code, 2545K rwdata, 7564K rodata,
2408K init, 3684K bss, 346912K reserved, 0K cma-reserved)
[Sat Jun 25 19:23:00 BST 2022] random: get_random_u64 called from __kmem_cache_create+0x2a/0x4d0 with crng_init=0
[Sat Jun 25 19:23:00 BST 2022] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=4, Nodes=1
[Sat Jun 25 19:23:00 BST 2022] Kernel/User page tables isolation: enabled
[Sat Jun 25 19:23:00 BST 2022] ftrace: allocating 36444 entries in 143 pages
[Sat Jun 25 19:23:00 BST 2022] ftrace: allocated 143 pages with 5 groups
[Sat Jun 25 19:23:00 BST 2022] rcu: Hierarchical RCU implementation.
[Sat Jun 25 19:23:00 BST 2022] rcu:     RCU restricting CPUs from NR_CPUS=8192 to nr_cpu_ids=4.
[Sat Jun 25 19:23:00 BST 2022]  Rude variant of Tasks RCU enabled.
[Sat Jun 25 19:23:00 BST 2022]  Tracing variant of Tasks RCU enabled.
[Sat Jun 25 19:23:00 BST 2022] rcu: RCU calculated value of scheduler-enlistment delay is 25 jiffies.
[Sat Jun 25 19:23:00 BST 2022] rcu: Adjusting geometry for rcu_fanout_leaf=16, nr_cpu_ids=4
[Sat Jun 25 19:23:00 BST 2022] NR_IRQS: 524544, nr_irqs: 456, preallocated irqs: 16
[Sat Jun 25 19:23:00 BST 2022] random: crng done (trusting CPU's manufacturer)
[Sat Jun 25 19:23:00 BST 2022] Console: colour VGA+ 80x25
[Sat Jun 25 19:23:00 BST 2022] printk: console [tty0] enabled
[Sat Jun 25 19:23:00 BST 2022] ACPI: Core revision 20200925
[Sat Jun 25 19:23:00 BST 2022] APIC: Switch to symmetric I/O mode setup
[Sat Jun 25 19:23:00 BST 2022] x2apic enabled
[Sat Jun 25 19:23:00 BST 2022] Switched APIC routing to physical x2apic.
[Sat Jun 25 19:23:00 BST 2022] ..TIMER: vector=0x30 apic1=0 pin1=2 apic2=-1 pin2=-1
[Sat Jun 25 19:23:00 BST 2022] clocksource: tsc-early: mask: 0xffffffffffffffff max_cycles: 0x396d5dac02a,
max_idle_ns: 881590811122 ns
[Sat Jun 25 19:23:00 BST 2022] Calibrating delay loop (skipped) preset value.. 3984.01 BogoMIPS (lpj=7968024)
[Sat Jun 25 19:23:00 BST 2022] pid_max: default: 32768 minimum: 301
[Sat Jun 25 19:23:00 BST 2022] LSM: Security Framework initializing
[Sat Jun 25 19:23:00 BST 2022] Yama: disabled by default; enable with sysctl kernel.yama.*
[Sat Jun 25 19:23:00 BST 2022] AppArmor: AppArmor initialized
[Sat Jun 25 19:23:00 BST 2022] TOMOYO Linux initialized
[Sat Jun 25 19:23:00 BST 2022] Mount-cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] Mountpoint-cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] Last level iTLB entries: 4KB 64, 2MB 8, 4MB 8
[Sat Jun 25 19:23:00 BST 2022] Last level dTLB entries: 4KB 64, 2MB 0, 4MB 0, 1GB 4
[Sat Jun 25 19:23:00 BST 2022] Spectre V1 : Mitigation: usercopy/swapgs barriers and __user pointer sanitization
[Sat Jun 25 19:23:00 BST 2022] Spectre V2 : Mitigation: Full generic retpoline
[Sat Jun 25 19:23:00 BST 2022] Spectre V2 : Spectre v2 / SpectreRSB mitigation: Filling RSB on context switch
[Sat Jun 25 19:23:00 BST 2022] Speculative Store Bypass: Vulnerable
[Sat Jun 25 19:23:00 BST 2022] SRBDS: Unknown: Dependent on hypervisor status
[Sat Jun 25 19:23:00 BST 2022] MDS: Mitigation: Clear CPU buffers
[Sat Jun 25 19:23:00 BST 2022] Freeing SMP alternatives memory: 32K
[Sat Jun 25 19:23:00 BST 2022] smpboot: CPU0: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz (family: 0x6, model: 0x8e,
stepping: 0xa)
[Sat Jun 25 19:23:00 BST 2022] Performance Events: unsupported p6 CPU model 142 no PMU driver, software events only.
[Sat Jun 25 19:23:00 BST 2022] rcu: Hierarchical SRCU implementation.
[Sat Jun 25 19:23:00 BST 2022] NMI watchdog: Perf NMI watchdog permanently disabled
[Sat Jun 25 19:23:00 BST 2022] smp: Bringing up secondary CPUs ...
[Sat Jun 25 19:23:00 BST 2022] x86: Booting SMP configuration:
[Sat Jun 25 19:23:00 BST 2022] .... node  #0, CPUs:      #1
[Sat Jun 25 19:23:00 BST 2022] kvm-clock: cpu 1, msr 3dab7041, secondary cpu clock
[Sat Jun 25 19:23:00 BST 2022]  #2
[Sat Jun 25 19:23:00 BST 2022] kvm-clock: cpu 2, msr 3dab7081, secondary cpu clock
[Sat Jun 25 19:23:00 BST 2022]  #3
[Sat Jun 25 19:23:00 BST 2022] kvm-clock: cpu 3, msr 3dab70c1, secondary cpu clock
[Sat Jun 25 19:23:00 BST 2022] smp: Brought up 1 node, 4 CPUs
[Sat Jun 25 19:23:00 BST 2022] smpboot: Max logical packages: 1
[Sat Jun 25 19:23:00 BST 2022] smpboot: Total of 4 processors activated (15936.04 BogoMIPS)
```

```
[Sat Jun 25 19:23:00 BST 2022] node 0 deferred pages initialised in 0ms
[Sat Jun 25 19:23:00 BST 2022] devtmpfs: initialized
[Sat Jun 25 19:23:00 BST 2022] x86/mm: Memory block size: 128MB
[Sat Jun 25 19:23:00 BST 2022] clocksource: jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns:
7645041785100000 ns
[Sat Jun 25 19:23:00 BST 2022] futex hash table entries: 1024 (order: 4, 65536 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] pinctrl core: initialized pinctrl subsystem
[Sat Jun 25 19:23:00 BST 2022] NET: Registered protocol family 16
[Sat Jun 25 19:23:00 BST 2022] audit: initializing netlink subsys (disabled)
[Sat Jun 25 19:23:00 BST 2022] audit: type=2000 audit(1656181391.984:1): state=initialized audit_enabled=0 res=1
[Sat Jun 25 19:23:00 BST 2022] thermal_sys: Registered thermal governor 'fair_share'
[Sat Jun 25 19:23:00 BST 2022] thermal_sys: Registered thermal governor 'bang_bang'
[Sat Jun 25 19:23:00 BST 2022] thermal_sys: Registered thermal governor 'step_wise'
[Sat Jun 25 19:23:00 BST 2022] thermal_sys: Registered thermal governor 'user_space'
[Sat Jun 25 19:23:00 BST 2022] thermal_sys: Registered thermal governor 'power_allocator'
[Sat Jun 25 19:23:00 BST 2022] cpuidle: using governor ladder
[Sat Jun 25 19:23:00 BST 2022] cpuidle: using governor menu
[Sat Jun 25 19:23:00 BST 2022] ACPI: bus type PCI registered
[Sat Jun 25 19:23:00 BST 2022] acpiphp: ACPI Hot Plug PCI Controller Driver version: 0.5
[Sat Jun 25 19:23:00 BST 2022] PCI: Using configuration type 1 for base access
[Sat Jun 25 19:23:00 BST 2022] Kprobes globally optimized
[Sat Jun 25 19:23:00 BST 2022] HugeTLB registered 2.00 MiB page size, pre-allocated 0 pages
[Sat Jun 25 19:23:00 BST 2022] ACPI: Added _OSI(Module Device)
[Sat Jun 25 19:23:00 BST 2022] ACPI: Added _OSI(Processor Device)
[Sat Jun 25 19:23:00 BST 2022] ACPI: Added _OSI(3.0 _SCP Extensions)
[Sat Jun 25 19:23:00 BST 2022] ACPI: Added _OSI(Processor Aggregator Device)
[Sat Jun 25 19:23:00 BST 2022] ACPI: Added _OSI(Linux-Dell-Video)
[Sat Jun 25 19:23:00 BST 2022] ACPI: Added _OSI(Linux-Lenovo-NV-HDMI-Audio)
[Sat Jun 25 19:23:00 BST 2022] ACPI: Added _OSI(Linux-HPI-Hybrid-Graphics)
[Sat Jun 25 19:23:00 BST 2022] ACPI: 2 ACPI AML tables successfully acquired and loaded
[Sat Jun 25 19:23:00 BST 2022] ACPI: Interpreter enabled
[Sat Jun 25 19:23:00 BST 2022] ACPI: (supports S0 S5)
[Sat Jun 25 19:23:00 BST 2022] ACPI: Using IOAPIC for interrupt routing
[Sat Jun 25 19:23:00 BST 2022] PCI: Using host bridge windows from ACPI; if necessary, use "pci=nocrs" and report a
bug
[Sat Jun 25 19:23:00 BST 2022] ACPI: Enabled 2 GPEs in block 00 to 07
[Sat Jun 25 19:23:00 BST 2022] ACPI: PCI Root Bridge [PCI0] (domain 0000 [bus 00-ff])
[Sat Jun 25 19:23:00 BST 2022] acpi PNP0A03:00: _OSC: OS supports [ASPM ClockPM Segments MSI HPX-Type3]
[Sat Jun 25 19:23:00 BST 2022] acpi PNP0A03:00: _OSC: not requesting OS control; OS requires [ExtendedConfig ASPM
ClockPM MSI]
[Sat Jun 25 19:23:00 BST 2022] acpi PNP0A03:00: fail to add MMCONFIG information, can't access extended PCI
configuration space under this bridge.
[Sat Jun 25 19:23:00 BST 2022] PCI host bridge to bus 0000:00
[Sat Jun 25 19:23:00 BST 2022] pci_bus 0000:00: root bus resource [io  0x0000-0x0cf7 window]
[Sat Jun 25 19:23:00 BST 2022] pci_bus 0000:00: root bus resource [io  0x0d00-0xffff window]
[Sat Jun 25 19:23:00 BST 2022] pci_bus 0000:00: root bus resource [mem 0x000a0000-0x000bffff window]
[Sat Jun 25 19:23:00 BST 2022] pci_bus 0000:00: root bus resource [mem 0xe0000000-0xfdffffff window]
[Sat Jun 25 19:23:00 BST 2022] pci_bus 0000:00: root bus resource [bus 00-ff]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:00.0: [8086:1237] type 00 class 0x060000
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:01.0: [8086:7000] type 00 class 0x060100
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:01.1: [8086:7111] type 00 class 0x01018a
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:01.1: reg 0x20: [io  0xd000-0xd00f]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x10: [io  0x01f0-0x01f7]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x14: [io  0x03f6]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x18: [io  0x0170-0x0177]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:01.1: legacy IDE quirk: reg 0x1c: [io  0x0376]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:02.0: [15ad:0405] type 00 class 0x030000
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:02.0: reg 0x10: [io  0xd010-0xd01f]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:02.0: reg 0x14: [mem 0xe0000000-0xe7ffffff pref]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:02.0: reg 0x18: [mem 0xf0000000-0xf01fffff]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:03.0: [8086:100e] type 00 class 0x020000
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:03.0: reg 0x10: [mem 0xf0200000-0xf021ffff]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:03.0: reg 0x18: [io  0xd020-0xd027]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:04.0: [80ee:cafe] type 00 class 0x088000
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:04.0: reg 0x10: [io  0xd040-0xd05f]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:04.0: reg 0x14: [mem 0xf0400000-0xf07fffff]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:04.0: reg 0x18: [mem 0xf0800000-0xf0803fff pref]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:05.0: [8086:2415] type 00 class 0x040100
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:05.0: reg 0x10: [io  0xd100-0xd1ff]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:05.0: reg 0x14: [io  0xd200-0xd23f]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:06.0: [106b:003f] type 00 class 0x0c0310
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:06.0: reg 0x10: [mem 0xf0804000-0xf0804fff]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:07.0: [8086:7113] type 00 class 0x068000
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:07.0: quirk: [io  0x4000-0x403f] claimed by PIIX4 ACPI
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:07.0: quirk: [io  0x4100-0x410f] claimed by PIIX4 SMB
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:0d.0: [8086:2829] type 00 class 0x010601
```

```
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:0d.0: reg 0x10: [io  0xd240-0xd247]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:0d.0: reg 0x14: [io  0xd248-0xd24b]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:0d.0: reg 0x18: [io  0xd250-0xd257]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:0d.0: reg 0x1c: [io  0xd258-0xd25b]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:0d.0: reg 0x20: [io  0xd260-0xd26f]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:0d.0: reg 0x24: [mem 0xf0806000-0xf0807fff]
[Sat Jun 25 19:23:00 BST 2022] ACPI: PCI Interrupt Link [LNKA] (IRQs 5 9 10 *11)
[Sat Jun 25 19:23:00 BST 2022] ACPI: PCI Interrupt Link [LNKB] (IRQs 5 9 *10 11)
[Sat Jun 25 19:23:00 BST 2022] ACPI: PCI Interrupt Link [LNKC] (IRQs 5 *9 10 11)
[Sat Jun 25 19:23:00 BST 2022] ACPI: PCI Interrupt Link [LNKD] (IRQs 5 9 10 *11)
[Sat Jun 25 19:23:00 BST 2022] iommu: Default domain type: Translated
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:02.0: vgaarb: setting as boot VGA device
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:02.0: vgaarb: VGA device added: decodes=io+mem,owns=io+mem,locks=none
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:02.0: vgaarb: bridge control possible
[Sat Jun 25 19:23:00 BST 2022] vgaarb: loaded
[Sat Jun 25 19:23:00 BST 2022] EDAC MC: Ver: 3.0.0
[Sat Jun 25 19:23:00 BST 2022] NetLabel: Initializing
[Sat Jun 25 19:23:00 BST 2022] NetLabel:  domain hash size = 128
[Sat Jun 25 19:23:00 BST 2022] NetLabel:  protocols = UNLABELED CIPSOv4 CALIPSO
[Sat Jun 25 19:23:00 BST 2022] NetLabel:  unlabeled traffic allowed by default
[Sat Jun 25 19:23:00 BST 2022] PCI: Using ACPI for IRQ routing
[Sat Jun 25 19:23:00 BST 2022] PCI: pci_cache_line_size set to 64 bytes
[Sat Jun 25 19:23:00 BST 2022] e820: reserve RAM buffer [mem 0x0009fc00-0x0009ffff]
[Sat Jun 25 19:23:00 BST 2022] e820: reserve RAM buffer [mem 0xdfff0000-0xdfffffff]
[Sat Jun 25 19:23:00 BST 2022] clocksource: Switched to clocksource kvm-clock
[Sat Jun 25 19:23:00 BST 2022] VFS: Disk quotas dquot_6.6.0
[Sat Jun 25 19:23:00 BST 2022] VFS: Dquot-cache hash table entries: 512 (order 0, 4096 bytes)
[Sat Jun 25 19:23:00 BST 2022] AppArmor: AppArmor Filesystem Enabled
[Sat Jun 25 19:23:00 BST 2022] pnp: PnP ACPI init
[Sat Jun 25 19:23:00 BST 2022] pnp 00:00: Plug and Play ACPI device, IDs PNP0303 (active)
[Sat Jun 25 19:23:00 BST 2022] pnp 00:01: Plug and Play ACPI device, IDs PNP0f03 (active)
[Sat Jun 25 19:23:00 BST 2022] pnp: PnP ACPI: found 2 devices
[Sat Jun 25 19:23:00 BST 2022] clocksource: acpi_pm: mask: 0xffffff max_cycles: 0xffffff, max_idle_ns: 2085701024 ns
[Sat Jun 25 19:23:00 BST 2022] NET: Registered protocol family 2
[Sat Jun 25 19:23:00 BST 2022] IP idents hash table entries: 65536 (order: 7, 524288 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] tcp_listen_portaddr_hash hash table entries: 2048 (order: 3, 32768 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] TCP established hash table entries: 32768 (order: 6, 262144 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] TCP bind hash table entries: 32768 (order: 7, 524288 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] TCP: Hash tables configured (established 32768 bind 32768)
[Sat Jun 25 19:23:00 BST 2022] UDP hash table entries: 2048 (order: 4, 65536 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] UDP-Lite hash table entries: 2048 (order: 4, 65536 bytes, linear)
[Sat Jun 25 19:23:00 BST 2022] NET: Registered protocol family 1
[Sat Jun 25 19:23:00 BST 2022] NET: Registered protocol family 44
[Sat Jun 25 19:23:00 BST 2022] pci_bus 0000:00: resource 4 [io  0x0000-0x0cf7 window]
[Sat Jun 25 19:23:00 BST 2022] pci_bus 0000:00: resource 5 [io  0x0d00-0xffff window]
[Sat Jun 25 19:23:00 BST 2022] pci_bus 0000:00: resource 6 [mem 0x000a0000-0x000bffff window]
[Sat Jun 25 19:23:00 BST 2022] pci_bus 0000:00: resource 7 [mem 0xe0000000-0xfdffffff window]
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:00.0: Limiting direct PCI/PCI transfers
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:01.0: Activating ISA DMA hang workarounds
[Sat Jun 25 19:23:00 BST 2022] pci 0000:00:02.0: Video device with shadowed ROM at [mem 0x000c0000-0x000dffff]
[Sat Jun 25 19:23:00 BST 2022] PCI: CLS 0 bytes, default 64
[Sat Jun 25 19:23:00 BST 2022] Trying to unpack rootfs image as initramfs...
[Sat Jun 25 19:23:01 BST 2022] Freeing initrd memory: 41552K
[Sat Jun 25 19:23:01 BST 2022] PCI-DMA: Using software bounce buffering for IO (SWIOTLB)
[Sat Jun 25 19:23:01 BST 2022] software IO TLB: mapped [mem 0x00000000d3000000-0x00000000d7000000] (64MB)
[Sat Jun 25 19:23:01 BST 2022] clocksource: tsc: mask: 0xffffffffffffffff max_cycles: 0x396d5dac02a, max_idle_ns:
881590811122 ns
[Sat Jun 25 19:23:01 BST 2022] clocksource: Switched to clocksource tsc
[Sat Jun 25 19:23:01 BST 2022] platform rtc_cmos: registered platform RTC device (no PNP device found)
[Sat Jun 25 19:23:01 BST 2022] Initialise system trusted keyrings
[Sat Jun 25 19:23:01 BST 2022] Key type blacklist registered
[Sat Jun 25 19:23:01 BST 2022] workingset: timestamp_bits=36 max_order=20 bucket_order=0
[Sat Jun 25 19:23:01 BST 2022] zbud: loaded
[Sat Jun 25 19:23:01 BST 2022] integrity: Platform Keyring initialized
[Sat Jun 25 19:23:01 BST 2022] Key type asymmetric registered
[Sat Jun 25 19:23:01 BST 2022] Asymmetric key parser 'x509' registered
[Sat Jun 25 19:23:01 BST 2022] Block layer SCSI generic (bsg) driver version 0.4 loaded (major 251)
[Sat Jun 25 19:23:01 BST 2022] io scheduler mq-deadline registered
[Sat Jun 25 19:23:01 BST 2022] shpchp: Standard Hot Plug PCI Controller Driver version: 0.4
[Sat Jun 25 19:23:01 BST 2022] intel_idle: Please enable MWAIT in BIOS SETUP
[Sat Jun 25 19:23:01 BST 2022] Serial: 8250/16550 driver, 4 ports, IRQ sharing enabled
[Sat Jun 25 19:23:01 BST 2022] Linux agpgart interface v0.103
[Sat Jun 25 19:23:01 BST 2022] AMD-Vi: AMD IOMMUv2 functionality not available on this system - This is not a bug.
[Sat Jun 25 19:23:01 BST 2022] i8042: PNP: PS/2 Controller [PNP0303:PS2K,PNP0f03:PS2M] at 0x60,0x64 irq 1,12
[Sat Jun 25 19:23:01 BST 2022] serio: i8042 KBD port at 0x60,0x64 irq 1
[Sat Jun 25 19:23:01 BST 2022] serio: i8042 AUX port at 0x60,0x64 irq 12
```

```
[Sat Jun 25 19:23:01 BST 2022] mousedev: PS/2 mouse device common for all mice
[Sat Jun 25 19:23:01 BST 2022] input: AT Translated Set 2 keyboard as /devices/platform/i8042/serio0/input/input0
[Sat Jun 25 19:23:01 BST 2022] rtc_cmos rtc_cmos: registered as rtc0
[Sat Jun 25 19:23:01 BST 2022] rtc_cmos rtc_cmos: setting system clock to 2022-06-25T18:23:01 UTC (1656181381)
[Sat Jun 25 19:23:01 BST 2022] rtc_cmos rtc_cmos: alarms up to one day, 114 bytes nvram
[Sat Jun 25 19:23:01 BST 2022] intel_pstate: CPU model not supported
[Sat Jun 25 19:23:01 BST 2022] ledtrig-cpu: registered to indicate activity on CPUs
[Sat Jun 25 19:23:01 BST 2022] NET: Registered protocol family 10
[Sat Jun 25 19:23:01 BST 2022] Segment Routing with IPv6
[Sat Jun 25 19:23:01 BST 2022] mip6: Mobile IPv6
[Sat Jun 25 19:23:01 BST 2022] NET: Registered protocol family 17
[Sat Jun 25 19:23:01 BST 2022] mpls_gso: MPLS GSO support
[Sat Jun 25 19:23:01 BST 2022] IPI shorthand broadcast: enabled
[Sat Jun 25 19:23:01 BST 2022] sched_clock: Marking stable (1645126907, 13276817)->(1657702798, 700926)
[Sat Jun 25 19:23:01 BST 2022] registered taskstats version 1
[Sat Jun 25 19:23:01 BST 2022] Loading compiled-in X.509 certificates
[Sat Jun 25 19:23:01 BST 2022] Loaded X.509 cert 'Debian Secure Boot CA: 6ccece7e4c6c0d1f6149f3dd27dfcc5cbb419ea1'
[Sat Jun 25 19:23:01 BST 2022] Loaded X.509 cert 'Debian Secure Boot Signer 2021 - linux:
4b6ef5abca669825178e052c84667ccbc0531f8c'
[Sat Jun 25 19:23:01 BST 2022] zswap: loaded using pool lzo/zbud
[Sat Jun 25 19:23:01 BST 2022] Key type ._fscrypt registered
[Sat Jun 25 19:23:01 BST 2022] Key type .fscrypt registered
[Sat Jun 25 19:23:01 BST 2022] Key type fscrypt-provisioning registered
[Sat Jun 25 19:23:01 BST 2022] AppArmor: AppArmor sha1 policy hashing enabled
[Sat Jun 25 19:23:01 BST 2022] Freeing unused kernel image (initmem) memory: 2408K
[Sat Jun 25 19:23:01 BST 2022] Write protecting the kernel read-only data: 22528k
[Sat Jun 25 19:23:01 BST 2022] Freeing unused kernel image (text/rodata gap) memory: 2040K
[Sat Jun 25 19:23:01 BST 2022] Freeing unused kernel image (rodata/data gap) memory: 628K
[Sat Jun 25 19:23:01 BST 2022] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[Sat Jun 25 19:23:01 BST 2022] x86/mm: Checking user space page tables
[Sat Jun 25 19:23:01 BST 2022] x86/mm: Checked W+X mappings: passed, no W+X pages found.
[Sat Jun 25 19:23:01 BST 2022] Run /init as init process
[Sat Jun 25 19:23:01 BST 2022]   with arguments:
[Sat Jun 25 19:23:01 BST 2022]     /init
[Sat Jun 25 19:23:01 BST 2022]   with environment:
[Sat Jun 25 19:23:01 BST 2022]     HOME=/
[Sat Jun 25 19:23:01 BST 2022]     TERM=linux
[Sat Jun 25 19:23:01 BST 2022]     BOOT_IMAGE=/boot/vmlinuz-5.10.0-10-amd64
[Sat Jun 25 19:23:01 BST 2022]     crashkernel=384M-:128M
[Sat Jun 25 19:23:01 BST 2022] input: Power Button as /devices/LNXSYSTM:00/LNXPWRBN:00/input/input2
[Sat Jun 25 19:23:01 BST 2022] ACPI: Video Device [GFX0] (multi-head: yes  rom: no  post: no)
[Sat Jun 25 19:23:01 BST 2022] input: Video Bus as
/devices/LNXSYSTM:00/LNXSYBUS:00/PNP0A03:00/LNXVIDEO:00/input/input3
[Sat Jun 25 19:23:01 BST 2022] battery: ACPI: Battery Slot [BAT0] (battery present)
[Sat Jun 25 19:23:01 BST 2022] piix4_smbus 0000:00:07.0: SMBus Host Controller at 0x4100, revision 0
[Sat Jun 25 19:23:01 BST 2022] ACPI: Power Button [PWRF]
[Sat Jun 25 19:23:01 BST 2022] input: Sleep Button as /devices/LNXSYSTM:00/LNXSLPBN:00/input/input4
[Sat Jun 25 19:23:01 BST 2022] ACPI: Sleep Button [SLPF]
[Sat Jun 25 19:23:01 BST 2022] e1000: Intel(R) PRO/1000 Network Driver
[Sat Jun 25 19:23:01 BST 2022] e1000: Copyright (c) 1999-2006 Intel Corporation.
[Sat Jun 25 19:23:01 BST 2022] SCSI subsystem initialized
[Sat Jun 25 19:23:01 BST 2022] ACPI: bus type USB registered
[Sat Jun 25 19:23:01 BST 2022] usbcore: registered new interface driver usbfs
[Sat Jun 25 19:23:01 BST 2022] usbcore: registered new interface driver hub
[Sat Jun 25 19:23:01 BST 2022] usbcore: registered new device driver usb
[Sat Jun 25 19:23:02 BST 2022] ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
[Sat Jun 25 19:23:02 BST 2022] libata version 3.00 loaded.
[Sat Jun 25 19:23:02 BST 2022] ata_piix 0000:00:01.1: version 2.13
[Sat Jun 25 19:23:02 BST 2022] ahci 0000:00:0d.0: version 3.0
[Sat Jun 25 19:23:02 BST 2022] scsi host0: ata_piix
[Sat Jun 25 19:23:02 BST 2022] ahci 0000:00:0d.0: SSS flag set, parallel bus scan disabled
[Sat Jun 25 19:23:02 BST 2022] ahci 0000:00:0d.0: AHCI 0001.0100 32 slots 1 ports 3 Gbps 0x1 impl SATA mode
[Sat Jun 25 19:23:02 BST 2022] ahci 0000:00:0d.0: flags: 64bit ncq stag only ccc
[Sat Jun 25 19:23:02 BST 2022] ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
[Sat Jun 25 19:23:02 BST 2022] scsi host2: ahci
[Sat Jun 25 19:23:02 BST 2022] ata3: SATA max UDMA/133 abar m8192@0xf0806000 port 0xf0806100 irq 21
[Sat Jun 25 19:23:02 BST 2022] ehci-pci: EHCI PCI platform driver
[Sat Jun 25 19:23:02 BST 2022] ohci-pci: OHCI PCI platform driver
[Sat Jun 25 19:23:02 BST 2022] ohci-pci 0000:00:06.0: OHCI PCI host controller
[Sat Jun 25 19:23:02 BST 2022] ohci-pci 0000:00:06.0: new USB bus registered, assigned bus number 1
[Sat Jun 25 19:23:02 BST 2022] ohci-pci 0000:00:06.0: irq 22, io mem 0xf0804000
[Sat Jun 25 19:23:02 BST 2022] scsi host1: ata_piix
[Sat Jun 25 19:23:02 BST 2022] ata1: PATA max UDMA/33 cmd 0x1f0 ctl 0x3f6 bmdma 0xd000 irq 14
[Sat Jun 25 19:23:02 BST 2022] ata2: PATA max UDMA/33 cmd 0x170 ctl 0x376 bmdma 0xd008 irq 15
[Sat Jun 25 19:23:02 BST 2022] [drm] DMA map mode: Caching DMA mappings.
[Sat Jun 25 19:23:02 BST 2022] [drm] Capabilities:
```

```
[Sat Jun 25 19:23:02 BST 2022] [drm]   Cursor.
[Sat Jun 25 19:23:02 BST 2022] [drm]   Cursor bypass 2.
[Sat Jun 25 19:23:02 BST 2022] [drm]   Alpha cursor.
[Sat Jun 25 19:23:02 BST 2022] [drm]   3D.
[Sat Jun 25 19:23:02 BST 2022] [drm]   Extended Fifo.
[Sat Jun 25 19:23:02 BST 2022] [drm]   Pitchlock.
[Sat Jun 25 19:23:02 BST 2022] [drm]   Irq mask.
[Sat Jun 25 19:23:02 BST 2022] [drm]   GMR.
[Sat Jun 25 19:23:02 BST 2022] [drm]   Traces.
[Sat Jun 25 19:23:02 BST 2022] [drm]   GMR2.
[Sat Jun 25 19:23:02 BST 2022] [drm]   Screen Object 2.
[Sat Jun 25 19:23:02 BST 2022] [drm] Max GMR ids is 8192
[Sat Jun 25 19:23:02 BST 2022] [drm] Max number of GMR pages is 1048576
[Sat Jun 25 19:23:02 BST 2022] [drm] Max dedicated hypervisor surface memory is 393216 kiB
[Sat Jun 25 19:23:02 BST 2022] [drm] Maximum display memory size is 131072 kiB
[Sat Jun 25 19:23:02 BST 2022] [drm] VRAM at 0xe0000000 size is 131072 kiB
[Sat Jun 25 19:23:02 BST 2022] [drm] MMIO at 0xf0000000 size is 2048 kiB
[Sat Jun 25 19:23:02 BST 2022] [TTM] Zone  kernel: Available graphics memory: 1946798 KiB
[Sat Jun 25 19:23:02 BST 2022] [TTM] Initializing pool allocator
[Sat Jun 25 19:23:02 BST 2022] [TTM] Initializing DMA pool allocator
[Sat Jun 25 19:23:02 BST 2022] [drm] Screen Objects Display Unit initialized
[Sat Jun 25 19:23:02 BST 2022] [drm] width 720
[Sat Jun 25 19:23:02 BST 2022] [drm] height 400
[Sat Jun 25 19:23:02 BST 2022] [drm] bpp 32
[Sat Jun 25 19:23:02 BST 2022] [drm] Fifo max 0x00200000 min 0x00001000 cap 0x00000355
[Sat Jun 25 19:23:02 BST 2022] [drm] Atomic: yes.
[Sat Jun 25 19:23:02 BST 2022] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.
[Sat Jun 25 19:23:02 BST 2022] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log message.
[Sat Jun 25 19:23:02 BST 2022] fbcon: svgadrmfb (fb0) is primary device
[Sat Jun 25 19:23:02 BST 2022] Console: switching to colour frame buffer device 100x37
[Sat Jun 25 19:23:02 BST 2022] [drm] Initialized vmwgfx 2.18.0 20200114 for 0000:00:02.0 on minor 0
[Sat Jun 25 19:23:02 BST 2022] usb usb1: New USB device found, idVendor=1d6b, idProduct=0001, bcdDevice= 5.10
[Sat Jun 25 19:23:02 BST 2022] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[Sat Jun 25 19:23:02 BST 2022] usb usb1: Product: OHCI PCI host controller
[Sat Jun 25 19:23:02 BST 2022] usb usb1: Manufacturer: Linux 5.10.0-10-amd64 ohci_hcd
[Sat Jun 25 19:23:02 BST 2022] usb usb1: SerialNumber: 0000:00:06.0
[Sat Jun 25 19:23:02 BST 2022] hub 1-0:1.0: USB hub found
[Sat Jun 25 19:23:02 BST 2022] hub 1-0:1.0: 12 ports detected
[Sat Jun 25 19:23:02 BST 2022] ata2.00: ATAPI: VBOX CD-ROM, 1.0, max UDMA/133
[Sat Jun 25 19:23:02 BST 2022] input: ImExPS/2 Generic Explorer Mouse as /devices/platform/i8042/serio1/input/input5
[Sat Jun 25 19:23:02 BST 2022] ata3: SATA link up 3.0 Gbps (SStatus 123 SControl 300)
[Sat Jun 25 19:23:02 BST 2022] ata3.00: ATA-6: VBOX HARDDISK, 1.0, max UDMA/133
[Sat Jun 25 19:23:02 BST 2022] ata3.00: 209715200 sectors, multi 128: LBA48 NCQ (depth 32)
[Sat Jun 25 19:23:02 BST 2022] ata3.00: configured for UDMA/133
[Sat Jun 25 19:23:02 BST 2022] scsi 2:0:0:0: Direct-Access     ATA      VBOX HARDDISK    1.0  PQ: 0 ANSI: 5
[Sat Jun 25 19:23:02 BST 2022] scsi 1:0:0:0: CD-ROM            VBOX     CD-ROM           1.0  PQ: 0 ANSI: 5
[Sat Jun 25 19:23:02 BST 2022] sd 2:0:0:0: [sda] 209715200 512-byte logical blocks: (107 GB/100 GiB)
[Sat Jun 25 19:23:02 BST 2022] sd 2:0:0:0: [sda] Write Protect is off
[Sat Jun 25 19:23:02 BST 2022] sd 2:0:0:0: [sda] Mode Sense: 00 3a 00 00
[Sat Jun 25 19:23:02 BST 2022] sd 2:0:0:0: [sda] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
[Sat Jun 25 19:23:02 BST 2022] e1000 0000:00:03.0 eth0: (PCI:33MHz:32-bit) 08:00:27:26:5a:6b
[Sat Jun 25 19:23:02 BST 2022] e1000 0000:00:03.0 eth0: Intel(R) PRO/1000 Network Connection
[Sat Jun 25 19:23:02 BST 2022] e1000 0000:00:03.0 enp0s3: renamed from eth0
[Sat Jun 25 19:23:02 BST 2022] usb 1-1: new full-speed USB device number 2 using ohci-pci
[Sat Jun 25 19:23:02 BST 2022] sr 1:0:0:0: [sr0] scsi3-mmc drive: 32x/32x xa/form2 tray
[Sat Jun 25 19:23:02 BST 2022] cdrom: Uniform CD-ROM driver Revision: 3.20
[Sat Jun 25 19:23:02 BST 2022]  sda: sda1 sda2 < sda5 >
[Sat Jun 25 19:23:02 BST 2022] sd 2:0:0:0: [sda] Attached SCSI disk
[Sat Jun 25 19:23:02 BST 2022] sr 1:0:0:0: Attached scsi CD-ROM sr0
[Sat Jun 25 19:23:02 BST 2022] usb 1-1: New USB device found, idVendor=80ee, idProduct=0021, bcdDevice= 1.00
[Sat Jun 25 19:23:02 BST 2022] usb 1-1: New USB device strings: Mfr=1, Product=3, SerialNumber=0
[Sat Jun 25 19:23:02 BST 2022] usb 1-1: Product: USB Tablet
[Sat Jun 25 19:23:02 BST 2022] usb 1-1: Manufacturer: VirtualBox
[Sat Jun 25 19:23:02 BST 2022] hid: raw HID events driver (C) Jiri Kosina
[Sat Jun 25 19:23:02 BST 2022] usbcore: registered new interface driver usbhid
[Sat Jun 25 19:23:02 BST 2022] usbhid: USB HID core driver
[Sat Jun 25 19:23:02 BST 2022] input: VirtualBox USB Tablet as /devices/pci0000:00/0000:00:06.0/usb1/1-1/1-
1:1.0/0003:80EE:0021.0001/input/input6
[Sat Jun 25 19:23:02 BST 2022] hid-generic 0003:80EE:0021.0001: input,hidraw0: USB HID v1.10 Mouse [VirtualBox USB
Tablet] on usb-0000:00:06.0-1/input0
[Sat Jun 25 19:23:02 BST 2022] PM: Image not found (code -22)
[Sat Jun 25 19:23:03 BST 2022] EXT4-fs (sda1): mounted filesystem with ordered data mode. Opts: (null)
[Sat Jun 25 19:23:03 BST 2022] Not activating Mandatory Access Control as /sbin/tomoyo-init does not exist.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Inserted module 'autofs4'
```

```
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: systemd 247.3-6 running in system mode. (+PAM +AUDIT +SELINUX +IMA
+APPARMOR +SMACK +SYSVINIT +UTMP +LIBCRYPTSETUP +GCRYPT +GNUTLS +ACL +XZ +LZ4 +ZSTD +SECCOMP +BLKID +ELFUTILS +KMOD
+IDN2 -IDN +PCRE2 default-hierarchy=unified)
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Detected virtualization oracle.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Detected architecture x86-64.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Set hostname to <coredump>.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: /lib/systemd/system/plymouth-start.service:16: Unit configured to use
KillMode=none. This is unsafe, as it disables systemd's process lifecycle management for the service. Please update
your service to use a safer KillMode=, such as 'mixed' or 'control-group'. Support for KillMode=none is deprecated and
will eventually be removed.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Queued start job for default target Graphical Interface.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Created slice system-getty.slice.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Created slice system-modprobe.slice.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Created slice User and Session Slice.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Started Forward Password Requests to Wall Directory Watch.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Set up automount Arbitrary Executable File Formats File System Automount
Point.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Reached target User and Group Name Lookups.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Reached target Remote File Systems.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Reached target Slices.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Reached target System Time Set.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Reached target System Time Synchronized.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Listening on Syslog Socket.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Listening on fsck to fsckd communication Socket.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Listening on initctl Compatibility Named Pipe.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Listening on Journal Audit Socket.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Listening on Journal Socket (/dev/log).
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Listening on Journal Socket.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Listening on udev Control Socket.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Listening on udev Kernel Socket.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Mounting Huge Pages File System...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Mounting POSIX Message Queue File System...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Mounting Kernel Debug File System...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Mounting Kernel Trace File System...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Starting Set the console keyboard layout...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Starting Create list of static device nodes for the current kernel...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Starting Load Kernel Module configfs...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Starting Load Kernel Module drm...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Starting Load Kernel Module fuse...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Condition check resulted in Set Up Additional Binary Formats being skipped.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Condition check resulted in File System Check on Root Device being skipped.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Starting Journal Service...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Starting Load Kernel Modules...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Starting Remount Root and Kernel File Systems...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Starting Coldplug All udev Devices...
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Mounted Huge Pages File System.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Mounted POSIX Message Queue File System.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Mounted Kernel Debug File System.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Mounted Kernel Trace File System.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Finished Create list of static device nodes for the current kernel.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: modprobe@configfs.service: Succeeded.
[Sat Jun 25 19:23:03 BST 2022] systemd[1]: Finished Load Kernel Module configfs.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: modprobe@drm.service: Succeeded.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Finished Load Kernel Module drm.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Mounting Kernel Configuration File System...
[Sat Jun 25 19:23:04 BST 2022] fuse: init (API version 7.32)
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: modprobe@fuse.service: Succeeded.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Finished Load Kernel Module fuse.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Mounted Kernel Configuration File System.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Mounting FUSE Control File System...
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Finished Load Kernel Modules.
[Sat Jun 25 19:23:04 BST 2022] EXT4-fs (sda1): re-mounted. Opts: errors=remount-ro
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Finished Remount Root and Kernel File Systems.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Condition check resulted in Rebuild Hardware Database being skipped.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Condition check resulted in Platform Persistent Storage Archival being
skipped.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Starting Load/Save Random Seed...
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Starting Apply Kernel Variables...
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Starting Create System Users...
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Mounted FUSE Control File System.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Finished Apply Kernel Variables.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Finished Create System Users.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Starting Create Static Device Nodes in /dev...
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Finished Load/Save Random Seed.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Condition check resulted in First Boot Complete being skipped.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Finished Create Static Device Nodes in /dev.
```

```
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Starting Rule-based Manager for Device Events and Files...
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Finished Coldplug All udev Devices.
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Starting Helper to synchronize boot up for ifupdown...
[Sat Jun 25 19:23:04 BST 2022] systemd[1]: Started Journal Service.
[Sat Jun 25 19:23:04 BST 2022] systemd-journald[244]: Received client request to flush runtime journal.
[Sat Jun 25 19:23:04 BST 2022] systemd-journald[244]: File
/var/log/journal/7a35ae5c9d954e019d1b34858d5e1923/system.journal corrupted or uncleanly shut down, renaming and
replacing.
[Sat Jun 25 19:23:04 BST 2022] audit: type=1400 audit(1656181384.028:2): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-senddoc" pid=280 comm="apparmor_parser"
[Sat Jun 25 19:23:04 BST 2022] audit: type=1400 audit(1656181384.028:3): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="nvidia_modprobe" pid=282 comm="apparmor_parser"
[Sat Jun 25 19:23:04 BST 2022] audit: type=1400 audit(1656181384.028:4): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="nvidia_modprobe//kmod" pid=282 comm="apparmor_parser"
[Sat Jun 25 19:23:04 BST 2022] audit: type=1400 audit(1656181384.028:5): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="/usr/bin/man" pid=279 comm="apparmor_parser"
[Sat Jun 25 19:23:04 BST 2022] audit: type=1400 audit(1656181384.028:6): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="man_filter" pid=279 comm="apparmor_parser"
[Sat Jun 25 19:23:04 BST 2022] audit: type=1400 audit(1656181384.028:7): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="man_groff" pid=279 comm="apparmor_parser"
[Sat Jun 25 19:23:04 BST 2022] audit: type=1400 audit(1656181384.036:8): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="lsb_release" pid=283 comm="apparmor_parser"
[Sat Jun 25 19:23:04 BST 2022] audit: type=1400 audit(1656181384.044:9): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-oopslash" pid=285 comm="apparmor_parser"
[Sat Jun 25 19:23:04 BST 2022] audit: type=1400 audit(1656181384.048:10): apparmor="STATUS" operation="profile_load"
profile="unconfined" name="libreoffice-xpdfimport" pid=287 comm="apparmor_parser"
[Sat Jun 25 19:23:04 BST 2022] vboxguest: loading out-of-tree module taints kernel.
[Sat Jun 25 19:23:04 BST 2022] vboxguest: module verification failed: signature and/or required key missing - tainting
kernel
[Sat Jun 25 19:23:04 BST 2022] ACPI: AC Adapter [AC] (off-line)
[Sat Jun 25 19:23:04 BST 2022] input: PC Speaker as /devices/platform/pcspkr/input/input7
[Sat Jun 25 19:23:04 BST 2022] sd 2:0:0:0: Attached scsi generic sg0 type 0
[Sat Jun 25 19:23:04 BST 2022] sr 1:0:0:0: Attached scsi generic sg1 type 5
[Sat Jun 25 19:23:04 BST 2022] vgdrvHeartbeatInit: Setting up heartbeat to trigger every 2000 milliseconds
[Sat Jun 25 19:23:04 BST 2022] input: Unspecified device as /devices/pci0000:00/0000:00:04.0/input/input8
[Sat Jun 25 19:23:04 BST 2022] vboxguest: Successfully loaded version 6.1.30 r148432
[Sat Jun 25 19:23:04 BST 2022] vboxguest: misc device minor 61, IRQ 20, I/O port d040, MMIO at 00000000f0400000 (size
0x400000)
[Sat Jun 25 19:23:04 BST 2022] vboxguest: Successfully loaded version 6.1.30 r148432 (interface 0x00010004)
[Sat Jun 25 19:23:04 BST 2022] Adding 998396k swap on /dev/sda5.  Priority:-2 extents:1 across:998396k FS
[Sat Jun 25 19:23:04 BST 2022] RAPL PMU: API unit is 2^-32 Joules, 0 fixed counters, 10737418240 ms ovfl timer
[Sat Jun 25 19:23:04 BST 2022] cryptd: max_cpu_qlen set to 1000
[Sat Jun 25 19:23:04 BST 2022] AVX2 version of gcm_enc/dec engaged.
[Sat Jun 25 19:23:04 BST 2022] AES CTR mode by8 optimization enabled
[Sat Jun 25 19:23:04 BST 2022] snd_intel8x0 0000:00:05.0: allow list rate for 1028:0177 is 48000
[Sat Jun 25 19:23:04 BST 2022] intel_pmc_core intel_pmc_core.0:  initialized
[Sat Jun 25 19:23:07 BST 2022] e1000: enp0s3 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[Sat Jun 25 19:23:07 BST 2022] IPv6: ADDRCONF(NETDEV_CHANGE): enp0s3: link becomes ready
[Sat Jun 25 19:23:14 BST 2022] vboxvideo: loading version 6.1.30 r148432
[Sat Jun 25 19:23:14 BST 2022] 18:23:14.504812 main     VBoxService 6.1.30 r148432 (verbosity: 0) linux.amd64 (Nov 22
2021 16:16:32) release log
                               18:23:14.504816 main     Log opened 2022-06-25T18:23:14.504807000Z
[Sat Jun 25 19:23:14 BST 2022] 18:23:14.504922 main     OS Product: Linux
[Sat Jun 25 19:23:14 BST 2022] 18:23:14.504957 main     OS Release: 5.10.0-10-amd64
[Sat Jun 25 19:23:14 BST 2022] 18:23:14.504985 main     OS Version: #1 SMP Debian 5.10.84-1 (2021-12-08)
[Sat Jun 25 19:23:14 BST 2022] 18:23:14.505023 main     Executable: /opt/VBoxGuestAdditions-6.1.30/sbin/VBoxService
                               18:23:14.505024 main     Process ID: 746
                               18:23:14.505025 main     Package type: LINUX_64BITS_GENERIC
[Sat Jun 25 19:23:14 BST 2022] 18:23:14.506767 main     6.1.30 r148432 started. Verbose level = 0
[Sat Jun 25 19:23:14 BST 2022] 18:23:14.508195 main     vbglR3GuestCtrlDetectPeekGetCancelSupport: Supported (#1)
[Sat Jun 25 19:23:14 BST 2022] vboxsf: g_fHostFeatures=0x8000000f g_fSfFeatures=0x1 g_uSfLastFunction=29
[Sat Jun 25 19:23:14 BST 2022] vboxsf: Successfully loaded version 6.1.30 r148432
[Sat Jun 25 19:23:14 BST 2022] vboxsf: Successfully loaded version 6.1.30 r148432 on 5.10.0-10-amd64
(LINUX_VERSION_CODE=0x50a54)
[Sat Jun 25 19:23:14 BST 2022] 18:23:14.527251 automount vbsvcAutomounterMountIt: Successfully mounted 'shared' on
'/media/sf_shared'
[Sat Jun 25 19:23:19 BST 2022] rfkill: input handler disabled
[Sat Jun 25 19:32:53 BST 2022] systemd-journald[244]: File /var/log/journal/7a35ae5c9d954e019d1b34858d5e1923/user-
1000.journal corrupted or uncleanly shut down, renaming and replacing.
[Sat Jun 25 19:32:53 BST 2022] rfkill: input handler enabled
[Sat Jun 25 19:32:56 BST 2022] rfkill: input handler disabled
[Sat Jun 25 19:47:41 BST 2022] rcu: INFO: rcu_sched self-detected stall on CPU
[Sat Jun 25 19:47:41 BST 2022] rcu:     0-....: (5249 ticks this GP) idle=542/1/0x4000000000000000 softirq=14110/14110
fqs=2612
[Sat Jun 25 19:47:41 BST 2022]  (t=5250 jiffies g=29013 q=16141)
[Sat Jun 25 19:47:41 BST 2022] NMI backtrace for cpu 0
```

```
[Sat Jun 25 19:47:41 BST 2022] CPU: 0 PID: 2999 Comm: mod_c thread Kdump: loaded Tainted: G           OE     5.10.0-
10-amd64 #1 Debian 5.10.84-1
[Sat Jun 25 19:47:41 BST 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:47:41 BST 2022] Call Trace:
[Sat Jun 25 19:47:41 BST 2022]  <IRQ>
[Sat Jun 25 19:47:41 BST 2022]  dump_stack+0x6b/0x83
[Sat Jun 25 19:47:41 BST 2022]  nmi_cpu_backtrace.cold+0x32/0x69
[Sat Jun 25 19:47:41 BST 2022]  ? lapic_can_unplug_cpu+0x80/0x80
[Sat Jun 25 19:47:41 BST 2022]  nmi_trigger_cpumask_backtrace+0xd7/0xe0
[Sat Jun 25 19:47:41 BST 2022]  rcu_dump_cpu_stacks+0xa2/0xd0
[Sat Jun 25 19:47:41 BST 2022]  rcu_sched_clock_irq.cold+0x1ff/0x3d6
[Sat Jun 25 19:47:41 BST 2022]  update_process_times+0x8c/0xc0
[Sat Jun 25 19:47:41 BST 2022]  tick_sched_handle+0x22/0x60
[Sat Jun 25 19:47:41 BST 2022]  tick_sched_timer+0x7c/0xb0
[Sat Jun 25 19:47:41 BST 2022]  ? tick_do_update_jiffies64.part.0+0xc0/0xc0
[Sat Jun 25 19:47:41 BST 2022]  __hrtimer_run_queues+0x12a/0x270
[Sat Jun 25 19:47:41 BST 2022]  hrtimer_interrupt+0x110/0x2c0
[Sat Jun 25 19:47:41 BST 2022]  __sysvec_apic_timer_interrupt+0x5f/0xd0
[Sat Jun 25 19:47:41 BST 2022]  asm_call_irq_on_stack+0x12/0x20
[Sat Jun 25 19:47:41 BST 2022]  </IRQ>
[Sat Jun 25 19:47:41 BST 2022]  sysvec_apic_timer_interrupt+0x72/0x80
[Sat Jun 25 19:47:41 BST 2022]  asm_sysvec_apic_timer_interrupt+0x12/0x20
[Sat Jun 25 19:47:41 BST 2022] RIP: 0010:foo+0x5/0xfa0 [mod_c]
[Sat Jun 25 19:47:41 BST 2022] Code: c1 d3 39 d2 48 89 c7 48 3d 00 f0 ff ff 77 08 e8 e1 d0 3a d2 31 c0 c3 c3 66 2e 0f
1f 84 00 00 00 00 00 0f 1f 00 0f 1f 44 00 00 <eb> fe 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[Sat Jun 25 19:47:41 BST 2022] RSP: 0018:ffffb5dbc3c13f08 EFLAGS: 00000246
[Sat Jun 25 19:47:41 BST 2022] RAX: 0000000000000000 RBX: ffffffffc0b10000 RCX: 0000000000000000
[Sat Jun 25 19:47:41 BST 2022] RDX: 0000000000000000 RSI: 0000000000000246 RDI: 0000000000000000
[Sat Jun 25 19:47:41 BST 2022] RBP: ffff9eb751648280 R08: 0000000000000000 R09: 0000000000000000
[Sat Jun 25 19:47:41 BST 2022] R10: 0000000000000000 R11: 0000000000000000 R12: ffff9eb678078fc0
[Sat Jun 25 19:47:41 BST 2022] R13: ffffb5dbc3ce3d28 R14: 0000000000000000 R15: ffff9eb7510e17c0
[Sat Jun 25 19:47:41 BST 2022]  ? 0xffffffffc0b10000
[Sat Jun 25 19:47:41 BST 2022]  ? 0xffffffffc0b10000
[Sat Jun 25 19:47:41 BST 2022]  kthread_f+0x14/0x20 [mod_c]
[Sat Jun 25 19:47:41 BST 2022]  kthread+0x11b/0x140
[Sat Jun 25 19:47:41 BST 2022]  ? __kthread_bind_mask+0x60/0x60
[Sat Jun 25 19:47:41 BST 2022]  ret_from_fork+0x22/0x30
[Sat Jun 25 19:48:07 BST 2022] watchdog: BUG: soft lockup - CPU#0 stuck for 23s! [mod_c thread:2999]
[Sat Jun 25 19:48:07 BST 2022] Modules linked in: mod_c(OE) vboxsf(OE) vboxvideo(OE) rfkill intel_rapl_msr
intel_rapl_common intel_pmc_core_pltdrv intel_pmc_core ghash_clmulni_intel aesni_intel libaes crypto_simd cryptd
glue_helper rapl snd_intel8x0 snd_ac97_codec ac97_bus snd_pcm joydev snd_timer sg snd serio_raw pcspkr ac
vboxguest(OE) soundcore evdev msr fuse configfs ip_tables x_tables autofs4 ext4 crc16 mbcache jbd2 crc32c_generic
hid_generic usbhid hid sd_mod sr_mod cdrom t10_pi crc_t10dif crct10dif_generic ata_generic vmwgfx ttm drm_kms_helper
ohci_pci ehci_pci ohci_hcd ahci libahci ata_piix psmouse cec crct10dif_pclmul crct10dif_common libata ehci_hcd drm
crc32_pclmul usbcore e1000 scsi_mod crc32c_intel i2c_piix4 usb_common battery video button
[Sat Jun 25 19:48:07 BST 2022] CPU: 0 PID: 2999 Comm: mod_c thread Kdump: loaded Tainted: G           OE     5.10.0-
10-amd64 #1 Debian 5.10.84-1
[Sat Jun 25 19:48:07 BST 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:48:07 BST 2022] RIP: 0010:foo+0x5/0xfa0 [mod_c]
[Sat Jun 25 19:48:07 BST 2022] Code: c1 d3 39 d2 48 89 c7 48 3d 00 f0 ff ff 77 08 e8 e1 d0 3a d2 31 c0 c3 c3 66 2e 0f
1f 84 00 00 00 00 00 0f 1f 00 0f 1f 44 00 00 <eb> fe 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[Sat Jun 25 19:48:07 BST 2022] RSP: 0018:ffffb5dbc3c13f08 EFLAGS: 00000246
[Sat Jun 25 19:48:07 BST 2022] RAX: 0000000000000000 RBX: ffffffffc0b10000 RCX: 0000000000000000
[Sat Jun 25 19:48:07 BST 2022] RDX: 0000000000000000 RSI: 0000000000000246 RDI: 0000000000000000
[Sat Jun 25 19:48:07 BST 2022] RBP: ffff9eb751648280 R08: 0000000000000000 R09: 0000000000000000
[Sat Jun 25 19:48:07 BST 2022] R10: 0000000000000000 R11: 0000000000000000 R12: ffff9eb678078fc0
[Sat Jun 25 19:48:07 BST 2022] R13: ffffb5dbc3ce3d28 R14: 0000000000000000 R15: ffff9eb7510e17c0
[Sat Jun 25 19:48:07 BST 2022] FS:  0000000000000000(0000) GS:ffff9eb75bc00000(0000) knlGS:0000000000000000
[Sat Jun 25 19:48:07 BST 2022] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[Sat Jun 25 19:48:07 BST 2022] CR2: 00007fe7b2063ef0 CR3: 000000003d20a005 CR4: 0000000000706f0
[Sat Jun 25 19:48:07 BST 2022] Call Trace:
[Sat Jun 25 19:48:07 BST 2022]  kthread_f+0x14/0x20 [mod_c]
[Sat Jun 25 19:48:07 BST 2022]  kthread+0x11b/0x140
[Sat Jun 25 19:48:07 BST 2022]  ? __kthread_bind_mask+0x60/0x60
[Sat Jun 25 19:48:07 BST 2022]  ret_from_fork+0x22/0x30
[Sat Jun 25 19:48:35 BST 2022] watchdog: BUG: soft lockup - CPU#0 stuck for 23s! [mod_c thread:2999]
[Sat Jun 25 19:48:35 BST 2022] Modules linked in: mod_c(OE) vboxsf(OE) vboxvideo(OE) rfkill intel_rapl_msr
intel_rapl_common intel_pmc_core_pltdrv intel_pmc_core ghash_clmulni_intel aesni_intel libaes crypto_simd cryptd
glue_helper rapl snd_intel8x0 snd_ac97_codec ac97_bus snd_pcm joydev snd_timer sg snd serio_raw pcspkr ac
vboxguest(OE) soundcore evdev msr fuse configfs ip_tables x_tables autofs4 ext4 crc16 mbcache jbd2 crc32c_generic
hid_generic usbhid hid sd_mod sr_mod cdrom t10_pi crc_t10dif crct10dif_generic ata_generic vmwgfx ttm drm_kms_helper
ohci_pci ehci_pci ohci_hcd ahci libahci ata_piix psmouse cec crct10dif_pclmul crct10dif_common libata ehci_hcd drm
crc32_pclmul usbcore e1000 scsi_mod crc32c_intel i2c_piix4 usb_common battery video button
[Sat Jun 25 19:48:35 BST 2022] CPU: 0 PID: 2999 Comm: mod_c thread Kdump: loaded Tainted: G           OEL    5.10.0-
10-amd64 #1 Debian 5.10.84-1
[Sat Jun 25 19:48:35 BST 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
```

```
[Sat Jun 25 19:48:35 BST 2022] RIP: 0010:foo+0x5/0xfa0 [mod_c]
[Sat Jun 25 19:48:35 BST 2022] Code: c1 d3 39 d2 48 89 c7 48 3d 00 f0 ff ff 77 08 e8 e1 d0 3a d2 31 c0 c3 c3 66 2e 0f
1f 84 00 00 00 00 00 0f 1f 00 0f 1f 44 00 00 <eb> fe 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[Sat Jun 25 19:48:35 BST 2022] RSP: 0018:ffffb5dbc3c13f08 EFLAGS: 00000246
[Sat Jun 25 19:48:35 BST 2022] RAX: 0000000000000000 RBX: ffffffffc0b10000 RCX: 0000000000000000
[Sat Jun 25 19:48:35 BST 2022] RDX: 0000000000000000 RSI: 0000000000000246 RDI: 0000000000000000
[Sat Jun 25 19:48:35 BST 2022] RBP: ffff9eb751648280 R08: 0000000000000000 R09: 0000000000000000
[Sat Jun 25 19:48:35 BST 2022] R10: 0000000000000000 R11: 0000000000000000 R12: ffff9eb678078fc0
[Sat Jun 25 19:48:35 BST 2022] R13: ffffb5dbc3ce3d28 R14: 0000000000000000 R15: ffff9eb7510e17c0
[Sat Jun 25 19:48:35 BST 2022] FS:  0000000000000000(0000) GS:ffff9eb75bc00000(0000) knlGS:0000000000000000
[Sat Jun 25 19:48:35 BST 2022] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[Sat Jun 25 19:48:35 BST 2022] CR2: 00007fe7b2063ef0 CR3: 000000003d20a005 CR4: 00000000000706f0
[Sat Jun 25 19:48:35 BST 2022] Call Trace:
[Sat Jun 25 19:48:35 BST 2022]  kthread_f+0x14/0x20 [mod_c]
[Sat Jun 25 19:48:35 BST 2022]  kthread+0x11b/0x140
[Sat Jun 25 19:48:35 BST 2022]  ? __kthread_bind_mask+0x60/0x60
[Sat Jun 25 19:48:35 BST 2022]  ret_from_fork+0x22/0x30
[Sat Jun 25 19:48:44 BST 2022] rcu: INFO: rcu_sched self-detected stall on CPU
[Sat Jun 25 19:48:44 BST 2022] rcu:     0-....: (21002 ticks this GP) idle=542/1/0x4000000000000000
softirq=14110/14110 fqs=10452
[Sat Jun 25 19:48:44 BST 2022]  (t=21003 jiffies g=29013 q=16431)
[Sat Jun 25 19:48:44 BST 2022] NMI backtrace for cpu 0
[Sat Jun 25 19:48:44 BST 2022] CPU: 0 PID: 2999 Comm: mod_c thread Kdump: loaded Tainted: G           OEL    5.10.0-
10-amd64 #1 Debian 5.10.84-1
[Sat Jun 25 19:48:44 BST 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:48:44 BST 2022] Call Trace:
[Sat Jun 25 19:48:44 BST 2022]  <IRQ>
[Sat Jun 25 19:48:44 BST 2022]  dump_stack+0x6b/0x83
[Sat Jun 25 19:48:44 BST 2022]  nmi_cpu_backtrace.cold+0x32/0x69
[Sat Jun 25 19:48:44 BST 2022]  ? lapic_can_unplug_cpu+0x80/0x80
[Sat Jun 25 19:48:44 BST 2022]  nmi_trigger_cpumask_backtrace+0xd7/0xe0
[Sat Jun 25 19:48:44 BST 2022]  rcu_dump_cpu_stacks+0xa2/0xd0
[Sat Jun 25 19:48:44 BST 2022]  rcu_sched_clock_irq.cold+0x1ff/0x3d6
[Sat Jun 25 19:48:44 BST 2022]  update_process_times+0x8c/0xc0
[Sat Jun 25 19:48:44 BST 2022]  tick_sched_handle+0x22/0x60
[Sat Jun 25 19:48:44 BST 2022]  tick_sched_timer+0x7c/0xb0
[Sat Jun 25 19:48:44 BST 2022]  ? tick_do_update_jiffies64.part.0+0xc0/0xc0
[Sat Jun 25 19:48:44 BST 2022]  __hrtimer_run_queues+0x12a/0x270
[Sat Jun 25 19:48:44 BST 2022]  hrtimer_interrupt+0x110/0x2c0
[Sat Jun 25 19:48:44 BST 2022]  __sysvec_apic_timer_interrupt+0x5f/0xd0
[Sat Jun 25 19:48:44 BST 2022]  asm_call_irq_on_stack+0x12/0x20
[Sat Jun 25 19:48:44 BST 2022]  </IRQ>
[Sat Jun 25 19:48:44 BST 2022]  sysvec_apic_timer_interrupt+0x72/0x80
[Sat Jun 25 19:48:44 BST 2022]  asm_sysvec_apic_timer_interrupt+0x12/0x20
[Sat Jun 25 19:48:44 BST 2022] RIP: 0010:foo+0x5/0xfa0 [mod_c]
[Sat Jun 25 19:48:44 BST 2022] Code: c1 d3 39 d2 48 89 c7 48 3d 00 f0 ff ff 77 08 e8 e1 d0 3a d2 31 c0 c3 c3 66 2e 0f
1f 84 00 00 00 00 00 0f 1f 00 0f 1f 44 00 00 <eb> fe 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[Sat Jun 25 19:48:44 BST 2022] RSP: 0018:ffffb5dbc3c13f08 EFLAGS: 00000246
[Sat Jun 25 19:48:44 BST 2022] RAX: 0000000000000000 RBX: ffffffffc0b10000 RCX: 0000000000000000
[Sat Jun 25 19:48:44 BST 2022] RDX: 0000000000000000 RSI: 0000000000000246 RDI: 0000000000000000
[Sat Jun 25 19:48:44 BST 2022] RBP: ffff9eb751648280 R08: 0000000000000000 R09: 0000000000000000
[Sat Jun 25 19:48:44 BST 2022] R10: 0000000000000000 R11: 0000000000000000 R12: ffff9eb678078fc0
[Sat Jun 25 19:48:44 BST 2022] R13: ffffb5dbc3ce3d28 R14: 0000000000000000 R15: ffff9eb7510e17c0
[Sat Jun 25 19:48:44 BST 2022]  ? 0xffffffffc0b10000
[Sat Jun 25 19:48:44 BST 2022]  ? 0xffffffffc0b10000
[Sat Jun 25 19:48:44 BST 2022]  kthread_f+0x14/0x20 [mod_c]
[Sat Jun 25 19:48:44 BST 2022]  kthread+0x11b/0x140
[Sat Jun 25 19:48:44 BST 2022]  ? __kthread_bind_mask+0x60/0x60
[Sat Jun 25 19:48:44 BST 2022]  ret_from_fork+0x22/0x30
[Sat Jun 25 19:49:11 BST 2022] watchdog: BUG: soft lockup - CPU#0 stuck for 22s! [mod_c thread:2999]
[Sat Jun 25 19:49:11 BST 2022] Modules linked in: mod_c(OE) vboxsf(OE) vboxvideo(OE) rfkill intel_rapl_msr
intel_rapl_common intel_pmc_core_pltdrv intel_pmc_core ghash_clmulni_intel aesni_intel libaes crypto_simd cryptd
glue_helper rapl snd_intel8x0 snd_ac97_codec ac97_bus snd_pcm joydev snd_timer sg snd serio_raw pcspkr ac
vboxguest(OE) soundcore evdev msr fuse configfs ip_tables x_tables autofs4 ext4 crc16 mbcache jbd2 crc32c_generic
hid_generic usbhid hid sd_mod sr_mod cdrom t10_pi crc_t10dif crct10dif_generic ata_generic vmwgfx ttm drm_kms_helper
ohci_pci ehci_pci ohci_hcd ahci libahci ata_piix psmouse cec crct10dif_pclmul crct10dif_common libata ehci_hcd drm
crc32_pclmul usbcore e1000 scsi_mod crc32c_intel i2c_piix4 usb_common battery video button
[Sat Jun 25 19:49:11 BST 2022] CPU: 0 PID: 2999 Comm: mod_c thread Kdump: loaded Tainted: G           OEL    5.10.0-
10-amd64 #1 Debian 5.10.84-1
[Sat Jun 25 19:49:11 BST 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:49:11 BST 2022] RIP: 0010:foo+0x5/0xfa0 [mod_c]
[Sat Jun 25 19:49:11 BST 2022] Code: c1 d3 39 d2 48 89 c7 48 3d 00 f0 ff ff 77 08 e8 e1 d0 3a d2 31 c0 c3 c3 66 2e 0f
1f 84 00 00 00 00 00 0f 1f 00 0f 1f 44 00 00 <eb> fe 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[Sat Jun 25 19:49:11 BST 2022] RSP: 0018:ffffb5dbc3c13f08 EFLAGS: 00000246
[Sat Jun 25 19:49:11 BST 2022] RAX: 0000000000000000 RBX: ffffffffc0b10000 RCX: 0000000000000000
[Sat Jun 25 19:49:11 BST 2022] RDX: 0000000000000000 RSI: 0000000000000246 RDI: 0000000000000000
```

```
[Sat Jun 25 19:49:11 BST 2022] RBP: ffff9eb751648280 R08: 0000000000000000 R09: 0000000000000000
[Sat Jun 25 19:49:11 BST 2022] R10: 0000000000000000 R11: 0000000000000000 R12: ffff9eb678078fc0
[Sat Jun 25 19:49:11 BST 2022] R13: ffffb5dbc3ce3d28 R14: 0000000000000000 R15: ffff9eb7510e17c0
[Sat Jun 25 19:49:11 BST 2022] FS:  0000000000000000(0000) GS:ffff9eb75bc00000(0000) knlGS:0000000000000000
[Sat Jun 25 19:49:11 BST 2022] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[Sat Jun 25 19:49:11 BST 2022] CR2: 00007fe7b2063ef0 CR3: 000000003d20a005 CR4: 0000000000706f0
[Sat Jun 25 19:49:11 BST 2022] Call Trace:
[Sat Jun 25 19:49:11 BST 2022]  kthread_f+0x14/0x20 [mod_c]
[Sat Jun 25 19:49:11 BST 2022]  kthread+0x11b/0x140
[Sat Jun 25 19:49:11 BST 2022]  ? __kthread_bind_mask+0x60/0x60
[Sat Jun 25 19:49:11 BST 2022]  ret_from_fork+0x22/0x30
[Sat Jun 25 19:49:39 BST 2022] watchdog: BUG: soft lockup - CPU#0 stuck for 22s! [mod_c thread:2999]
[Sat Jun 25 19:49:39 BST 2022] Modules linked in: mod_c(OE) vboxsf(OE) vboxvideo(OE) rfkill intel_rapl_msr
intel_rapl_common intel_pmc_core_pltdrv intel_pmc_core ghash_clmulni_intel aesni_intel libaes crypto_simd cryptd
glue_helper rapl snd_intel8x0 snd_ac97_codec ac97_bus snd_pcm joydev snd_timer sg snd serio_raw pcspkr ac
vboxguest(OE) soundcore evdev msr fuse configfs ip_tables x_tables autofs4 ext4 crc16 mbcache jbd2 crc32c_generic
hid_generic usbhid hid sd_mod sr_mod cdrom t10_pi crc_t10dif crct10dif_generic ata_generic vmwgfx ttm drm_kms_helper
ohci_pci ehci_pci ohci_hcd ahci libahci ata_piix psmouse cec crct10dif_pclmul crct10dif_common libata ehci_hcd drm
crc32_pclmul usbcore e1000 scsi_mod crc32c_intel i2c_piix4 usb_common battery video button
[Sat Jun 25 19:49:39 BST 2022] CPU: 0 PID: 2999 Comm: mod_c thread Kdump: loaded Tainted: G           OEL    5.10.0-
10-amd64 #1 Debian 5.10.84-1
[Sat Jun 25 19:49:39 BST 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:49:39 BST 2022] RIP: 0010:foo+0x5/0xfa0 [mod_c]
[Sat Jun 25 19:49:39 BST 2022] Code: c1 d3 39 d2 48 89 c7 48 3d 00 f0 ff ff 77 08 e8 e1 d0 3a d2 31 c0 c3 c3 66 2e 0f
1f 84 00 00 00 00 00 0f 1f 00 0f 1f 44 00 00 <eb> fe 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[Sat Jun 25 19:49:39 BST 2022] RSP: 0018:ffffb5dbc3c13f08 EFLAGS: 00000246
[Sat Jun 25 19:49:39 BST 2022] RAX: 0000000000000000 RBX: ffffffffc0b10000 RCX: 0000000000000000
[Sat Jun 25 19:49:39 BST 2022] RDX: 0000000000000000 RSI: 0000000000000246 RDI: 0000000000000000
[Sat Jun 25 19:49:39 BST 2022] RBP: ffff9eb751648280 R08: 0000000000000000 R09: 0000000000000000
[Sat Jun 25 19:49:39 BST 2022] R10: 0000000000000000 R11: 0000000000000000 R12: ffff9eb678078fc0
[Sat Jun 25 19:49:39 BST 2022] R13: ffffb5dbc3ce3d28 R14: 0000000000000000 R15: ffff9eb7510e17c0
[Sat Jun 25 19:49:39 BST 2022] FS:  0000000000000000(0000) GS:ffff9eb75bc00000(0000) knlGS:0000000000000000
[Sat Jun 25 19:49:39 BST 2022] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[Sat Jun 25 19:49:39 BST 2022] CR2: 00007fe7b2063ef0 CR3: 000000003d20a005 CR4: 0000000000706f0
[Sat Jun 25 19:49:39 BST 2022] Call Trace:
[Sat Jun 25 19:49:39 BST 2022]  kthread_f+0x14/0x20 [mod_c]
[Sat Jun 25 19:49:39 BST 2022]  kthread+0x11b/0x140
[Sat Jun 25 19:49:39 BST 2022]  ? __kthread_bind_mask+0x60/0x60
[Sat Jun 25 19:49:39 BST 2022]  ret_from_fork+0x22/0x30
[Sat Jun 25 19:49:47 BST 2022] rcu: INFO: rcu_sched self-detected stall on CPU
[Sat Jun 25 19:49:47 BST 2022] rcu:     0-....: (36754 ticks this GP) idle=542/1/0x4000000000000000
softirq=14110/14110 fqs=18300
[Sat Jun 25 19:49:47 BST 2022]  (t=36756 jiffies g=29013 q=24069)
[Sat Jun 25 19:49:47 BST 2022] NMI backtrace for cpu 0
[Sat Jun 25 19:49:47 BST 2022] CPU: 0 PID: 2999 Comm: mod_c thread Kdump: loaded Tainted: G           OEL    5.10.0-
10-amd64 #1 Debian 5.10.84-1
[Sat Jun 25 19:49:47 BST 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:49:47 BST 2022] Call Trace:
[Sat Jun 25 19:49:47 BST 2022]  <IRQ>
[Sat Jun 25 19:49:47 BST 2022]  dump_stack+0x6b/0x83
[Sat Jun 25 19:49:47 BST 2022]  nmi_cpu_backtrace.cold+0x32/0x69
[Sat Jun 25 19:49:47 BST 2022]  ? lapic_can_unplug_cpu+0x80/0x80
[Sat Jun 25 19:49:47 BST 2022]  nmi_trigger_cpumask_backtrace+0xd7/0xe0
[Sat Jun 25 19:49:47 BST 2022]  rcu_dump_cpu_stacks+0xa2/0xd0
[Sat Jun 25 19:49:47 BST 2022]  rcu_sched_clock_irq.cold+0x1ff/0x3d6
[Sat Jun 25 19:49:47 BST 2022]  update_process_times+0x8c/0xc0
[Sat Jun 25 19:49:47 BST 2022]  tick_sched_handle+0x22/0x60
[Sat Jun 25 19:49:47 BST 2022]  tick_sched_timer+0x7c/0xb0
[Sat Jun 25 19:49:47 BST 2022]  ? tick_do_update_jiffies64.part.0+0xc0/0xc0
[Sat Jun 25 19:49:47 BST 2022]  __hrtimer_run_queues+0x12a/0x270
[Sat Jun 25 19:49:47 BST 2022]  hrtimer_interrupt+0x110/0x2c0
[Sat Jun 25 19:49:47 BST 2022]  __sysvec_apic_timer_interrupt+0x5f/0xd0
[Sat Jun 25 19:49:47 BST 2022]  asm_call_irq_on_stack+0x12/0x20
[Sat Jun 25 19:49:47 BST 2022]  </IRQ>
[Sat Jun 25 19:49:47 BST 2022]  sysvec_apic_timer_interrupt+0x72/0x80
[Sat Jun 25 19:49:47 BST 2022]  asm_sysvec_apic_timer_interrupt+0x12/0x20
[Sat Jun 25 19:49:47 BST 2022] RIP: 0010:foo+0x5/0xfa0 [mod_c]
[Sat Jun 25 19:49:47 BST 2022] Code: c1 d3 39 d2 48 89 c7 48 3d 00 f0 ff ff 77 08 e8 e1 d0 3a d2 31 c0 c3 c3 66 2e 0f
1f 84 00 00 00 00 00 0f 1f 00 0f 1f 44 00 00 <eb> fe 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[Sat Jun 25 19:49:47 BST 2022] RSP: 0018:ffffb5dbc3c13f08 EFLAGS: 00000246
[Sat Jun 25 19:49:47 BST 2022] RAX: 0000000000000000 RBX: ffffffffc0b10000 RCX: 0000000000000000
[Sat Jun 25 19:49:47 BST 2022] RDX: 0000000000000000 RSI: 0000000000000246 RDI: 0000000000000000
[Sat Jun 25 19:49:47 BST 2022] RBP: ffff9eb751648280 R08: 0000000000000000 R09: 0000000000000000
[Sat Jun 25 19:49:47 BST 2022] R10: 0000000000000000 R11: 0000000000000000 R12: ffff9eb678078fc0
[Sat Jun 25 19:49:47 BST 2022] R13: ffffb5dbc3ce3d28 R14: 0000000000000000 R15: ffff9eb7510e17c0
[Sat Jun 25 19:49:47 BST 2022]  ? 0xffffffffc0b10000
```

```
[Sat Jun 25 19:49:47 BST 2022]  ? 0xffffffffc0b10000
[Sat Jun 25 19:49:47 BST 2022]  kthread_f+0x14/0x20 [mod_c]
[Sat Jun 25 19:49:47 BST 2022]  kthread+0x11b/0x140
[Sat Jun 25 19:49:47 BST 2022]  ? __kthread_bind_mask+0x60/0x60
[Sat Jun 25 19:49:47 BST 2022]  ret_from_fork+0x22/0x30
[Sat Jun 25 19:50:15 BST 2022] watchdog: BUG: soft lockup - CPU#0 stuck for 23s! [mod_c thread:2999]
[Sat Jun 25 19:50:15 BST 2022] Modules linked in: mod_c(OE) vboxsf(OE) vboxvideo(OE) rfkill intel_rapl_msr
intel_rapl_common intel_pmc_core_pltdrv intel_pmc_core ghash_clmulni_intel aesni_intel libaes crypto_simd cryptd
glue_helper rapl snd_intel8x0 snd_ac97_codec ac97_bus snd_pcm joydev snd_timer sg snd serio_raw pcspkr ac
vboxguest(OE) soundcore evdev msr fuse configfs ip_tables x_tables autofs4 ext4 crc16 mbcache jbd2 crc32c_generic
hid_generic usbhid hid sd_mod sr_mod cdrom t10_pi crc_t10dif crct10dif_generic ata_generic vmwgfx ttm drm_kms_helper
ohci_pci ehci_pci ohci_hcd ahci libahci ata_piix psmouse cec crct10dif_pclmul crct10dif_common libata ehci_hcd drm
crc32_pclmul usbcore e1000 scsi_mod crc32c_intel i2c_piix4 usb_common battery video button
[Sat Jun 25 19:50:15 BST 2022] CPU: 0 PID: 2999 Comm: mod_c thread Kdump: loaded Tainted: G           OEL    5.10.0-
10-amd64 #1 Debian 5.10.84-1
[Sat Jun 25 19:50:15 BST 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:50:15 BST 2022] RIP: 0010:foo+0x5/0xfa0 [mod_c]
[Sat Jun 25 19:50:15 BST 2022] Code: c1 d3 39 d2 48 89 c7 48 3d 00 f0 ff ff 77 08 e8 e1 d0 3a d2 31 c0 c3 c3 66 2e 0f
1f 84 00 00 00 00 00 0f 1f 00 0f 1f 44 00 00 <eb> fe 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[Sat Jun 25 19:50:15 BST 2022] RSP: 0018:ffffb5dbc3c13f08 EFLAGS: 00000246
[Sat Jun 25 19:50:15 BST 2022] RAX: 0000000000000000 RBX: ffffffffc0b10000 RCX: 0000000000000000
[Sat Jun 25 19:50:15 BST 2022] RDX: 0000000000000000 RSI: 0000000000000246 RDI: 0000000000000000
[Sat Jun 25 19:50:15 BST 2022] RBP: ffff9eb751648280 R08: 0000000000000000 R09: 0000000000000000
[Sat Jun 25 19:50:15 BST 2022] R10: 0000000000000000 R11: 0000000000000000 R12: ffff9eb678078fc0
[Sat Jun 25 19:50:15 BST 2022] R13: ffffb5dbc3ce3d28 R14: 0000000000000000 R15: ffff9eb7510e17c0
[Sat Jun 25 19:50:15 BST 2022] FS:  0000000000000000(0000) GS:ffff9eb75bc00000(0000) knlGS:0000000000000000
[Sat Jun 25 19:50:15 BST 2022] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[Sat Jun 25 19:50:15 BST 2022] CR2: 00007fe7b2063ef0 CR3: 000000003d20a005 CR4: 0000000000706f0
[Sat Jun 25 19:50:15 BST 2022] Call Trace:
[Sat Jun 25 19:50:15 BST 2022]  kthread_f+0x14/0x20 [mod_c]
[Sat Jun 25 19:50:15 BST 2022]  kthread+0x11b/0x140
[Sat Jun 25 19:50:15 BST 2022]  ? __kthread_bind_mask+0x60/0x60
[Sat Jun 25 19:50:15 BST 2022]  ret_from_fork+0x22/0x30
[Sat Jun 25 19:50:39 BST 2022] sysrq: Trigger a crash
[Sat Jun 25 19:50:39 BST 2022] Kernel panic - not syncing: sysrq triggered crash
[Sat Jun 25 19:50:39 BST 2022] CPU: 2 PID: 2172 Comm: bash Kdump: loaded Tainted: G           OEL    5.10.0-10-amd64
#1 Debian 5.10.84-1
[Sat Jun 25 19:50:39 BST 2022] Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[Sat Jun 25 19:50:39 BST 2022] Call Trace:
[Sat Jun 25 19:50:39 BST 2022]  dump_stack+0x6b/0x83
[Sat Jun 25 19:50:39 BST 2022]  panic+0x101/0x2d7
[Sat Jun 25 19:50:39 BST 2022]  ? printk+0x58/0x6f
[Sat Jun 25 19:50:39 BST 2022]  sysrq_handle_crash+0x16/0x20
[Sat Jun 25 19:50:39 BST 2022]  __handle_sysrq.cold+0x43/0x113
[Sat Jun 25 19:50:39 BST 2022]  write_sysrq_trigger+0x24/0x40
[Sat Jun 25 19:50:39 BST 2022]  proc_reg_write+0x51/0x90
[Sat Jun 25 19:50:39 BST 2022]  vfs_write+0xc0/0x260
[Sat Jun 25 19:50:39 BST 2022]  ksys_write+0x5f/0xe0
[Sat Jun 25 19:50:39 BST 2022]  do_syscall_64+0x33/0x80
[Sat Jun 25 19:50:39 BST 2022]  entry_SYSCALL_64_after_hwframe+0x44/0xa9
[Sat Jun 25 19:50:39 BST 2022] RIP: 0033:0x7f4ab1536f33
[Sat Jun 25 19:50:39 BST 2022] Code: 8b 15 61 ef 0c 00 f7 d8 64 89 02 48 c7 c0 ff ff ff ff eb b7 0f 1f 00 64 8b 04 25
18 00 00 00 85 c0 75 14 b8 01 00 00 00 0f 05 <48> 3d 00 f0 ff ff 77 55 c3 0f 1f 40 00 48 83 ec 28 48 89 54 24 18
[Sat Jun 25 19:50:39 BST 2022] RSP: 002b:00007ffe545645e8 EFLAGS: 00000246 ORIG_RAX: 0000000000000001
[Sat Jun 25 19:50:39 BST 2022] RAX: ffffffffffffffda RBX: 0000000000000002 RCX: 00007f4ab1536f33
[Sat Jun 25 19:50:39 BST 2022] RDX: 0000000000000002 RSI: 0000560032d7a560 RDI: 0000000000000001
[Sat Jun 25 19:50:39 BST 2022] RBP: 0000560032d7a560 R08: 000000000000000a R09: 0000000000000001
[Sat Jun 25 19:50:39 BST 2022] R10: 0000560032d7b5d0 R11: 0000000000000246 R12: 0000000000000002
[Sat Jun 25 19:50:39 BST 2022] R13: 00007f4ab16076a0 R14: 0000000000000002 R15: 00007f4ab16078a0
```

6. If we look at the interrupted **foo+0x5** location (also shown as called from **kthread_f** with **kthread_f+0x14** return address), we see it cycles indefinitely:

```
crash> dis foo+0x5
0xffffffffc0b10065 <foo+5>:     jmp    0xffffffffc0b10065 <foo+5>
```

```
crash> dis kthread_f
0xffffffffc0b10000 <kthread_f>: nopl   0x0(%rax,%rax,1) [FTRACE NOP]
0xffffffffc0b10005 <kthread_f+5>:       mov    $0x2710,%edi
0xffffffffc0b1000a <kthread_f+10>:      call   0xffffffff92f17cf0 <msleep>
0xffffffffc0b1000f <kthread_f+15>:      call   0xffffffffc0b10060
0xffffffffc0b10014 <kthread_f+20>:      xor    %eax,%eax
0xffffffffc0b10016 <kthread_f+22>:      ret
0xffffffffc0b10017 <kthread_f+23>:      nopw   0x0(%rax,%rax,1)

crash> dis 0xffffffffc0b10060 2
0xffffffffc0b10060 <foo>:       nopl   0x0(%rax,%rax,1) [FTRACE NOP]
0xffffffffc0b10065 <foo+5>:     jmp    0xffffffffc0b10065 <foo+5>
```

Exercise K5

- **Goal:** Learn how to identify kernel stack overflow and kernel stack boundaries
- **Patterns:** Stack Overflow (Kernel Mode)
- \ALCDA-Dumps\Exercise-K5-x64-GDB.pdf

Exercise K5 (x64, GDB)

Goal: Learn how to identify kernel stack overflow and kernel stack boundaries.

Patterns: Stack Overflow (Kernel Mode).

1. Load a core dump *dump.202206252109* from the x64/K5 directory and the matching *vmlinux-5.10.0-10-amd64* file from the x64/KSym directory:

```
~/ALCDA2/x64/K5$ crash dump.202206252109 ../KSym/vmlinux-5.10.0-10-amd64

crash 8.0.0++
Copyright (C) 2002-2021  Red Hat, Inc.
Copyright (C) 2004, 2005, 2006, 2010  IBM Corporation
Copyright (C) 1999-2006  Hewlett-Packard Co
Copyright (C) 2005, 2006, 2011, 2012  Fujitsu Limited
Copyright (C) 2006, 2007  VA Linux Systems Japan K.K.
Copyright (C) 2005, 2011, 2020-2021  NEC Corporation
Copyright (C) 1999, 2002, 2007  Silicon Graphics, Inc.
Copyright (C) 1999, 2000, 2001, 2002  Mission Critical Linux, Inc.
Copyright (C) 2015, 2021  VMware, Inc.
This program is free software, covered by the GNU General Public License,
and you are welcome to change it and/or distribute copies of it under
certain conditions.  Enter "help copying" to see the conditions.
This program has absolutely no warranty.  Enter "help warranty" for details.

GNU gdb (GDB) 10.2
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-pc-linux-gnu".
Type "show configuration" for configuration details.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...

      KERNEL: ../KSym/vmlinux-5.10.0-10-amd64  [TAINTED]
    DUMPFILE: dump.202206252109  [PARTIAL DUMP]
        CPUS: 4
        DATE: Sat Jun 25 21:09:26 BST 2022
      UPTIME: 00:06:30
LOAD AVERAGE: 0.32, 0.22, 0.11
       TASKS: 455
    NODENAME: coredump
     RELEASE: 5.10.0-10-amd64
     VERSION: #1 SMP Debian 5.10.84-1 (2021-12-08)
     MACHINE: x86_64  (1992 Mhz)
      MEMORY: 4 GB
       PANIC: ""
         PID: 3831
     COMMAND: "mod_d thread"
        TASK: ffff9845f766af80  [THREAD_INFO: ffff9845f766af80]
         CPU: 3
```

```
       STATE: TASK_RUNNING (PANIC)

crash>
```

2. The panic description is empty, but the backtrace shows recursive calls and double fault pointing to stack overflow:

```
crash> bt
PID: 3831   TASK: ffff9845f766af80  CPU: 3   COMMAND: "mod_d thread"
#0 [fffffe00000bbdc0] machine_kexec at ffffffff9986436b
 #1 [fffffe00000bbe18] __crash_kexec at ffffffff9993aaad
 #2 [fffffe00000bbee0] crash_kexec at ffffffff9993bbe5
 #3 [fffffe00000bbef0] oops_end at ffffffff9982da9b
 #4 [fffffe00000bbf10] handle_stack_overflow at ffffffff9a079633
 #5 [fffffe00000bbf28] exc_double_fault at ffffffff9a0b3ffe
 #6 [fffffe00000bbf50] asm_exc_double_fault at ffffffff9a200bce
    [exception RIP: foo+6]
    RIP: ffffffffc0676066  RSP: ffffbc76000b8000  RFLAGS: 00010246
    RAX: 0000000000000000  RBX: 0000000000000000  RCX: 0000000000000000
    RDX: 0000000000000000  RSI: 0000000000000246  RDI: 0000000000000000
    RBP: ffff9845d1a66780   R8: 0000000000000000   R9: 0000000000000000
    R10: 0000000000000000  R11: 0000000000000000  R12: ffff9845f2010d80
    R13: ffffbc7603e9fd28  R14: 0000000000000000  R15: ffff9845f766af80
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
--- <DOUBLEFAULT exception stack> ---
 #7 [ffffbc76000b8000] foo at ffffffffc0676066 [mod_d]
 #8 [ffffbc76000b8008] bar at ffffffffc067609f [mod_d]
 #9 [ffffbc76000b8020] foo at ffffffffc067606f [mod_d]
#10 [ffffbc76000b8038] bar at ffffffffc067609f [mod_d]
#11 [ffffbc76000b8050] foo at ffffffffc067606f [mod_d]
#12 [ffffbc76000b8068] bar at ffffffffc067609f [mod_d]
#13 [ffffbc76000b8080] foo at ffffffffc067606f [mod_d]
#14 [ffffbc76000b8098] bar at ffffffffc067609f [mod_d]
#15 [ffffbc76000b80b0] foo at ffffffffc067606f [mod_d]
#16 [ffffbc76000b80c8] bar at ffffffffc067609f [mod_d]
#17 [ffffbc76000b80e0] foo at ffffffffc067606f [mod_d]
#18 [ffffbc76000b80f8] bar at ffffffffc067609f [mod_d]
#19 [ffffbc76000b8110] foo at ffffffffc067606f [mod_d]
#20 [ffffbc76000b8128] bar at ffffffffc067609f [mod_d]
#21 [ffffbc76000b8140] foo at ffffffffc067606f [mod_d]
#22 [ffffbc76000b8158] bar at ffffffffc067609f [mod_d]
#23 [ffffbc76000b8170] foo at ffffffffc067606f [mod_d]
#24 [ffffbc76000b8188] bar at ffffffffc067609f [mod_d]
#25 [ffffbc76000b81a0] foo at ffffffffc067606f [mod_d]
#26 [ffffbc76000b81b8] bar at ffffffffc067609f [mod_d]
#27 [ffffbc76000b81d0] foo at ffffffffc067606f [mod_d]
#28 [ffffbc76000b81e8] bar at ffffffffc067609f [mod_d]
#29 [ffffbc76000b8200] foo at ffffffffc067606f [mod_d]
#30 [ffffbc76000b8218] bar at ffffffffc067609f [mod_d]
#31 [ffffbc76000b8230] foo at ffffffffc067606f [mod_d]
#32 [ffffbc76000b8248] bar at ffffffffc067609f [mod_d]
#33 [ffffbc76000b8260] foo at ffffffffc067606f [mod_d]
#34 [ffffbc76000b8278] bar at ffffffffc067609f [mod_d]
#35 [ffffbc76000b8290] foo at ffffffffc067606f [mod_d]
#36 [ffffbc76000b82a8] bar at ffffffffc067609f [mod_d]
#37 [ffffbc76000b82c0] foo at ffffffffc067606f [mod_d]
#38 [ffffbc76000b82d8] bar at ffffffffc067609f [mod_d]
#39 [ffffbc76000b82f0] foo at ffffffffc067606f [mod_d]
#40 [ffffbc76000b8308] bar at ffffffffc067609f [mod_d]
```

```
#41 [ffffbc76000b8320] foo at ffffffffc067606f [mod_d]
#42 [ffffbc76000b8338] bar at ffffffffc067609f [mod_d]
#43 [ffffbc76000b8350] foo at ffffffffc067606f [mod_d]
#44 [ffffbc76000b8368] bar at ffffffffc067609f [mod_d]
#45 [ffffbc76000b8380] foo at ffffffffc067606f [mod_d]
#46 [ffffbc76000b8398] bar at ffffffffc067609f [mod_d]
 -- MORE --  forward: <SPACE>, <ENTER> or j  backward: b or k  quit: qq
```

3. We can get the stack limit from the task structure (on x64 systems, the stack is 4 4K pages, 0x4000 bytes):

```
crash> task
PID: 3831   TASK: ffff9845f766af80  CPU: 3   COMMAND: "mod_d thread"
struct task_struct {
  thread_info = {
    flags = 16384,
    status = 0
  },
  state = 0,
  stack = 0xffffbc76000b8000,
  usage = {
    refs = {
      counter = 1
    }
  },
  flags = 2129984,
  ptrace = 0,
  on_cpu = 1,
  wake_entry = {
    llist = {
      next = 0x0
    },
    {
      u_flags = 48,
      a_flags = {
        counter = 48
      }
    },
    src = 0,
    dst = 0
  },
  cpu = 3,
  wakee_flips = 1,
  wakee_flip_decay_ts = 4294987395,
  last_wakee = 0xffff9845f75adf00,
  recent_used_cpu = 1,
  wake_cpu = 3,
  on_rq = 1,
  prio = 120,
  static_prio = 120,
  normal_prio = 120,
  rt_priority = 0,
  sched_class = 0xffffffff9a974c60 <fair_sched_class>,
  se = {
    load = {
      weight = 1048576,
      inv_weight = 4194304
    },
    run_node = {
      __rb_parent_color = 1,
```

```
      rb_right = 0x0,
      rb_left = 0x0
    },
    group_node = {
      next = 0xffff9846dbdb0710,
      prev = 0xffff9846dbdb0710
    },
    on_rq = 1,
    exec_start = 390541726402,
 -- MORE --  forward: <SPACE>, <ENTER> or j  backward: b or k  quit: qq
```

Note: We see from the backtrace that RSP reached stack region limit **0xffffbc76000b8000**.

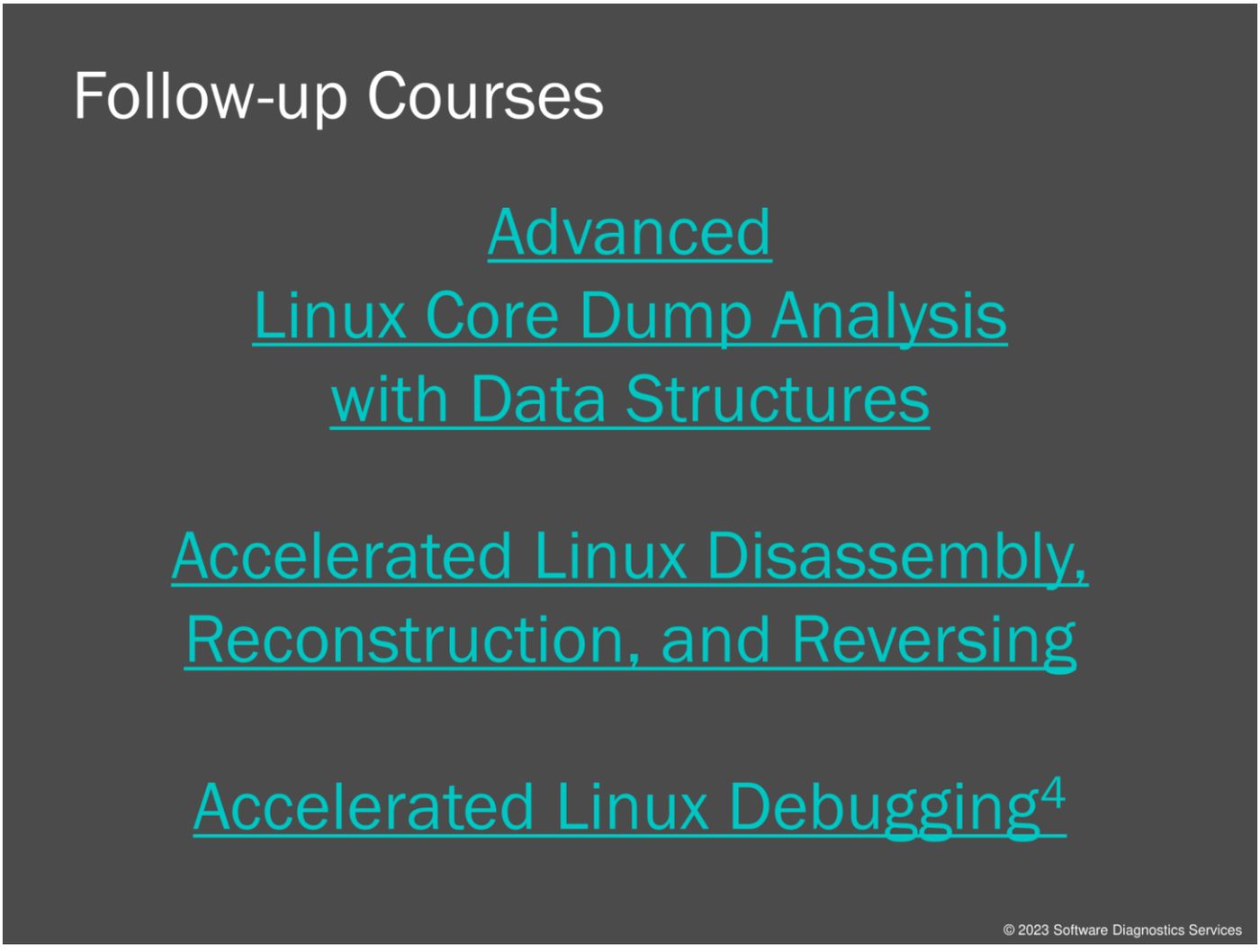
4. The stack region base is **0xffffbc76000b8000** + 0x4000 = **0xffffbc76000bc000**. We can specify RSP hint to **bt** command to get to the bottom of the stack trace (we choose a close address to the botton, **0xffffbc76000bc000** – 0x200 = **0xffffbc76000bbe00**):

```
crash> bt -S 0xffffbc76000bbe00
PID: 3831   TASK: ffff9845f766af80  CPU: 3   COMMAND: "mod_d thread"
 #0 [ffffbc76000bbe00] __schedule at ffffffff9a0c0112
 #1 [ffffbc76000bbe00] foo at ffffffffc067606f [mod_d]
 #2 [ffffbc76000bbe18] bar at ffffffffc067609f [mod_d]
 #3 [ffffbc76000bbe30] foo at ffffffffc067606f [mod_d]
 #4 [ffffbc76000bbe48] bar at ffffffffc067609f [mod_d]
 #5 [ffffbc76000bbe60] foo at ffffffffc067606f [mod_d]
 #6 [ffffbc76000bbe78] bar at ffffffffc067609f [mod_d]
 #7 [ffffbc76000bbe90] foo at ffffffffc067606f [mod_d]
 #8 [ffffbc76000bbea8] bar at ffffffffc067609f [mod_d]
 #9 [ffffbc76000bbec0] foo at ffffffffc067606f [mod_d]
#10 [ffffbc76000bbed8] bar at ffffffffc067609f [mod_d]
#11 [ffffbc76000bbef0] foo at ffffffffc067606f [mod_d]
#12 [ffffbc76000bbf08] kthread_f at ffffffffc0676016 [mod_d]
#13 [ffffbc76000bbf10] kthread at ffffffff998ac91b
#14 [ffffbc76000bbf50] ret_from_fork at ffffffff99804442
```

5. If we disassemble *foo* and *bar,* we see they call each other:

```
crash> dis foo
0xffffffffc0676060 <foo>:       nopl   0x0(%rax,%rax,1) [FTRACE NOP]
0xffffffffc0676065 <foo+5>:     push   %rbp
0xffffffffc0676066 <foo+6>:     push   %rbx
0xffffffffc0676067 <foo+7>:     mov    %rdi,%rbx
0xffffffffc067606a <foo+10>:    call   0xffffffffc0676090
0xffffffffc067606f <foo+15>:    lea    0x1(%rbx),%rdi
0xffffffffc0676073 <foo+19>:    mov    %rax,%rbp
0xffffffffc0676076 <foo+22>:    call   0xffffffffc0676090
0xffffffffc067607b <foo+27>:    pop    %rbx
0xffffffffc067607c <foo+28>:    add    %rbp,%rax
0xffffffffc067607f <foo+31>:    pop    %rbp
0xffffffffc0676080 <foo+32>:    ret
0xffffffffc0676081 <foo+33>:    nopw   %cs:0x0(%rax,%rax,1)
0xffffffffc067608b <foo+43>:    nopl   0x0(%rax,%rax,1)
```

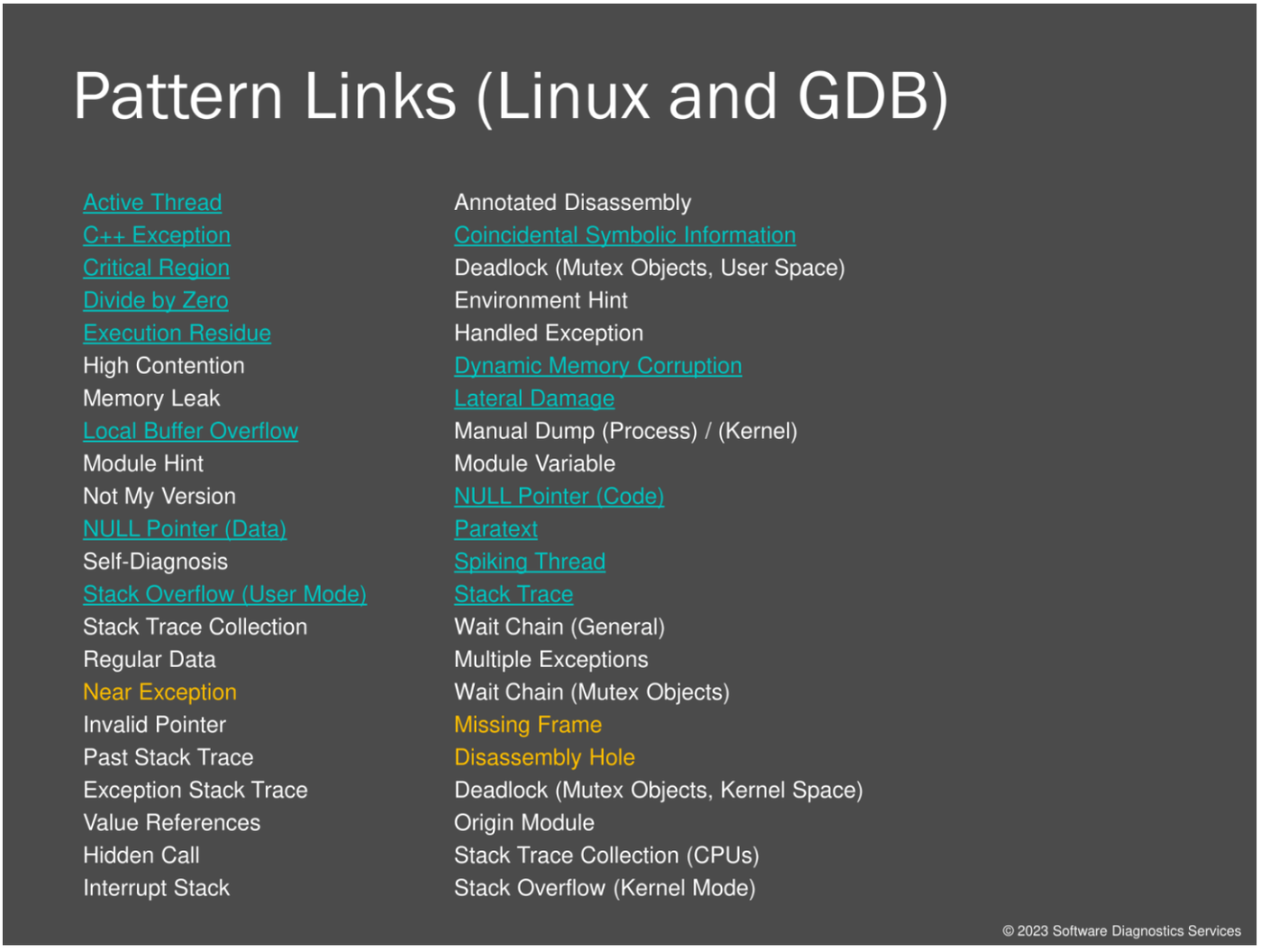
```
crash> dis bar
0xffffffffc0676090 <bar>:       nopl   0x0(%rax,%rax,1) [FTRACE NOP]
0xffffffffc0676095 <bar+5>:     push   %rbp
0xffffffffc0676096 <bar+6>:     push   %rbx
0xffffffffc0676097 <bar+7>:     mov    %rdi,%rbx
0xffffffffc067609a <bar+10>:    call   0xffffffffc0676060
0xffffffffc067609f <bar+15>:    lea    0x1(%rbx),%rdi
0xffffffffc06760a3 <bar+19>:    mov    %rax,%rbp
0xffffffffc06760a6 <bar+22>:    call   0xffffffffc0676060
0xffffffffc06760ab <bar+27>:    pop    %rbx
0xffffffffc06760ac <bar+28>:    add    %rbp,%rax
0xffffffffc06760af <bar+31>:    pop    %rbp
0xffffffffc06760b0 <bar+32>:    ret
0xffffffffc06760b1 <bar+33>:    add    %al,(%rax)
0xffffffffc06760b3 <bar+35>:    add    %al,(%rax)
0xffffffffc06760b5 <bar+37>:    add    %al,(%rax)
0xffffffffc06760b7 <bar+39>:    add    %al,(%rax)
0xffffffffc06760b9 <bar+41>:    add    %al,(%rax)
0xffffffffc06760bb <bar+43>:    add    %al,(%rax)
0xffffffffc06760bd <bar+45>:    add    %al,(%rax)
0xffffffffc06760bf <bar+47>:    add    %al,(%rax)
0xffffffffc06760c1 <bar+49>:    add    %al,(%rax)
0xffffffffc06760c3 <bar+51>:    add    %al,(%rax)
0xffffffffc06760c5 <bar+53>:    add    %al,(%rax)
0xffffffffc06760c7 <bar+55>:    add    %al,(%rax)
0xffffffffc06760c9 <bar+57>:    add    %al,(%rax)
0xffffffffc06760cb <bar+59>:    add    %al,(%rax)
0xffffffffc06760cd <bar+61>:    add    %al,(%rax)
0xffffffffc06760cf <bar+63>:    add    %al,(%rax)
0xffffffffc06760d1 <bar+65>:    add    %al,(%rax)
0xffffffffc06760d3 <bar+67>:    add    %al,(%rax)
0xffffffffc06760d5 <bar+69>:    add    %al,(%rax)
0xffffffffc06760d7 <bar+71>:    add    %al,(%rax)
0xffffffffc06760d9 <bar+73>:    add    %al,(%rax)
0xffffffffc06760db <bar+75>:    add    %al,(%rax)
0xffffffffc06760dd <bar+77>:    add    %al,(%rax)
0xffffffffc06760df <bar+79>:    add    %al,(%rax)
0xffffffffc06760e1 <bar+81>:    add    %al,(%rax)
0xffffffffc06760e3 <bar+83>:    add    %al,(%rax)
0xffffffffc06760e5 <bar+85>:    add    %al,(%rax)
0xffffffffc06760e7 <bar+87>:    add    %al,(%rax)
0xffffffffc06760e9 <bar+89>:    add    %al,(%rax)
0xffffffffc06760eb <bar+91>:    add    %al,(%rax)
0xffffffffc06760ed <bar+93>:    add    %al,(%rax)
0xffffffffc06760ef <bar+95>:    add    %al,(%rax)
0xffffffffc06760f1 <bar+97>:    add    %al,(%rax)
0xffffffffc06760f3 <bar+99>:    add    %al,(%rax)
0xffffffffc06760f5 <bar+101>:   add    %al,(%rax)
0xffffffffc06760f7 <bar+103>:   add    %al,(%rax)
0xffffffffc06760f9 <bar+105>:   add    %al,(%rax)
0xffffffffc06760fb <bar+107>:   add    %al,(%rax)
0xffffffffc06760fd <bar+109>:   add    %al,(%rax)
0xffffffffc06760ff <bar+111>:   add    %al,(%rax)
0xffffffffc0676101 <bar+113>:   add    %al,(%rax)
0xffffffffc0676103 <bar+115>:   add    %al,(%rax)
0xffffffffc0676105 <bar+117>:   add    %al,(%rax)
0xffffffffc0676107 <bar+119>:   add    %al,(%rax)
0xffffffffc0676109 <bar+121>:   add    %al,(%rax)
 -- MORE --  forward: <SPACE>, <ENTER> or j  backward: b or k  quit: qq
```



Advanced Linux Core Dump Analysis with Data Structures
https://www.patterndiagnostics.com/advanced-linux-core-dump-analysis

Accelerated Linux Disassembly, Reconstruction, and Reversing
https://www.patterndiagnostics.com/accelerated-linux-disassembly-reconstruction-reversing-book

Accelerated Linux Debugging[4]
https://www.patterndiagnostics.com/accelerated-linux-debugging-4d



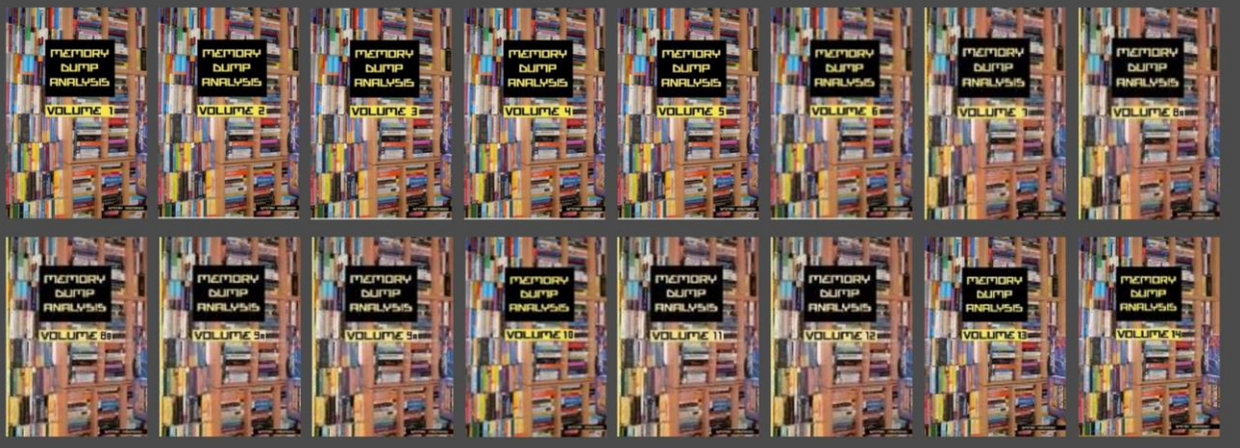
Here is the link to pattern descriptions and additional GDB examples:

http://www.dumpanalysis.org/blog/index.php/category/core-dump-analysis/

Selected pattern descriptions are provided at the end of this book.

Resources

- DumpAnalysis.org / SoftwareDiagnostics.Institute / PatternDiagnostics.com
- Debugging.TV / YouTube.com/DebuggingTV / YouTube.com/PatternDiagnostics
- Rosetta Stone for Debuggers
- Accelerated macOS Core Dump Analysis (also covers LLDB)
- GDB Pocket Reference
- A64 Instruction Set Architecture
- A64 Base Instructions
- Encyclopedia of Crash Dump Analysis Patterns, Third Edition
- Debugging, Disassembly & Reversing in Linux for x64 Architecture
- Foundations of Linux Debugging, Disassembling, and Reversing
- Foundations of ARM64 Linux Debugging, Disassembling, and Reversing
- Memory Dump Analysis Anthology (some articles in volumes 1, 7, 9A cover GDB)



WinDbg quick links
http://WinDbg.org

Software Diagnostics Institute
https://www.dumpanalysis.org

Debugging.TV
http://debugging.tv

Pattern Diagnostics Seminars
https://www.youtube.com/PatternDiagnostics

Software Diagnostics Services
https://www.patterndiagnostics.com

Encyclopedia of Crash Dump Analysis Patterns, 3rd edition
https://www.patterndiagnostics.com/encyclopedia-crash-dump-analysis-patterns

Memory Dump Analysis Anthology
https://www.patterndiagnostics.com/ultimate-memory-analysis-reference

Rosetta Stone for Debuggers
https://www.dumpanalysis.org/rosetta-stone-debuggers

Accelerated macOS Core Dump Analysis
https://www.patterndiagnostics.com/accelerated-macosx-core-dump-analysis-book

Foundations of Linux Debugging, Disassembling, and Reversing
https://www.patterndiagnostics.com/practical-foundations-linux-debugging-disassembling-reversing

Foundations of ARM64 Linux Debugging, Disassembling, and Reversing
https://www.patterndiagnostics.com/practical-foundations-arm64-linux-debugging-disassembling-reversing

A64 Instruction Set Architecture
https://developer.arm.com/documentation/102374/latest/

A64 Base Instructions
https://developer.arm.com/documentation/ddi0596/2021-12/Base-Instructions?lang=en

Selected Q&A

Q. What is **anon** in the *pmap* output?

A. This is anonymous memory that doesn't have filesystem backing. For example, memory allocated by *malloc* and thread stacks is **anon**.

Q. What does 'at' means in the **maintenance info sections** command output?

A. This might be related to section descriptions in object files. These numbers can be safely ignored in our analysis exercises. For further information, please see Binary File Descriptor library: https://en.wikipedia.org/wiki/Binary_File_Descriptor_library

Q. If we don't have the *pmap* output as an input for debugging, is there any method or sequence to know the *.data* region from the **maintenance info sections** command?

A. The output of the command contains the range of the *.data* section in the **Exec file** portion.

Q. Can we dump C/C++ code together with disassembly?

A. Yes, we can specify the **/s** option if symbols are available:

```
(gdb) disassemble /s main
Dump of assembler code for function main:
main.c:
64      {
   0x0000000000400888 <+0>:     stp     x29, x30, [sp, #-48]!
   0x000000000040088c <+4>:     mov     x29, sp
   0x0000000000400890 <+8>:     str     w0, [sp, #28]
   0x0000000000400894 <+12>:    str     x1, [sp, #16]
   0x0000000000400898 <+16>:    adrp    x0, 0x49c000 <tunable_list+1312>
   0x000000000040089c <+20>:    ldr     x0, [x0, #3024]
   0x00000000004008a0 <+24>:    ldr     x1, [x0]
   0x00000000004008a4 <+28>:    str     x1, [sp, #40]
   0x00000000004008a8 <+32>:    mov     x1, #0x0                        // #0

65          THREAD_CREATE(one)
   0x00000000004008ac <+36>:    add     x4, sp, #0x20
   0x00000000004008b0 <+40>:    mov     x3, #0x0                        // #0
   0x00000000004008b4 <+44>:    adrp    x0, 0x400000
   0x00000000004008b8 <+48>:    add     x2, x0, #0x754
   0x00000000004008bc <+52>:    mov     x1, #0x0                        // #0
   0x00000000004008c0 <+56>:    mov     x0, x4
   0x00000000004008c4 <+60>:    bl      0x40ee00 <pthread_create>

66          THREAD_CREATE(two)
   0x00000000004008c8 <+64>:    add     x4, sp, #0x20
   0x00000000004008cc <+68>:    mov     x3, #0x0                        // #0
   0x00000000004008d0 <+72>:    adrp    x0, 0x400000
   0x00000000004008d4 <+76>:    add     x2, x0, #0x798
   0x00000000004008d8 <+80>:    mov     x1, #0x0                        // #0
   0x00000000004008dc <+84>:    mov     x0, x4
   0x00000000004008e0 <+88>:    bl      0x40ee00 <pthread_create>

67          THREAD_CREATE(three)
   0x00000000004008e4 <+92>:    add     x4, sp, #0x20
   0x00000000004008e8 <+96>:    mov     x3, #0x0                        // #0
   0x00000000004008ec <+100>:   adrp    x0, 0x400000
   0x00000000004008f0 <+104>:   add     x2, x0, #0x7e0
```

```
   0x00000000004008f4 <+108>:   mov     x1, #0x0                        // #0
   0x00000000004008f8 <+112>:   mov     x0, x4
   0x00000000004008fc <+116>:   bl      0x40ee00 <pthread_create>

68          THREAD_CREATE(four)
   0x0000000000400900 <+120>:   add     x4, sp, #0x20
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400904 <+124>:   mov     x3, #0x0                        // #0
   0x0000000000400908 <+128>:   adrp    x0, 0x400000
   0x000000000040090c <+132>:   add     x2, x0, #0x824
   0x0000000000400910 <+136>:   mov     x1, #0x0                        // #0
   0x0000000000400914 <+140>:   mov     x0, x4
   0x0000000000400918 <+144>:   bl      0x40ee00 <pthread_create>

69          THREAD_CREATE(five)
   0x000000000040091c <+148>:   add     x4, sp, #0x20
   0x0000000000400920 <+152>:   mov     x3, #0x0                        // #0
   0x0000000000400924 <+156>:   adrp    x0, 0x400000
   0x0000000000400928 <+160>:   add     x2, x0, #0x86c
   0x000000000040092c <+164>:   mov     x1, #0x0                        // #0
   0x0000000000400930 <+168>:   mov     x0, x4
   0x0000000000400934 <+172>:   bl      0x40ee00 <pthread_create>

70
71          sleep(3);
   0x0000000000400938 <+176>:   mov     w0, #0x3                        // #3
   0x000000000040093c <+180>:   bl      0x41c490 <sleep>

72          return 0;
   0x0000000000400940 <+184>:   mov     w0, #0x0                        // #0

73      }
   0x0000000000400944 <+188>:   mov     w1, w0
   0x0000000000400948 <+192>:   adrp    x0, 0x49c000 <tunable_list+1312>
   0x000000000040094c <+196>:   ldr     x0, [x0, #3024]
   0x0000000000400950 <+200>:   ldr     x3, [sp, #40]
   0x0000000000400954 <+204>:   ldr     x2, [x0]
   0x0000000000400958 <+208>:   subs    x3, x3, x2
   0x000000000040095c <+212>:   mov     x2, #0x0                        // #0
   0x0000000000400960 <+216>:   b.eq    0x400968 <main+224>  // b.none
   0x0000000000400964 <+220>:   bl      0x41f930 <__stack_chk_fail_local>
   0x0000000000400968 <+224>:   mov     w0, w1
   0x000000000040096c <+228>:   ldp     x29, x30, [sp], #48
   0x0000000000400970 <+232>:   ret
End of assembler dump.
```

Q. Is this crash tool procedure to analyze Linux kernel core dumps also working for ESXi?

A. It should work if core dumps are generated with this tool: https://flings.vmware.com/vmss2core.

Q. Is there an **!analyze** equivalent command in GDB?

A. You can use GDB scripting to emulate some functionality. Crash tool shows some analysis information summary when you open a kernel core dump.

Q. Is it possible to use scripts in GDB?

A. Yes, for example, in the past, I wrote the following script to emulate the WinDbg **dpp** command (*UserCommands.txt*):

```
define dpp
    set $i = 0
    set $p = $arg0
    while $i < $arg1
        printf "%p: ", $p
        x/ga *(long *)$p
        set $i = $i + 1
        set $p = $p + 8
    end
end
```

We load the file in GDB and execute the **dpp** command supplying the initial address and the number of addresses to iterate (we also double-check its correctness):

```
(gdb) source UserCommands.txt

(gdb) dpp 0x7ffdf45637f8 10
0x7ffdf45637f8: 0x7ffdf4565756: 0x622f3d4c4c454853
0x7ffdf4563800: 0x7ffdf4565766: 0x544e4f4354534948
0x7ffdf4563808: 0x7ffdf456577d: 0x545349445f4c5357
0x7ffdf4563810: 0x7ffdf4565794: 0x5345443d454d414e
0x7ffdf4563818: 0x7ffdf45657a9: 0x6d6f682f3d445750
0x7ffdf4563820: 0x7ffdf45657c7: 0x3d454d414e474f4c
0x7ffdf4563828: 0x7ffdf45657d8: 0x4944504d545f434d
0x7ffdf4563830: 0x7ffdf45657f3: 0x313d4449535f434d
0x7ffdf4563838: 0x7ffdf45657fe: 0x6f682f3d454d4f48
0x7ffdf4563840: 0x7ffdf4565812: 0x5f6e653d474e414c

(gdb) x/a 0x7ffdf4563840
0x7ffdf4563840: 0x7ffdf4565812

(gdb) x/a  0x7ffdf4565812
0x7ffdf4565812: 0x5f6e653d474e414c
```

Q. Is I/O or PCI-mapped memory included in process core dumps?

A. Certain memory-mapped I/O pages like frame buffer are excluded from dumping according to a man page: https://man7.org/linux/man-pages/man5/core.5.html

Q. Is there a way to know how much each function takes space on a stack?

A. In WinDbg, it is possible by using the **kf** command variant:

```
0:000> kf
 #   Memory  Child-SP          RetAddr               Call Site
00           0000fffc`cd38e5f0 00000000`00424cb4     App1!_libc_nanosleep+0x24
01        40 0000fffc`cd38e630 00000000`004031f8     App1!sleep+0x110
02       1f0 0000fffc`cd38e820 00000000`0040320c     App1!bar_one+0x10
03        10 0000fffc`cd38e830 00000000`00403224     App1!foo_one+0xc
04        10 0000fffc`cd38e840 00000000`00404c34     App1!thread_one+0x10
05        20 0000fffc`cd38e860 00000000`00429b60     App1!start_thread+0xb4
06       130 0000fffc`cd38e990 ffffffff`ffffffff     App1!thread_start+0x30
07         0 0000fffc`cd38e990 00000000`00000000     0xffffffff`ffffffff
```

In GDB, it is possible by examining the stack pointer for each frame and calculating the difference.

Q. In case of multiple threads, will GDB show the thread which got a signal or another thread?

A. The thread that got a signal is thread #1 in the output of the **info threads** command.

Q. Sometimes, we get truncated core dumps. When does this happen?

A. This could be insufficient disk space or configured limit. Also, it could be that certain regions are excluded from dumping or by a dump filter (see the core man page referenced earlier).

Q. What happens if process memory is relocated?

A. You can add symbol offsets (-o option) for symbol-file and add-symbol-file GDB commands.

Q. Can I search for a pattern in the dump?

A. Yes, the **find** command for GDB and the **s** command for WinDbg. The two exercises, A1, contain corresponding examples.

Q. Can I dump entire memory contents from a core dump? For example, I want to examine the entire contents of the memory in one command.

A. The **find** command for GDB stops at invalid memory. The **s** command for WinDbg continues, although it may have memory size limitations. The **search** command in the *crash* wrapper may be used for the entire available kernel memory search.

Q. If a thread is in kernel context, do we get to know any info on what kernel function it was executing?

A. We can see from the top frame and get an idea, for example:

```
(gdb) info threads
  Id   Target Id         Frame
* 1    LWP 9             0x00007facb3d2a437 in __GI___waitpid (pid=-1, stat_loc=0x7ffc6b178670,
options=10)
    at ../sysdeps/unix/sysv/linux/waitpid.c:30

(gdb) bt
#0  0x00007facb3d2a437 in __GI___waitpid (pid=-1, stat_loc=0x7ffc6b178670, options=10)
    at ../sysdeps/unix/sysv/linux/waitpid.c:30
#1  0x00005637dc4e8869 in ?? ()
#2  0x00005637dc4e9cc3 in wait_for ()
#3  0x00005637dc4d7b85 in execute_command_internal ()
#4  0x00005637dc4d7df2 in execute_command ()
#5  0x00005637dc4bf833 in reader_loop ()
#6  0x00005637dc4be104 in main ()
```

Q. Sometimes, GDB says that it optimized away some local variables. Does it mean it doesn't use a stack for those variables (and uses registers)?

A. Yes, the values are in registers. Another optimization type I encountered in the past is reusing stack locations for different variables.

Q. Can I search for an address?

A. Yes, addresses are just 64-bit values, so you need to specify the **/g** option for GDB **find** command, the option **-64** for the **search** command in the *crash* tool, and the **q** type in the WinDbg **s** command. For example, see exercises App1 and K2.

Q. I got this output when I tried to load Exercise K1 core dump:

```
WARNING: kernel relocated [500MB]: patching 105514 gdb minimal_symbol values

      KERNEL: ../KSym/vmlinux-5.10.0-10-amd64
    DUMPFILE: dump.202201020022  [PARTIAL DUMP]
        CPUS: 4
        DATE: Sun Jan  2 00:19:33 2022
      UPTIME: 00:12:07
LOAD AVERAGE: 0.09, 0.07, 0.08
       TASKS: 454
    NODENAME: coredump
     RELEASE: 5.10.0-10-amd64
     VERSION: #1 SMP Debian 5.10.84-1 (2021-12-08)
     MACHINE: x86_64  (1991 Mhz)
      MEMORY: 4 GB
       PANIC:
crash: cannot determine length of symbol: log_end
```

A. Your distribution *crash* tool is older than the kernel. Therefore, you need to build the *crash* tool from the source. Please check the steps in exercise K1.

Q. When I load *x64\App1.core.253* in WinDbg, set the symbol path, and reload, I get only this stack trace:

```
0:000> k
 # Child-SP          RetAddr               Call Site
00 00007ffd`f4563610 00000000`00000000     App1+0x41a10
```

A. There's a problem at the time of this writing with the gcore-generated dumps on the latest Debian WSL2 distribution used for x64 exercises. It can be resolved by using the SYMOPT_LOAD_ANYTHING option and making sure that *App1* is in the search path:

```
0:000> .symopt+ 0x40
Symbol options are 0x30377:
  0x00000001 - SYMOPT_CASE_INSENSITIVE
  0x00000002 - SYMOPT_UNDNAME
  0x00000004 - SYMOPT_DEFERRED_LOADS
  0x00000010 - SYMOPT_LOAD_LINES
  0x00000020 - SYMOPT_OMAP_FIND_NEAREST
  0x00000040 - SYMOPT_LOAD_ANYTHING
  0x00000100 - SYMOPT_NO_UNQUALIFIED_LOADS
  0x00000200 - SYMOPT_FAIL_CRITICAL_ERRORS
  0x00010000 - SYMOPT_AUTO_PUBLICS
  0x00020000 - SYMOPT_NO_IMAGE_SEARCH

0:000> .reload
.
Unable to load image /home/coredump/ALCDA/App1/App1, Win32 error 0n2
*** WARNING: Unable to verify timestamp for App1

************* Symbol Loading Error Summary **************
```

```
Module name            Error
App1                   The system cannot find the file specified

You can troubleshoot most symbol related issues by turning on symbol loading diagnostics (!sym
noisy) and repeating the command that caused symbols to be loaded.
You should also verify that your symbol search path (.sympath) is correct.

0:000> k
 # Child-SP          RetAddr               Call Site
00 00007ffd`f4563610 00000000`0044199a     App1!nanosleep+0x40
01 00007ffd`f4563640 00000000`00401d92     App1!sleep+0x3a
02 00007ffd`f4563680 00000000`00407581     App1!main+0xaa
03 00007ffd`f45636d0 00000000`00401aba     App1!_libc_start_main+0x3d1
04 00007ffd`f45637d0 ffffffff`ffffffff     App1!start+0x2a
05 00007ffd`f45637d8 00000000`00000000     0xffffffff`ffffffff
```

App Source Code

App0

```
//
//  main.c
//  App0 - Exercise 0 - Testing Linux GDB
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -pthread -static -o App0
//

#include <stdlib.h>

void bar()
{
    abort();
}

void foo()
{
    bar();
}

int main(int argc, const char * argv[])
{
    foo();
    return 0;
}
```

App1

```
//
//  main.c
//  App1 - Normal application with multiple threads
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -pthread -static -o App1
//      gcc main.c -pthread -o App1.shared
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

#define THREAD_DECLARE(num) void bar_##num()\
{\
    sleep(-1);\
}\
\
void foo_##num()\
{\
    bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
    foo_##num();\
\
    return 0;\
}

THREAD_DECLARE(one)
THREAD_DECLARE(two)
THREAD_DECLARE(three)
THREAD_DECLARE(four)
THREAD_DECLARE(five)

#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}

int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(-1);
    return 0;
}
```

App2D

```
//
//  main.c
//  App2D - Shows NULL data pointer exception
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -pthread -static -o App2D
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void procA()
{
    int *p = NULL;

    *p = 1;
}

void procB()
{
    sleep(1);

    void (*pf)() = NULL;

    pf();
}

#define THREAD_DECLARE(num,func) void bar_##num()\
{\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,sleep(-1))
THREAD_DECLARE(two,procA())
THREAD_DECLARE(three,sleep(-1))
THREAD_DECLARE(four,procB())
THREAD_DECLARE(five,sleep(-1))
```

```
#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}

int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(3);
    return 0;
}
```

App2C

```
//
//  main.c
//  App2C - Shows NULL code pointer exception
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -pthread -static -o App2C
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void procA()
{
    sleep(2);

    int *p = NULL;

    *p = 1;
}

void procB()
{
    sleep(1);

    void (*pf)() = NULL;

    pf();
}

#define THREAD_DECLARE(num,func) void bar_##num()\
{\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,sleep(-1))
THREAD_DECLARE(two,procA())
THREAD_DECLARE(three,sleep(-1))
THREAD_DECLARE(four,procB())
THREAD_DECLARE(five,sleep(-1))
```

```
#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}

int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(3);
    return 0;
}
```

App2S

```
//
//  main.c
//  App2S - Shows how to use external debugging information
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -g -pthread -static -o App2S
//      cp App2S App2S.debug
//      objcopy --strip-debug App2S
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void procA()
{
    sleep(1);

    int *p = NULL;

    *p = 1;
}

void procB()
{
    sleep(2);

    void (*pf)() = NULL;

    pf();
}

#define THREAD_DECLARE(num,func) void bar_##num()\
{\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,sleep(-1))
THREAD_DECLARE(two,procA())
THREAD_DECLARE(three,sleep(-1))
THREAD_DECLARE(four,procB())
```

```
THREAD_DECLARE(five,sleep(-1))

#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}

int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(3);
    return 0;
}
```

App3

```
//
//  main.c
//  App3 - Spiking Thread pattern
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -pthread -lm -static -o App3
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>

void procA()
{
    while (1)
    {
        sleep(1);
    }
}

void procB()
{
    double d = 1.0/3.0;
    while (1)
    {
        d = sqrt(d);
    }
}

#define THREAD_DECLARE(num,func) void bar_##num()\
{\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,sleep(-1))
THREAD_DECLARE(two,sleep(-1))
THREAD_DECLARE(three,procA())
THREAD_DECLARE(four,sleep(-1))
THREAD_DECLARE(five,procB())
```

```
#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}

int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(-1);
    return 0;
}
```

App4

```
//
//  main.c
//  App4 - Heap Corruption pattern
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -pthread -static -o App4
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void proc()
{
    sleep(1);

    char *p1 = (char *) malloc (256);
    char *p2 = (char *) malloc (256);
    char *p3 = (char *) malloc (256);
    char *p4 = (char *) malloc (256);
    char *p5 = (char *) malloc (256);
    char *p6 = (char *) malloc (256);
    char *p7 = (char *) malloc (256);

    free(p6);
    free(p4);
    free(p2);

    strcpy(p2, "Hello Crash2! Hello Crash2! Hello Crash2! Hello Crash2! Hello Crash2!");
    strcpy(p4, "Hello Crash4! Hello Crash4! Hello Crash4! Hello Crash4! Hello Crash4! Hello
Crash4!");
    strcpy(p6, "Hello Crash6! Hello Crash6! Hello Crash6! Hello Crash6! Hello Crash6! Hello
Crash6! Hello Crash6!");

    p2 = (char *) malloc (256);
    p4 = (char *) malloc (256);
    p6 = (char *) malloc (256);

    sleep(300);

    free (p7);
    free (p6);
    free (p5);
    free (p4);
    free (p3);
    free (p2);
    free (p1);

    sleep(-1);
}
```

```
#define THREAD_DECLARE(num,func) void bar_##num()\
{\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,sleep(-1))
THREAD_DECLARE(two,sleep(-1))
THREAD_DECLARE(three,proc())
THREAD_DECLARE(four,sleep(-1))
THREAD_DECLARE(five,sleep(-1))

#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}

int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(-1);
    return 0;
}
```

App5

```
//
//  main.c
//  App5 - Local Buffer Overflow
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -pthread -static -o App5
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void procB(char *buffer)
{
    sleep(1);
    char data[100] = "My New Bigger Buffer";
    memcpy (buffer, data, sizeof(data));
}

void procA()
{
    char data[10] = "My Buffer";
    procB(data);
}

#define THREAD_DECLARE(num,func) void bar_##num()\
{\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,procA())
THREAD_DECLARE(two,sleep(-1))
THREAD_DECLARE(three,sleep(-1))
THREAD_DECLARE(four,sleep(-1))
THREAD_DECLARE(five,sleep(-1))

#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}
```

```
int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(-1);
    return 0;
}
```

App6

```
//
//  main.c
//  App6 - Stack Overflow
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -pthread -static -o App6
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void procF(int i)
{
    int buffer[128] = {-1, 0, i+1, 0, -1};

    procF(buffer[2]);
}

void procE()
{
    procF(1);
}

#define THREAD_DECLARE(num,func) void bar_##num()\
{\
sleep(300);\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,procE())
THREAD_DECLARE(two,sleep(-1))
THREAD_DECLARE(three,sleep(-1))
THREAD_DECLARE(four,sleep(-1))
THREAD_DECLARE(five,sleep(-1))

#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}
```

```
int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(-1);
    return 0;
}
```

App7

```
//
//  main.c
//  App7 - Divide by Zero and Active Threads
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//      gcc main.c -pthread -static -o App7

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void procF(int i)
{
    int buffer[1024] = {-1, 0, i+1, 0, -1};

    procF(buffer[2]);
}

void procE()
{
    procF(1);
}

int procD(int a, int b)
{
    return a/b;
}

int procC()
{
    return procD(1,0);
}

void procB(char *buffer)
{
    char data[100] = "My New Bigger Buffer";
    memcpy (buffer, data, sizeof(data));
}

void procA()
{
    char data[10] = "My Buffer";
    procB(data);
}
```

```
#define THREAD_DECLARE(num,func) void bar_##num()\
{\
sleep(10);\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,procA())
THREAD_DECLARE(two,sleep(-1))
THREAD_DECLARE(three,procC())
THREAD_DECLARE(four,sleep(-1))
THREAD_DECLARE(five,procE())

#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}

int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(-1);
    return 0;
}
```

App8

```
//
//  main.cpp
//  App8 - C++ Exception, Execution Residue, Handled Exception
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      g++ main.cpp -pthread -static -o App8
//

#include <string>
#include <unistd.h>

#define def_call(name,x,y) void name##_##x() { name##_##y(); }
#define def_final(name,x) void name##_##x() { }
#define def_init(name,y,size) void name() { int arr[size]; name##_##y(); *arr=0; }

def_final(work,9)
def_call(work,8,9)
def_call(work,7,8)
def_call(work,6,7)
def_call(work,5,6)
def_call(work,4,5)
def_call(work,3,4)
def_call(work,2,3)
def_call(work,1,2)
def_init(work,1,256)

class Exception
{
    int code;
    std::string description;

public:
    Exception(int _code, std::string _desc) : code(_code), description(_desc) {}
};

void procB()
{
    throw new Exception(5, "Access Denied");
}

void procNB()
{
    work();
}

void procA()
{
    procB();
}

void procNA()
{
    procNB();
}
```

```
void procH()
{
    try {
        procA();
    } catch (...) {
        sleep(-1);
    }
}

void procNH()
{
    sleep(10);
    procA();
}

void procNE()
{
    try {
        procNA();
    }
    catch (...)
    {
    }
    sleep(-1);
}

#define THREAD_DECLARE(num,func) void bar_##num()\
{\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,procNH())
THREAD_DECLARE(two,procNE())
THREAD_DECLARE(three,procH())
THREAD_DECLARE(four,procNE())
THREAD_DECLARE(five,procNE())

#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}
```

```c
int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(-1);
    return 0;
}
```

App9

```
//
//  main.c
//  App9 - Heap Leak pattern
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -pthread -static -o App9
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void procD()
{
}

typedef void (**PFUNC)();

void procC(int iter)
{
    for (int i = 0; i < iter; ++i)
    {
        char *p = malloc(256);
        strcpy(p, "allocated memory");

        *(PFUNC)(p + 32) = &procD;
    }
}

void procB()
{
    procC(250000);
    sleep(300);
    procC(250000);
    sleep(-1);
}

void procA()
{
    procC(5000);
    sleep(300);
    procB();
}
```

```
#define THREAD_DECLARE(num,func) void bar_##num()\
{\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,sleep(-1))
THREAD_DECLARE(two,procA())
THREAD_DECLARE(three,sleep(-1))
THREAD_DECLARE(four,sleep(-1))
THREAD_DECLARE(five,sleep(-1))

#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}

int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(-1);
    return 0;
}
```

App10

```
//
//  main.c
//  App10 - Heap Corruption, Heap Contention, Critical Region, Wait Chains, Self-Diagnostics
patterns
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build:
//
//      gcc main.c -pthread -static -o App10
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

#define ARR_SIZE 10000

char *pAllocBuf [ARR_SIZE] = {0};

void proc()
{
    while (1)
    {
        int idx = rand()%ARR_SIZE;
        int malloc_size = rand()%ARR_SIZE;

        if (pAllocBuf[idx])
        {
            free(pAllocBuf[idx]);
            pAllocBuf[idx] = 0;
        }

        pAllocBuf[idx] = malloc(malloc_size);
    }
}

#define THREAD_DECLARE(num,func) void bar_##num()\
{\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}
```

```
THREAD_DECLARE(one,proc())
THREAD_DECLARE(two,proc())
THREAD_DECLARE(three,proc())
THREAD_DECLARE(four,proc())
THREAD_DECLARE(five,proc())

#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}

int main(int argc, const char * argv[])
{
    THREAD_CREATE(one)
    THREAD_CREATE(two)
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(-1);
    return 0;
}
```

App11 / App12

```
//
//  main.c
//  App11 - Wait Chains, Deadlock, Handled Exception patterns
//
//  Copyright (c) 2015 - 2022 Software Diagnostics Services. All rights reserved.
//
//  Build (App11):
//
//      g++ main.cpp -pthread -static -o App11
//
//  Build (App12):
//
//      g++ main.cpp -g -pthread -static -o App12
//      cp App12 App12.debug
//      objcopy --strip-debug App12
//

#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

pthread_mutex_t mutexA, mutexB;

void procC()
{
    throw 0;
}

void procA()
{
    try
    {
        pthread_mutex_lock(&mutexA);
        procC();
        pthread_mutex_unlock(&mutexA);
    }
    catch(...)
    {

    }

    sleep(20);
    pthread_mutex_lock(&mutexB);
    pthread_mutex_unlock(&mutexB);
}

void procB()
{
    pthread_mutex_lock(&mutexB);
    pthread_mutex_lock(&mutexA);
    sleep(30);
    pthread_mutex_unlock(&mutexA);
    pthread_mutex_unlock(&mutexB);
}
```

```
#define THREAD_DECLARE(num,func) void bar_##num()\
{\
func;\
}\
\
void foo_##num()\
{\
bar_##num();\
}\
\
void * thread_##num (void *arg)\
{\
foo_##num();\
\
return 0;\
}

THREAD_DECLARE(one,sleep(-1))
THREAD_DECLARE(two,procA())
THREAD_DECLARE(three,sleep(-1))
THREAD_DECLARE(four,procB())
THREAD_DECLARE(five,sleep(-1))

#define THREAD_CREATE(num) {pthread_t threadID_##num; pthread_create (&threadID_##num, NULL,
thread_##num, NULL);}

int main(int argc, const char * argv[])
{
    pthread_mutex_init(&mutexA, NULL);
    pthread_mutex_init(&mutexB, NULL);

    THREAD_CREATE(one)
    THREAD_CREATE(two)
    sleep(10);
    THREAD_CREATE(three)
    THREAD_CREATE(four)
    THREAD_CREATE(five)

    sleep(-1);
    return 0;
}
```

K2

```
//
// mod_a.c
// Models NULL Pointer (Data) memory analysis pattern in kernel space
//
// Copyright (c) 2022 Software Diagnostics Services. All rights reserved.
//

#include <linux/module.h>

void foo(void);
void bar(void);

int init_module(void)
{
    foo();

    return 0;
}

void bar(void)
{
    int *pi = NULL;
    *pi = 1;
}

void foo(void)
{
    bar();
}

MODULE_LICENSE("GPL");
```

K3

```
//
// mod_b.c
// Models Null Pointer (Code) memory analysis pattern in kernel space
//
// Copyright (c) 2022 Software Diagnostics Services. All rights reserved.
//

#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>
#include <linux/kthread.h>
#include <linux/sched.h>
#include <linux/delay.h>

void foo(void);

int kthread_f(void *arg)
{
    msleep(10000);

    foo();

    return 0;
}

int init_module(void)
{
    struct task_struct *ts;

    ts = kthread_run(kthread_f, NULL, "mod_b thread");

    if (IS_ERR(ts))
    {
    return PTR_ERR(ts);
    }

    return 0;
}
```

```c
//
// foo.c
//

MODULE_LICENSE("GPL");

#include <linux/module.h>

void bar(void);

void foo(void)
{
    bar();
}

//
// bar.c
//

MODULE_LICENSE("GPL");

#include <linux/module.h>
#include <linux/kernel.h>

void bar(void)
{
    int (*pf)(void) = NULL;
    pf();
}

MODULE_LICENSE("GPL");
```

K4

```
//
// mod_c.c
// Models Spiking Thread memory analysis pattern in kernel space
//
// Copyright (c) 2022 Software Diagnostics Services. All rights reserved.
//

#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>
#include <linux/kthread.h>
#include <linux/sched.h>
#include <linux/delay.h>

void foo(void);

int kthread_f(void *arg)
{
    msleep(10000);

    foo();

    return 0;
}

int init_module(void)
{
    struct task_struct *ts;

    ts = kthread_run(kthread_f, NULL, "mod_c thread");

    if (IS_ERR(ts))
    {
    return PTR_ERR(ts);
    }

    return 0;
}
```

```c
//
// foo.c
//

MODULE_LICENSE("GPL");

#include <linux/module.h>

void foo(void)
{
    foo();
}

MODULE_LICENSE("GPL");
```

K5

```
//
// mod_d.c
// Models Stack Overflow (Kernel Mode) memory analysis pattern
//
// Copyright (c) 2022 Software Diagnostics Services. All rights reserved.
//

#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>
#include <linux/kthread.h>
#include <linux/sched.h>
#include <linux/delay.h>

long foo(long n);

int kthread_f(void *arg)
{
    msleep(10000);

    foo(0);

    return 0;
}

int init_module(void)
{
    struct task_struct *ts;

    ts = kthread_run(kthread_f, NULL, "mod_d thread");

    if (IS_ERR(ts))
    {
    return PTR_ERR(ts);
    }

    return 0;
}

MODULE_LICENSE("GPL");
```

```c
//
// foo.c
//

#include <linux/module.h>

long bar(long n);

long foo(long n)
{
    return bar(n) + bar(n + 1);
}

//
// bar.c
//

MODULE_LICENSE("GPL");

#include <linux/module.h>
#include <linux/kernel.h>

long foo(long n);

long bar(long n)
{
    return foo(n) + foo(n + 1);
}

MODULE_LICENSE("GPL");
```

Selected Analysis Patterns

(edited articles from Software Diagnostics Institute, www.DumpAnalysis.org)

NULL Pointer (Data)

This pattern is a Linux variant of **NULL Pointer** (data) pattern previously described for Mac OS X[2] and Windows[3] platforms:

```
(gdb) bt
#0  0x0000000000400500 in procA ()
#1  0x000000000040057a in bar_two ()
#2  0x000000000040058a in foo_two ()
#3  0x00000000004005a2 in thread_two ()
#4  0x0000000000401630 in start_thread (arg=<optimized out>)
at pthread_create.c:304
#5  0x00000000004324e9 in clone ()
#6  0x0000000000000000 in ?? ()

(gdb) x/i 0x400500
=> 0x400500 <procA+16>: movl   $0x1,(%rax)

(gdb) info r $rax
rax            0x0 0

(gdb) x $rax
0x0: Cannot access memory at address 0x0
```

[2] https://www.dumpanalysis.org/blog/index.php/2012/03/25/crash-dump-analysis-patterns-part-6b-mac-os-x/
[3] https://www.dumpanalysis.org/blog/index.php/2009/04/14/crash-dump-analysis-patterns-part-6b/

Incomplete Stack Trace

Users of WinDbg debugger accustomed to full thread stack traces will wonder whether a thread starts from *main*:

```
(gdb) bt
#0  0x000000000042fed1 in nanosleep ()
#1  0x000000000042fda0 in sleep ()
#2  0x000000000040078a in main ()
```

Of course, not, and by default, a stack trace is shown starting from the *main* function. You can change this behavior by using the following command:

```
(gdb) set backtrace past-main
```

Now we see an additional frame:

```
(gdb) bt
#0  0x000000000042fed1 in nanosleep ()
#1  0x000000000042fda0 in sleep ()
#2  0x000000000040078a in main ()
#3  0x0000000000405283 in __libc_start_main ()
#4  0x00000000004003e9 in _start ()
```

Stack Trace

This pattern is a Linux variant of **Stack Trace** pattern previously described for Mac OS X[4] and Windows[5] platforms. Here we show a stack trace when debug symbols are not available (stripped executable) and also how to apply debug symbols from the executable where they were preserved:

```
(gdb) bt
#0 0x000000000043e4f1 in nanosleep ()
#1 0x000000000043e3c0 in sleep ()
#2 0x0000000000400789 in main ()

(gdb) symbol-file ./App/App.debug
Reading symbols from /home/Apps/App/App.debug...done.

(gdb) bt
#0 0x000000000043e4f1 in nanosleep ()
#1 0x000000000043e3c0 in sleep ()
#2 0x0000000000400789 in main (argc=1, argv=0x7fff5d1572d8) at main.cpp:85
```

[4] https://www.dumpanalysis.org/blog/index.php/2012/03/25/crash-dump-analysis-patterns-part-25-mac-os-x/
[5] https://www.dumpanalysis.org/blog/index.php/2007/09/10/crash-dump-analysis-patterns-part-25/

NULL Pointer (Code)

This pattern is a Linux variant of **NULL Pointer** (code) pattern previously described for Mac OS X[6] and Windows[7] platforms:

```
(gdb) bt
#0  0x0000000000000000 in ?? ()
#1  0x0000000000400531 in procB ()
#2  0x00000000004005f8 in bar_four ()
#3  0x0000000000400608 in foo_four ()
#4  0x0000000000400620 in thread_four ()
#5  0x0000000000401630 in start_thread (arg=<optimized out>)
at pthread_create.c:304
#6  0x00000000004324e9 in clone ()
#7  0x0000000000000000 in ?? ()

(gdb) disassemble procB
Dump of assembler code for function procB:
0x0000000000400516 <+0>: push   %rbp
0x0000000000400517 <+1>: mov    %rsp,%rbp
0x000000000040051a <+4>: sub    $0x10,%rsp
0x000000000040051e <+8>: movq   $0x0,-0x8(%rbp)
0x0000000000400526 <+16>: mov    -0x8(%rbp),%rdx
0x000000000040052a <+20>: mov    $0x0,%eax
0x000000000040052f <+25>: callq  *%rdx
0x0000000000400531 <+27>: leaveq
0x0000000000400532 <+28>: retq
End of assembler dump.

(gdb) info r rdx
rdx            0x0 0
```

[6] https://www.dumpanalysis.org/blog/index.php/2012/05/03/crash-dump-analysis-patterns-part-6a-mac-os-x/
[7] https://www.dumpanalysis.org/blog/index.php/2008/04/28/crash-dump-analysis-patterns-part-6a/

Spiking Thread

This pattern is a variant of **Spiking Thread** pattern previously described for Mac OS X[8] and Windows[9] platforms:

```
(gdb) info threads
Id Target Id Frame
6 LWP 3712 0x00000000004329d1 in nanosleep ()
5 LWP 3717 0x00000000004007a3 in isnan ()
4 LWP 3716 0x00000000004329d1 in nanosleep ()
3 LWP 3715 0x00000000004329d1 in nanosleep ()
2 LWP 3714 0x00000000004329d1 in nanosleep ()
* 1 LWP 3713 0x00000000004329d1 in nanosleep ()
```

We notice a non-waiting thread and switch to it:

```
(gdb) thread 5
[Switching to thread 5 (LWP 3717)]
#0 0x00000000004007a3 in isnan ()

(gdb) bt
#0 0x00000000004007a3 in isnan ()
#1 0x0000000000400743 in sqrt ()
#2 0x0000000000400528 in procB ()
#3 0x0000000000400639 in bar_five ()
#4 0x0000000000400649 in foo_five ()
#5 0x0000000000400661 in thread_five ()
#6 0x0000000000403e30 in start_thread ()
#7 0x0000000000435089 in clone ()
#8 0x0000000000000000 in ?? ()
```

If we disassemble the return address for *procB* function to come back from *sqrt* call, we see an infinite loop:

```
(gdb) disassemble 0x400528
Dump of assembler code for function procB:
0x0000000000400500 <+0>: push   %rbp
0x0000000000400501 <+1>: mov    %rsp,%rbp
0x0000000000400504 <+4>: sub    $0x20,%rsp
0x0000000000400508 <+8>: movabs $0x3fd5555555555555,%rax
0x0000000000400512 <+18>: mov    %rax,-0x8(%rbp)
0x0000000000400516 <+22>: mov    -0x8(%rbp),%rax
0x000000000040051a <+26>: mov    %rax,-0x18(%rbp)
0x000000000040051e <+30>: movsd  -0x18(%rbp),%xmm0
0x0000000000400523 <+35>: callq  0x400710 <sqrt>
0x0000000000400528 <+40>: movsd  %xmm0,-0x18(%rbp)
0x000000000040052d <+45>: mov    -0x18(%rbp),%rax
0x0000000000400531 <+49>: mov    %rax,-0x8(%rbp)
0x0000000000400535 <+53>: jmp    0x400516 <procB+22>
End of assembler dump.
```

[8] https://www.dumpanalysis.org/blog/index.php/2012/05/09/crash-dump-analysis-patterns-part-14-mac-os-x/
[9] https://www.dumpanalysis.org/blog/index.php/2007/05/11/crash-dump-analysis-patterns-part-14/

Dynamic Memory Corruption (Process Heap)

This pattern is a Linux variant of **Dynamic Memory Corruption** (process heap) pattern previously described for Mac OS X[10] and Windows[11] platforms.

The corruption may be internal to heap structures with a subsequent memory access violation:

```
(gdb) bt
#0  0x000000000041482e in _int_malloc ()
#1  0x0000000000416d88 in malloc ()
#2  0x00000000004005dc in proc ()
#3  0x00000000004006ee in bar_three ()
#4  0x00000000004006fe in foo_three ()
#5  0x0000000000400716 in thread_three ()
#6  0x0000000000401760 in start_thread (arg=<optimized out>)
at pthread_create.c:304
#7  0x0000000000432609 in clone ()
#8  0x0000000000000000 in ?? ()

(gdb) x/i $rip
=> 0x41482e <_int_malloc+622>: mov    %rbx,0x10(%r12)

(gdb) x $r12+0x10
0x21687371: Cannot access memory at address 0x21687371

(gdb) p (char[4])0x21687371
$1 = "qsh!"
```

Or it may be detected with a diagnostic message (similar to double free):

```
(gdb) bt
#0  0x000000000043ef65 in raise ()
#1  0x0000000000409fc0 in abort ()
#2  0x000000000040bf5b in __libc_message ()
#3  0x0000000000412042 in malloc_printerr ()
#4  0x0000000000416c27 in free ()
#5  0x0000000000400586 in proc ()
#6  0x000000000040067e in bar_four ()
#7  0x000000000040068e in foo_four ()
#8  0x00000000004006a6 in thread_four ()
#9  0x00000000004016c0 in start_thread (arg=<optimized out>)
at pthread_create.c:304
#10 0x0000000000432589 in clone ()
#11 0x0000000000000000 in ?? ()
```

[10] https://www.dumpanalysis.org/blog/index.php/2012/05/27/crash-dump-analysis-patterns-part-2-mac-os-x/
[11] https://www.dumpanalysis.org/blog/index.php/2006/10/31/crash-dump-analysis-patterns-part-2/

Execution Residue (User Space)

This pattern is a Linux variant of **Execution Residue** pattern previously described for Mac OS X[12] and Windows[13] platforms. This residue is symbolic information left in a stack region, including ASCII and UNICODE fragments or pointers to them, for example, return addresses from past function calls:

```
(gdb) bt
#0  0x00000000004431f1 in nanosleep ()
#1  0x00000000004430c0 in sleep ()
#2  0x0000000000400771 in procNE() ()
#3  0x00000000004007aa in bar_two() ()
#4  0x00000000004007b5 in foo_two() ()
#5  0x00000000004007c8 in thread_two(void*) ()
#6  0x00000000004140f0 in start_thread (arg=<optimized out>)
at pthread_create.c:304
#7  0x0000000000445879 in clone ()
#8  0x0000000000000000 in ?? ()

(gdb) x/512a $rsp-2000
0x7f4cacc42360: 0x0 0x0
0x7f4cacc42370: 0x0 0x0
0x7f4cacc42380: 0x0 0x0
0x7f4cacc42390: 0x0 0x0
[...]
0x7f4cacc42830: 0x0 0x0
0x7f4cacc42840: 0x0 0x0
0x7f4cacc42850: 0x0 0x0
0x7f4cacc42860: 0x7f4cacc42870 0x4005af <_Z6work_8v+9>
0x7f4cacc42870: 0x7f4cacc42880 0x4005ba <_Z6work_7v+9>
0x7f4cacc42880: 0x7f4cacc42890 0x4005c5 <_Z6work_6v+9>
0x7f4cacc42890: 0x7f4cacc428a0 0x4005d0 <_Z6work_5v+9>
0x7f4cacc428a0: 0x7f4cacc428b0 0x4005db <_Z6work_4v+9>
0x7f4cacc428b0: 0x7f4cacc428c0 0x4005e6 <_Z6work_3v+9>
0x7f4cacc428c0: 0x7f4cacc428d0 0x4005f1 <_Z6work_2v+9>
0x7f4cacc428d0: 0x7f4cacc428e0 0x4005fc <_Z6work_1v+9>
0x7f4cacc428e0: 0x7f4cacc42cf0 0x40060e <_Z4workv+16>
0x7f4cacc428f0: 0x0 0x0
0x7f4cacc42900: 0x0 0x0
0x7f4cacc42910: 0x0 0x0
[...]
0x7f4cacc42af0: 0x0 0x0
0x7f4cacc42b00: 0x0 0x0
0x7f4cacc42b10: 0x0 0x0
0x7f4cacc42b20: 0x0 0x4431e6 <nanosleep+38>
0x7f4cacc42b30: 0x0 0x4430c0 <sleep+224>
0x7f4cacc42b40: 0x0 0x0
0x7f4cacc42b50: 0x0 0x0
0x7f4cacc42b60: 0x0 0x0
0x7f4cacc42b70: 0x0 0x0
[...]
0x7f4cacc42cb0: 0x0 0x0
0x7f4cacc42cc0: 0x0 0x0
0x7f4cacc42cd0: 0x0 0x0
0x7f4cacc42ce0: 0xfffffed2 0x3ad3affa
0x7f4cacc42cf0: 0x7f4cacc42d00 0x0
```

[12] https://www.dumpanalysis.org/blog/index.php/2012/06/05/crash-dump-analysis-patterns-part-60-mac-os-x/
[13] https://www.dumpanalysis.org/blog/index.php/2008/04/29/crash-dump-analysis-patterns-part-60/

```
0x7f4cacc42d00: 0x7f4cacc42d20 0x49c740 <default_attr>
0x7f4cacc42d10: 0x7f4cacc439c0 0x400771 <_Z6procNEv+19>
0x7f4cacc42d20: 0x7f4cacc42d30 0x4007aa <_Z7bar_twov+9>
0x7f4cacc42d30: 0x7f4cacc42d40 0x4007b5 <_Z7foo_twov+9>
0x7f4cacc42d40: 0x7f4cacc42d60 0x4007c8 <_Z10thread_twoPv+17>
0x7f4cacc42d50: 0x0 0x0
0x7f4cacc42d60: 0x0 0x4140f0 <start_thread+208>
0x7f4cacc42d70: 0x0 0x7f4cacc43700
0x7f4cacc42d80: 0x0 0x0
0x7f4cacc42d90: 0x0 0x0
[...]
```

However, supposed return addresses need to be checked for **Coincidental Symbolic Information** pattern.

Coincidental Symbolic Information

This pattern is a Linux variant of **Coincidental Symbolic Information** pattern previously described for Mac OS X[14] and Windows[15] platforms. The idea is the same: to disassemble the address to see if the preceding instruction is a call. If it is indeed, then most likely the symbolic address is a return address from past **Execution Residue**:

```
(gdb) x/i 0x4005e6
0x4005e6 <_Z6work_3v+9>: pop    %rbp

(gdb) disassemble 0x4005e6
Dump of assembler code for function _Z6work_3v:
0x00000000004005dd <+0>: push   %rbp
0x00000000004005de <+1>: mov    %rsp,%rbp
0x00000000004005e1 <+4>: callq  0x4005d2 <_Z6work_4v>
0x00000000004005e6 <+9>: pop    %rbp
0x00000000004005e7 <+10>: retq
End of assembler dump.

(gdb) x/4i 0x49c740-4
0x49c73c: add    %al,(%rax)
0x49c73e: add    %al,(%rax)
0x49c740 <default_attr>: add    %al,(%rax)
0x49c742 <default_attr+2>: add    %al,(%rax)
```

[14] https://www.dumpanalysis.org/blog/index.php/2012/06/09/crash-dump-analysis-patterns-part-24-mac-os-x/
[15] https://www.dumpanalysis.org/blog/index.php/2007/08/30/crash-dump-analysis-patterns-part-24/

Stack Overflow (User Mode)

This pattern is a Linux variant of **Stack Overflow** (user mode) pattern previously described for Mac OS X[16] and Windows[17] platforms:

```
(gdb) bt
#0  0x00000000004004fb in procF ()
#1  0x000000000040054b in procF ()
#2  0x000000000040054b in procF ()
#3  0x000000000040054b in procF ()
#4  0x000000000040054b in procF ()
#5  0x000000000040054b in procF ()
#6  0x000000000040054b in procF ()
#7  0x000000000040054b in procF ()
#8  0x000000000040054b in procF ()
#9  0x000000000040054b in procF ()
#10 0x000000000040054b in procF ()
#11 0x000000000040054b in procF ()
#12 0x000000000040054b in procF ()
[...]

(gdb) bt -10
#15409 0x000000000040054b in procF ()
#15410 0x000000000040054b in procF ()
#15411 0x000000000040054b in procF ()
#15412 0x000000000040055b in procE ()
#15413 0x0000000000400575 in bar_one ()
#15414 0x0000000000400585 in foo_one ()
#15415 0x000000000040059d in thread_one ()
#15416 0x0000000000401690 in start_thread (arg=<optimized out>)
at pthread_create.c:304
#15417 0x0000000000432549 in clone ()
#15418 0x0000000000000000 in ?? ()
```

In case of a stack overflow, the stack pointer is decremented beyond the stack region boundary into a non-accessible region, so any stack memory access triggers an access violation:

```
(gdb) x $rsp
0x7eff46109ec0: 0x0

(gdb) frame 1
#1  0x000000000040054b in procF ()

(gdb) x $rsp
0x7eff4610a0e0: 0x0

(gdb) maintenance info sections
[...]
Core file:
[...]
0x7eff46109000->0x7eff4610a000 at 0x02034000: load13 ALLOC LOAD READONLY HAS_CONTENTS
0x7eff4610a000->0x7eff4690a000 at 0x02035000: load14 ALLOC LOAD HAS_CONTENTS
[...]
```

[16] https://www.dumpanalysis.org/blog/index.php/2012/07/17/crash-dump-analysis-patterns-part-16b-mac-os-x/
[17] https://www.dumpanalysis.org/blog/index.php/2008/06/10/crash-dump-analysis-patterns-part-16b/

Divide by Zero (User Mode)

This pattern is a Linux variant of **Divide by Zero** (user mode) pattern previously described for Mac OS X[18] and Windows[19] platforms:

```
GNU gdb (GDB)
[...]
Program terminated with signal 8, Arithmetic exception.
#0 0x000000000040056f in procD ()

(gdb) x/i $rip
=> 0x40056f <procD+18>: idivl -0x8(%rbp)

(gdb) info r $rax
rax 0x1 1

(gdb) x/w $rbp-0x8
0x7f0f6806bd28: 0x00000000
```

[18] https://www.dumpanalysis.org/blog/index.php/2012/07/18/crash-dump-analysis-patterns-part-78a-mac-os-x/
[19] https://www.dumpanalysis.org/blog/index.php/2008/12/01/crash-dump-analysis-patterns-part-78a/

Local Buffer Overflow (User Space)

This pattern is a Linux variant of **Local Buffer Overflow** pattern previously described for Mac OS X[20] and Windows[21] platforms. Most of the time, simple mistakes in using memory and string manipulation functions are easily detected by the runtime. The more sophisticated example which overwrites stack trace without being detected involves overwriting indirectly via a pointer to a local buffer passed to the called function. In such cases, we might see incorrect and truncated stack traces:

```
(gdb) bt
#0  0x0000000000000000 in ?? ()
#1  0x0000000000000000 in ?? ()

(gdb) x/100a $rsp
[...]
0x7fc3dd9dece8: 0x0 0x0
0x7fc3dd9decf8: 0x0 0x0
0x7fc3dd9ded08: 0x0 0x0
0x7fc3dd9ded18: 0x0 0x0
0x7fc3dd9ded28: 0x7fc3dd9ded48 0x4005cc <procA+40>
0x7fc3dd9ded38: 0x422077654e20794d 0x7542207265676769
0x7fc3dd9ded48: 0x72656666 0x0
0x7fc3dd9ded58: 0x0 0x0
0x7fc3dd9ded68: 0x0 0x0
0x7fc3dd9ded78: 0x0 0x0
[...]
```

[20] https://www.dumpanalysis.org/blog/index.php/2012/07/19/crash-dump-analysis-patterns-part-36-mac-os-x/
[21] https://www.dumpanalysis.org/blog/index.php/2007/11/14/crash-dump-analysis-patterns-part-36/

C++ Exception

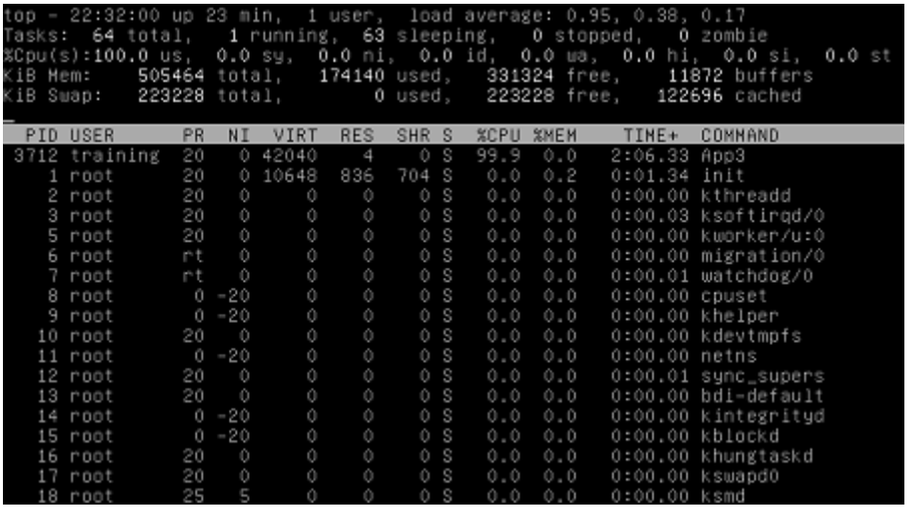
This pattern is a Linux variant of **C++ Exception** pattern previously described for Mac OS X[22] and Windows[23] platforms:

```
(gdb) bt
#0 0x00007f0a1d0e5165 in *__GI_raise ()
at ../nptl/sysdeps/unix/sysv/linux/raise.c:64
#1 0x00007f0a1d0e83e0 in *__GI_abort () at abort.c:92
#2 0x00007f0a1db5789d in __gnu_cxx::__verbose_terminate_handler() ()
from /usr/lib/x86_64-linux-gnu/libstdc++.so.6
#3 0x00007f0a1db55996 in ?? () from /usr/lib/x86_64-linux-gnu/libstdc++.so.6
#4 0x00007f0a1db559c3 in std::terminate() ()
from /usr/lib/x86_64-linux-gnu/libstdc++.so.6
#5 0x00007f0a1db55bee in __cxa_throw ()
from /usr/lib/x86_64-linux-gnu/libstdc++.so.6
#6 0x0000000000400dcf in procB() ()
#7 0x0000000000400e26 in procA() ()
#8 0x0000000000400e88 in procNH() ()
#9 0x0000000000400ea8 in bar_one() ()
#10 0x0000000000400eb3 in foo_one() ()
#11 0x0000000000400ec6 in thread_one(void*) ()
#12 0x00007f0a1d444b50 in start_thread ()
#13 0x00007f0a1d18e95d in clone ()
at ../sysdeps/unix/sysv/linux/x86_64/clone.S:112
#14 0x0000000000000000 in ?? ()
```

[22] https://www.dumpanalysis.org/blog/index.php/2012/07/20/crash-dump-analysis-patterns-part-77-mac-os-x/
[23] https://www.dumpanalysis.org/blog/index.php/2008/10/21/crash-dump-analysis-patterns-part-77/

Paratext

This pattern is Linux variant of **Paratext** pattern for Mac OS X[24]. Because of debugger tool limitations, additional software logs and the output of other tools may help in memory dump analysis. Typical examples of such pattern usage can be the list of modules with version and path info, application crash-specific information from instrumentation tools such as *Valgrind*, memory region names with attribution and boundaries, and CPU usage information. For example, *top* and *pmap* commands output:




```
14039: ./App1.shared
0000000000400000 4K r-x-- /home/training/ALCDA/App1/App1.shared
0000000000600000 4K rw--- /home/training/ALCDA/App1/App1.shared
0000000000611000 132K rw--- [ anon ]
00007fe8999a6000 4K ----- [ anon ]
00007fe8999a7000 8192K rw--- [ anon ]
00007fe89a1a7000 4K ----- [ anon ]
00007fe89a1a8000 8192K rw--- [ anon ]
00007fe89a9a8000 4K ----- [ anon ]
00007fe89a9a9000 8192K rw--- [ anon ]
00007fe89b1a9000 4K ----- [ anon ]
00007fe89b1aa000 8192K rw--- [ anon ]
00007fe89b9aa000 4K ----- [ anon ]
00007fe89b9ab000 8192K rw--- [ anon ]
00007fe89c1ab000 1540K r-x-- /lib/x86_64-linux-gnu/libc-2.13.so
00007fe89c32c000 2048K ----- /lib/x86_64-linux-gnu/libc-2.13.so
00007fe89c52c000 16K r---- /lib/x86_64-linux-gnu/libc-2.13.so
00007fe89c530000 4K rw--- /lib/x86_64-linux-gnu/libc-2.13.so
00007fe89c531000 20K rw--- [ anon ]
00007fe89c536000 92K r-x-- /lib/x86_64-linux-gnu/libpthread-2.13.so
00007fe89c54d000 2044K ----- /lib/x86_64-linux-gnu/libpthread-2.13.so
00007fe89c74c000 4K r---- /lib/x86_64-linux-gnu/libpthread-2.13.so
00007fe89c74d000 4K rw--- /lib/x86_64-linux-gnu/libpthread-2.13.so
00007fe89c74e000 16K rw--- [ anon ]
00007fe89c752000 128K r-x-- /lib/x86_64-linux-gnu/ld-2.13.so
00007fe89c966000 12K rw--- [ anon ]
00007fe89c96f000 8K rw--- [ anon ]
00007fe89c971000 4K r---- /lib/x86_64-linux-gnu/ld-2.13.so
00007fe89c972000 4K rw--- /lib/x86_64-linux-gnu/ld-2.13.so
00007fe89c973000 4K rw--- [ anon ]
00007ffd458c1000 132K rw--- [ stack ]
00007ffd459e9000 4K r-x-- [ anon ]
```

[24] https://www.dumpanalysis.org/blog/index.php/2012/07/28/crash-dump-analysis-patterns-part-180-mac-os-x/

```
ffffffffff600000 4K r-x-- [ anon ]
total 47208K
```

Active Thread

Here we publish a Linux variant of **Active Thread** pattern that was previously introduced for Mac OS X[25] and Windows[26]. Basically, it is a thread that is not waiting, sleeping, or suspended (most threads are). However, from a memory dump, it is not possible to find out whether it was **Spiking Thread** at the dump generation time (unless we have a set of memory snapshots and in each one, we have the same or similar backtrace), and we don't have any **Paratext** with CPU consumption stats for threads. For example, in one core dump, we have this thread:

```
(gdb) info threads
Id Target Id Frame
6 Thread 0x7f560d467700 (LWP 3483) 0x00000000004324a9 in clone ()
5 Thread 0x7f560c465700 (LWP 3485) 0x000000000042fe31 in nanosleep ()
4 Thread 0x7f560bc64700 (LWP 3486) 0x000000000042fe31 in nanosleep ()
3 Thread 0x7f560b463700 (LWP 3487) 0x000000000042fe31 in nanosleep ()
2 Thread 0x18b9860 (LWP 3482) 0x000000000042fe31 in nanosleep ()
1 Thread 0x7f560cc66700 (LWP 3484) 0x000000000042fe31 in nanosleep ()
```

Thread #6 is not waiting so we inspect its back trace:

```
(gdb) thread 6
[Switching to thread 6 (Thread 0x7f560d467700 (LWP 3483))]
#0 0x00000000004324a9 in clone ()

(gdb) bt
#0 0x00000000004324a9 in clone ()
#1 0x0000000000401560 in ?? () at pthread_create.c:217
#2 0x00007f560d467700 in ?? ()
#3 0x0000000000000000 in ?? ()

(gdb) x/i 0x4324a9
=> 0x4324a9 : test %rax,%rax
```

Perhaps the core dump was saved at the thread creation time.

[25] https://www.dumpanalysis.org/blog/index.php/2012/11/17/crash-dump-analysis-patterns-part-187-mac-os-x/
[26] https://www.dumpanalysis.org/blog/index.php/2015/10/31/crash-dump-analysis-patterns-part-232/

Lateral Damage

This pattern is a Linux variant of **Lateral Damage** pattern previously described for the Windows[27] platform. It also covers memory dumps where some usual commands may not work, and we have to find a workaround to simulate their output, for example, by using other commands:

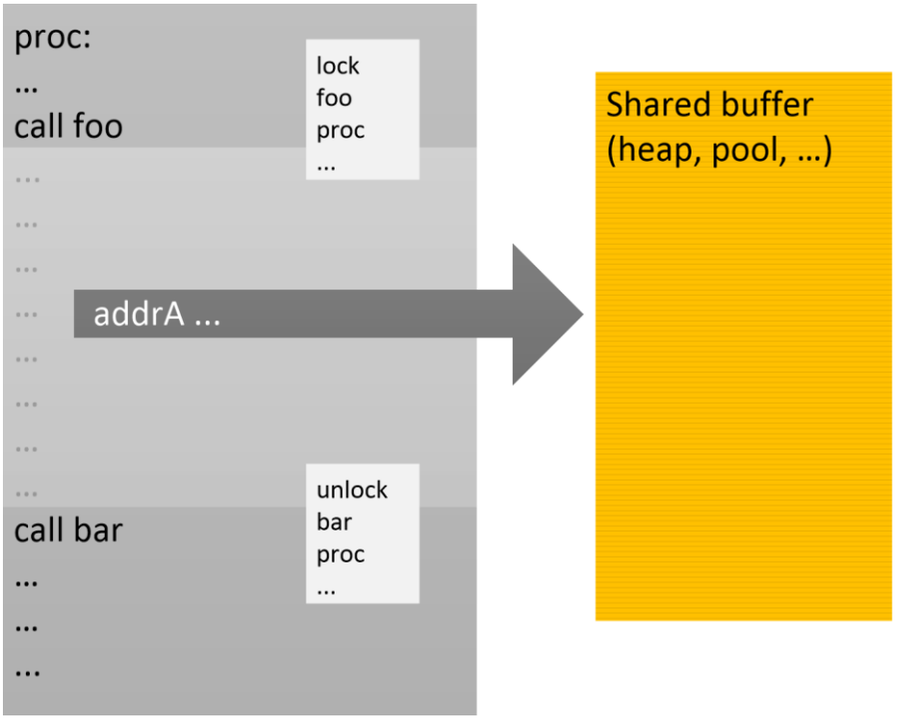
```
(gdb) info threads
Cannot find new threads: generic error

(gdb) thread apply all bt
Cannot find new threads: generic error

(gdb) thread 2
[Switching to thread 2 (LWP 12567)]
#0 0x000000000042ff51 in nanosleep ()

(gdb) thread 3
[Switching to thread 3 (LWP 12566)]
#0 0x000000000041482e in _int_malloc ()
```

[27] https://www.dumpanalysis.org/blog/index.php/2006/11/03/crash-dump-analysis-patterns-part-4/

Critical Region

We first introduced **Critical Region** pattern in Accelerated Mac OS X Core Dump Analysis[28] training but didn't submit the pattern itself to the catalog at that time.

A critical region is usually a region of code protected by synchronization objects such as critical sections and mutexes. However, **Critical Region** analysis pattern is about identifying code regions "sandwiched" between contending function calls (which may or may not involve synchronization objects and corresponding synchronization calls such as identified in Contention[29] patterns), and then identifying any possibly shared data referenced by such code regions:




```
(gdb) thread apply all bt

Thread 6 (Thread 0x7f2665377700 (LWP 17000)):
#0  0x00000000004151a1 in _int_malloc ()
#1  0x0000000000416cf8 in malloc ()
#2  0x00000000004005a4 in proc ()
#3  0x0000000000400604 in bar_two ()
#4  0x0000000000400614 in foo_two ()
#5  0x000000000040062c in thread_two ()
#6  0x00000000004016c0 in start_thread (arg=<optimized out>)
at pthread_create.c:304
#7  0x0000000000432589 in clone ()
#8  0x0000000000000000 in ?? ()
```

[28] https://www.patterndiagnostics.com/accelerated-macosx-core-dump-analysis-book
[29] https://www.dumpanalysis.org/blog/index.php/2010/09/21/contention-patterns/

```
Thread 5 (Thread 0x7f2664b76700 (LWP 17001)):
#0  __lll_unlock_wake_private ()
at ../nptl/sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:343
#1  0x000000000041886d in _L_unlock_9670 ()
#2  0x0000000000416d22 in malloc ()
#3  0x00000000004005a4 in proc ()
#4  0x0000000000400641 in bar_three ()
#5  0x0000000000400651 in foo_three ()
#6  0x0000000000400669 in thread_three ()
#7  0x00000000004016c0 in start_thread (arg=<optimized out>)
at pthread_create.c:304
#8  0x0000000000432589 in clone ()
#9  0x0000000000000000 in ?? ()

Thread 4 (Thread 0x7f2665b78700 (LWP 16999)):
#0  __lll_lock_wait_private ()
at ../nptl/sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:97
#1  0x0000000000418836 in _L_lock_9558 ()
#2  0x0000000000416c1c in free ()
#3  0x0000000000400586 in proc ()
#4  0x00000000004005c7 in bar_one ()
#5  0x00000000004005d7 in foo_one ()
#6  0x00000000004005ef in thread_one ()
#7  0x00000000004016c0 in start_thread (arg=<optimized out>)
at pthread_create.c:304
#8  0x0000000000432589 in clone ()
#9  0x0000000000000000 in ?? ()

Thread 3 (Thread 0x1ab1860 (LWP 16998)):
#0  0x000000000042fed1 in nanosleep ()
#1  0x000000000042fda0 in sleep ()
#2  0x000000000040078a in main ()

Thread 2 (Thread 0x7f2663b74700 (LWP 17003)):
#0  __lll_lock_wait_private ()
at ../nptl/sysdeps/unix/sysv/linux/x86_64/lowlevellock.S:97
#1  0x0000000000418836 in _L_lock_9558 ()
#2  0x0000000000416c1c in free ()
#3  0x0000000000400586 in proc ()
#4  0x00000000004006bb in bar_five ()
#5  0x00000000004006cb in foo_five ()
#6  0x00000000004006e3 in thread_five ()
#7  0x00000000004016c0 in start_thread (arg=<optimized out>)
at pthread_create.c:304
#8  0x0000000000432589 in clone ()
#9  0x0000000000000000 in ?? ()

Thread 1 (Thread 0x7f2664375700 (LWP 17002)):
#0  0x000000000043ef65 in raise ()
#1  0x0000000000409fc0 in abort ()
#2  0x000000000040bf5b in __libc_message ()
#3  0x0000000000412042 in malloc_printerr ()
#4  0x0000000000416c27 in free ()
#5  0x0000000000400586 in proc ()
#6  0x000000000040067e in bar_four ()
#7  0x000000000040068e in foo_four ()
#8  0x00000000004006a6 in thread_four ()
#9  0x00000000004016c0 in start_thread (arg=<optimized out>)
at pthread_create.c:304
```

```
#10 0x0000000000432589 in clone ()
#11 0x0000000000000000 in ?? ()
```

From threads #4 and #5, we can identify one such a region with a shared buffer 0x6b8fc0, which may further point to heap entries.

```
(gdb) disassemble proc
Dump of assembler code for function proc:
0x00000000004004f0 <+0>: push   %rbp
0x00000000004004f1 <+1>: mov    %rsp,%rbp
0x00000000004004f4 <+4>: push   %rbx
0x00000000004004f5 <+5>: sub    $0x18,%rsp
0x00000000004004f9 <+9>: callq  0x40ac70 <rand>
0x00000000004004fe <+14>: mov    %eax,%ecx
0x0000000000400500 <+16>: mov    $0x68db8bad,%edx
0x0000000000400505 <+21>: mov    %ecx,%eax
0x0000000000400507 <+23>: imul   %edx
0x0000000000400509 <+25>: sar    $0xc,%edx
0x000000000040050c <+28>: mov    %ecx,%eax
0x000000000040050e <+30>: sar    $0x1f,%eax
0x0000000000400511 <+33>: mov    %edx,%ebx
0x0000000000400513 <+35>: sub    %eax,%ebx
0x0000000000400515 <+37>: mov    %ebx,%eax
0x0000000000400517 <+39>: mov    %eax,-0x14(%rbp)
0x000000000040051a <+42>: mov    -0x14(%rbp),%eax
0x000000000040051d <+45>: imul   $0x2710,%eax,%eax
0x0000000000400523 <+51>: mov    %ecx,%edx
0x0000000000400525 <+53>: sub    %eax,%edx
0x0000000000400527 <+55>: mov    %edx,%eax
0x0000000000400529 <+57>: mov    %eax,-0x14(%rbp)
0x000000000040052c <+60>: callq  0x40ac70 <rand>
0x0000000000400531 <+65>: mov    %eax,%ecx
0x0000000000400533 <+67>: mov    $0x68db8bad,%edx
0x0000000000400538 <+72>: mov    %ecx,%eax
0x000000000040053a <+74>: imul   %edx
0x000000000040053c <+76>: sar    $0xc,%edx
0x000000000040053f <+79>: mov    %ecx,%eax
0x0000000000400541 <+81>: sar    $0x1f,%eax
0x0000000000400544 <+84>: mov    %edx,%ebx
0x0000000000400546 <+86>: sub    %eax,%ebx
0x0000000000400548 <+88>: mov    %ebx,%eax
0x000000000040054a <+90>: mov    %eax,-0x18(%rbp)
0x000000000040054d <+93>: mov    -0x18(%rbp),%eax
0x0000000000400550 <+96>: imul   $0x2710,%eax,%eax
0x0000000000400556 <+102>: mov    %ecx,%edx
0x0000000000400558 <+104>: sub    %eax,%edx
0x000000000040055a <+106>: mov    %edx,%eax
0x000000000040055c <+108>: mov    %eax,-0x18(%rbp)
0x000000000040055f <+111>: mov    -0x14(%rbp),%eax
0x0000000000400562 <+114>: cltq
0x0000000000400564 <+116>: mov    0x6b8fc0(,%rax,8),%rax
0x000000000040056c <+124>: test   %rax,%rax
0x000000000040056f <+127>: je     0x400597 <proc+167>
0x0000000000400571 <+129>: mov    -0x14(%rbp),%eax
0x0000000000400574 <+132>: cltq
0x0000000000400576 <+134>: mov    0x6b8fc0(,%rax,8),%rax
0x000000000040057e <+142>: mov    %rax,%rdi
```

```
0x0000000000400581 <+145>: callq  0x416bc0 <free>
0x0000000000400586 <+150>: mov    -0x14(%rbp),%eax
0x0000000000400589 <+153>: cltq
0x000000000040058b <+155>: movq   $0x0,0x6b8fc0(,%rax,8)
0x0000000000400597 <+167>: mov    -0x18(%rbp),%eax
0x000000000040059a <+170>: cltq
0x000000000040059c <+172>: mov    %rax,%rdi
0x000000000040059f <+175>: callq  0x416c90 <malloc>
0x00000000004005a4 <+180>: mov    %rax,%rdx
0x00000000004005a7 <+183>: mov    -0x14(%rbp),%eax
0x00000000004005aa <+186>: cltq
0x00000000004005ac <+188>: mov    %rdx,0x6b8fc0(,%rax,8)
0x00000000004005b4 <+196>: jmpq   0x4004f9 <proc+9>
End of assembler dump.
```