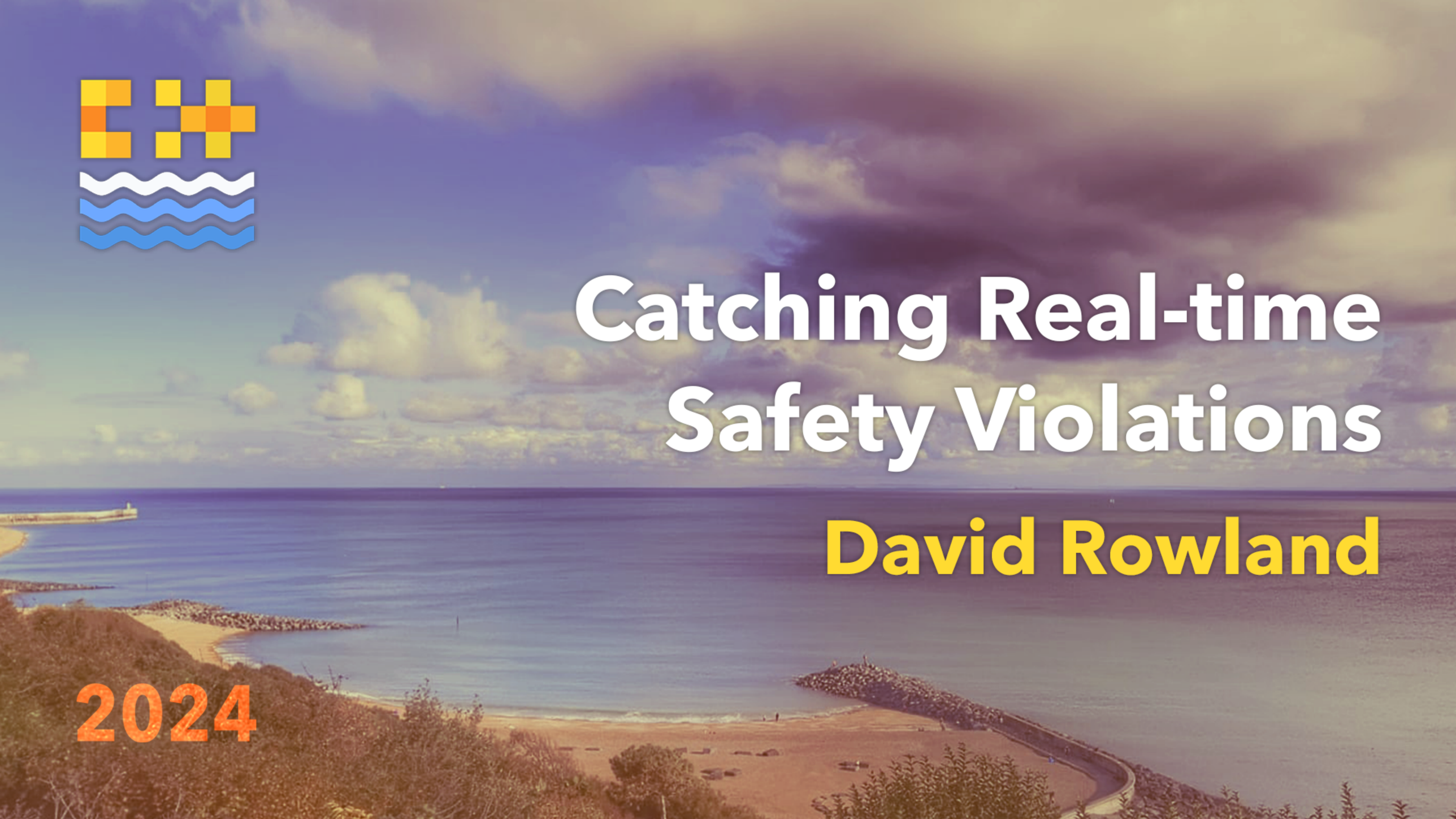


Catching Real-time Safety Violations

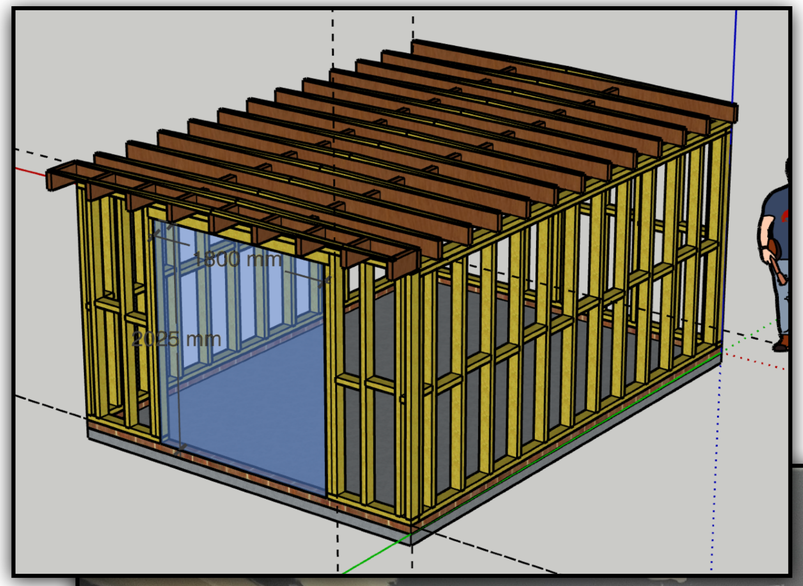
David Rowland

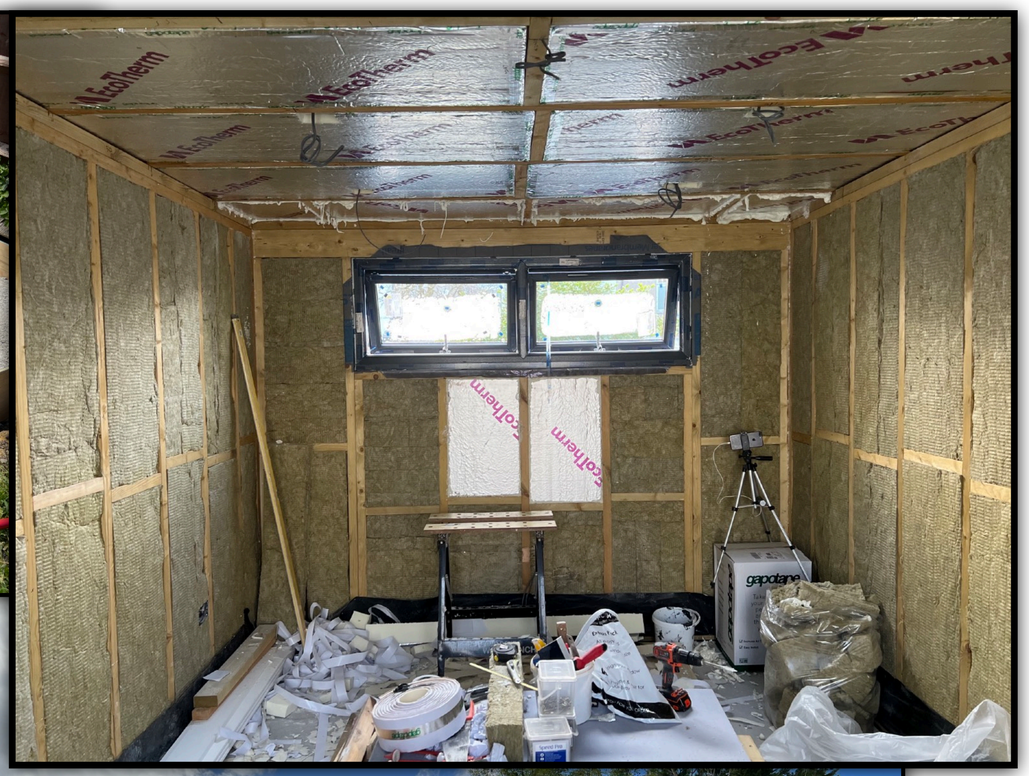
2024

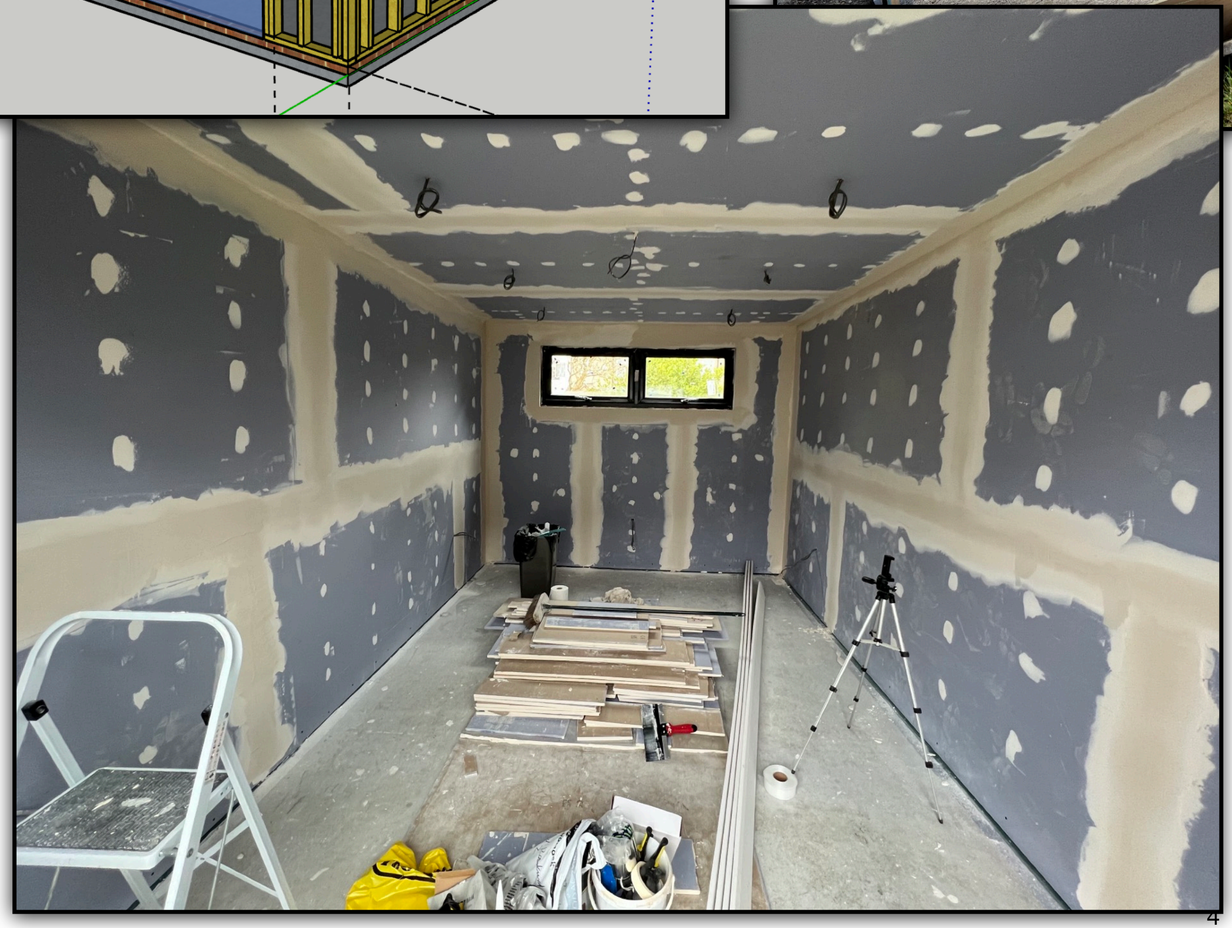
PRISM SOUND

tracktion

POWERED BY tracktion ENGINE


1500 mm
2025 mm


gapotane



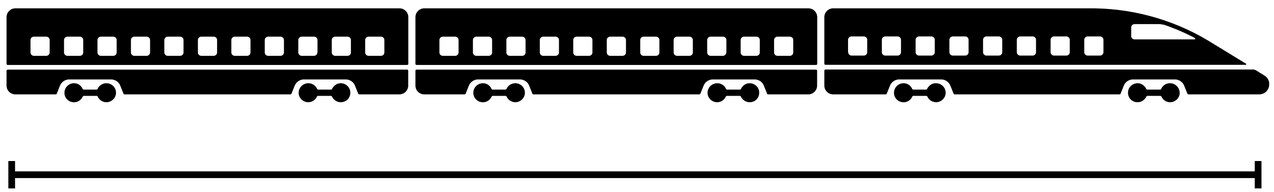

JUTA
Breather Membranes
MASTER
BS5534

Intro

- Mostly discuss macOS/Linux
- Different on Windows?
- Not exhaustive!
- There will be quiz questions!

Latency vs Bandwidth

- Bandwidth is how “much data”
- Latency is “how quickly”

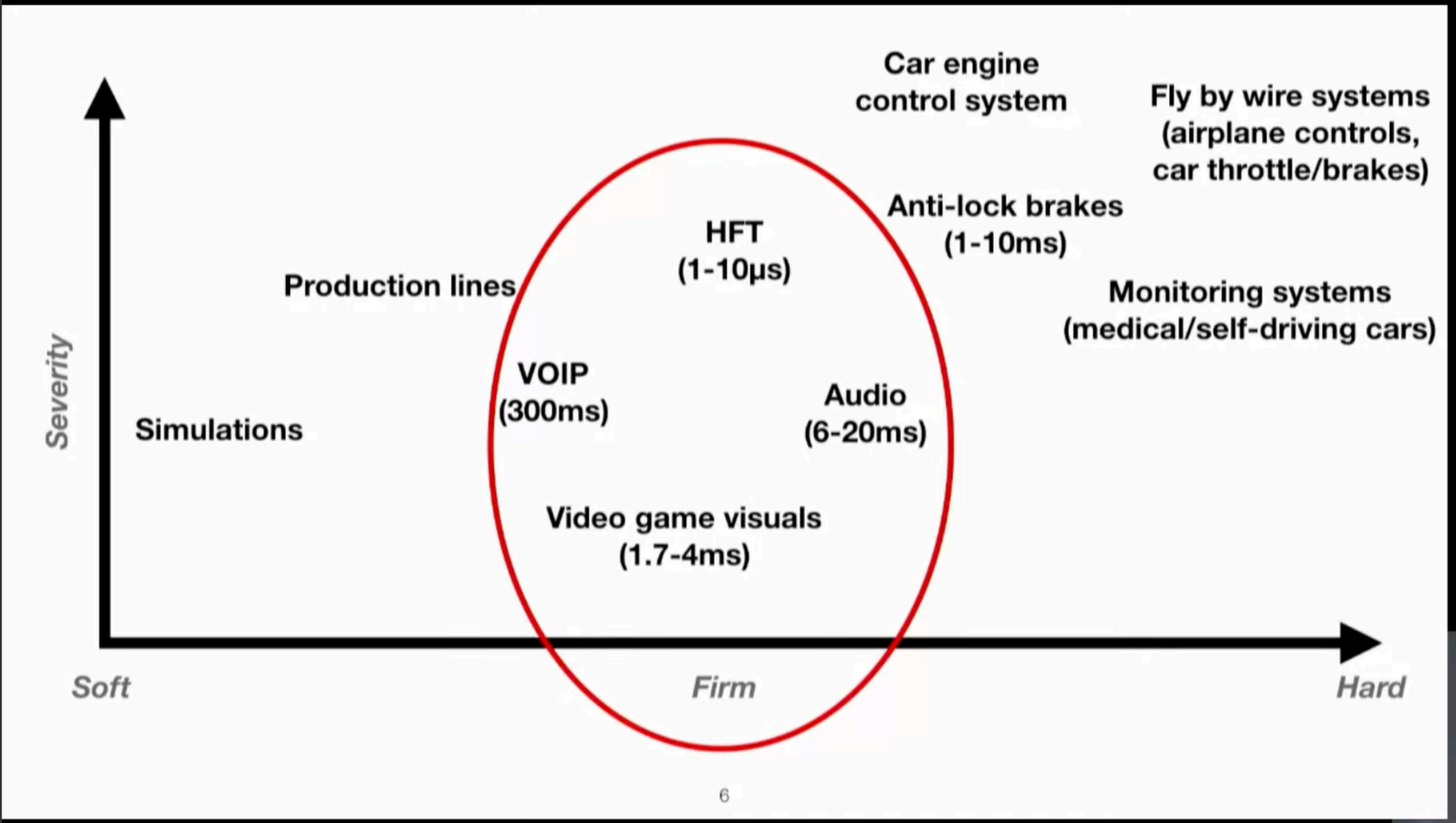


Bandwith

Real-time vs Low-latency

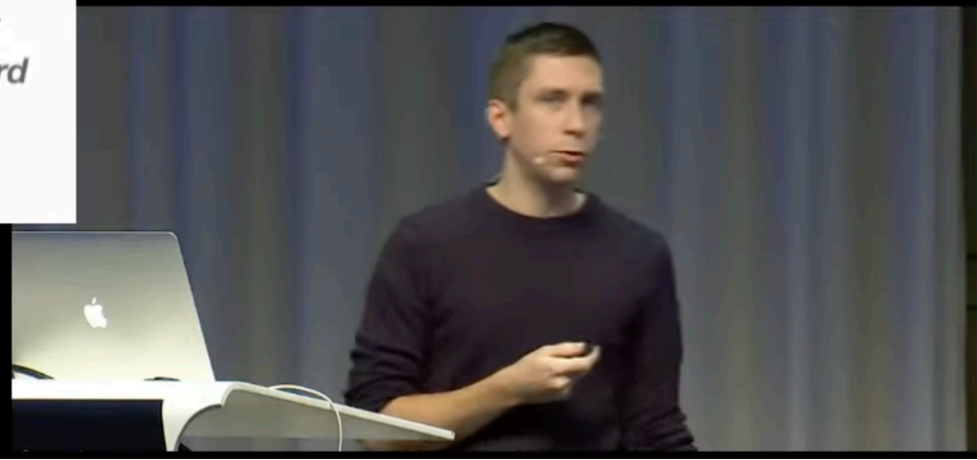
- Real-time has deadlines (“soft" to “hard”)

Meeting C++ 2019



David Rowland
Fabian Renn-Giles

Real-time 101



What is “real-time”?

- ***Not necessarily*** “wait-free”
- Counting “real-time” as “making forward progress”
  - At least one thread makes progress
  - No threads are starved
- Avoid:
  - Unbounded calls
  - Locks

What do we want to catch?

- System calls
  - Thread scheduling/synchronisation
  - I/O
    - File system accesses e.g. stat/open/read/write/close
    - Network activity
- Memory allocation/deallocation e.g. malloc/free
  - Including throwing exceptions
- Locks
  - Can lead to priority inversion
  - Can lead to system calls
  - Can lead to thread starvation

What you *might* want catch…

- Amortised O(1) calls
  - `std::*map`
    - Assuming no allocations:
    - Average O(1), worst O(N)
  - `std::uniform_*distribution`
    - In theory unbounded as they are unbiased

Cppcon 2021 The C++ Conference | October 24-29

Timur Doumler

Real-Time programming with the C++ standard library

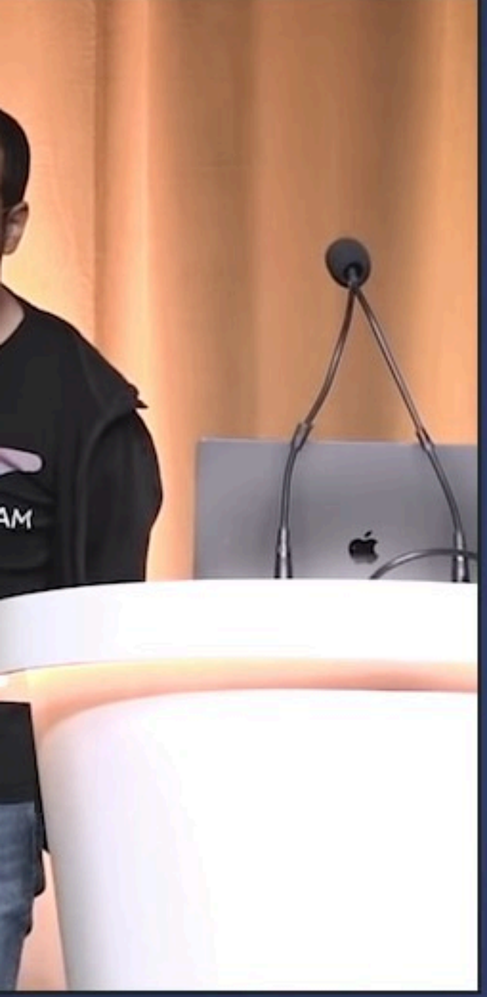
Peter Bindels @dascandy42 · Jan 7, 2020
Replying to @timur_audio
They can either be perfectly unbiased or constant time. The std ones are perfectly unbiased, so not constant time, only amortized constant time.
1

Timur Doumler @timur_audio · Jan 7, 2020
Why are these two properties mutually exclusive?
2

Peter Bindels @dascandy42 · Jan 7, 2020
Take a die (6-sided). Roll it. Now somehow make this into a balanced number between 1-5. 1-5 are easy - just direct map.

You get to pick what you do with the 6. Reroll or map to some number. This is the same problem, except a 4.3 billion sided die.
1 6

Peter Bindels @dascandy42 · Jan 7, 2020
And of course, if you map to some number it's biased to that number, if you reroll it *could* keep coming up 6es.
1 1



oumler

rogramming
+ standard

Peter Bindels @dascandy42 · Jan 7, 2020

Replying to @timur_audio

They can either be perfectly unbiased or constant time. The std ones are perfectly unbiased, so not constant time, only amortized constant time.

1

Timur Doumler @timur_audio · Jan 7, 2020

Why are these two properties mutually exclusive?

2

Peter Bindels @dascandy42 · Jan 7, 2020

Take a die (6-sided). Roll it. Now somehow make this into a balanced number between 1-5. 1-5 are easy - just direct map.

You get to pick what you do with the 6. Reroll or map to some number. This is the same problem, except a 4.3 billion sided die.

1 6

Peter Bindels @dascandy42 · Jan 7, 2020

And of course, if you map to some number it's biased to that number, if you reroll it *could* keep coming up 6es.

1 1

Timur Doumler

Real-Time programming with the C++ standard library

- don't call anything that might block
  (non-deterministic execution time + priority inversion!)
  - don't try to acquire a mutex
  - don't allocate / deallocate memory
  - don't do any I/O
  - don't interact with the thread scheduler
  - don't do any other system calls
- don't call any 3rdparty code if you don't know what it's doing
- don't use algorithms with > *O(1)* complexity
- don't use algorithms with *amortised O(1)* complexity

How do you know what 3rd party code is doing?

What do we want to catch?

- System calls
- malloc/free
- lock/unlock

```cpp
int main()
{
    std::thread t1 ([&] { run_rt_thread();});
    t1.join();
```

```cpp
std::uintmax_t get_file_size()
{
    std::filesystem::path awk_path ("/usr/bin/awk");
    std::uintmax_t file_size
        = std::filesystem::file_size (awk_path);

    return file_size;
}

void do_malloc_free()
{
    auto m = malloc (1024);
    free (m);
}

void do_vector_reserve()
{
    std::vector<int> v;
    v.reserve (42);
}

void do_mutex_lock_unlock (std::mutex& m)
{
    std::unique_lock l (m);
}
```

What do we want to catch?

- System calls

`get_file_size`

- malloc/free

`do_malloc_free`

`do_vector_reserve`

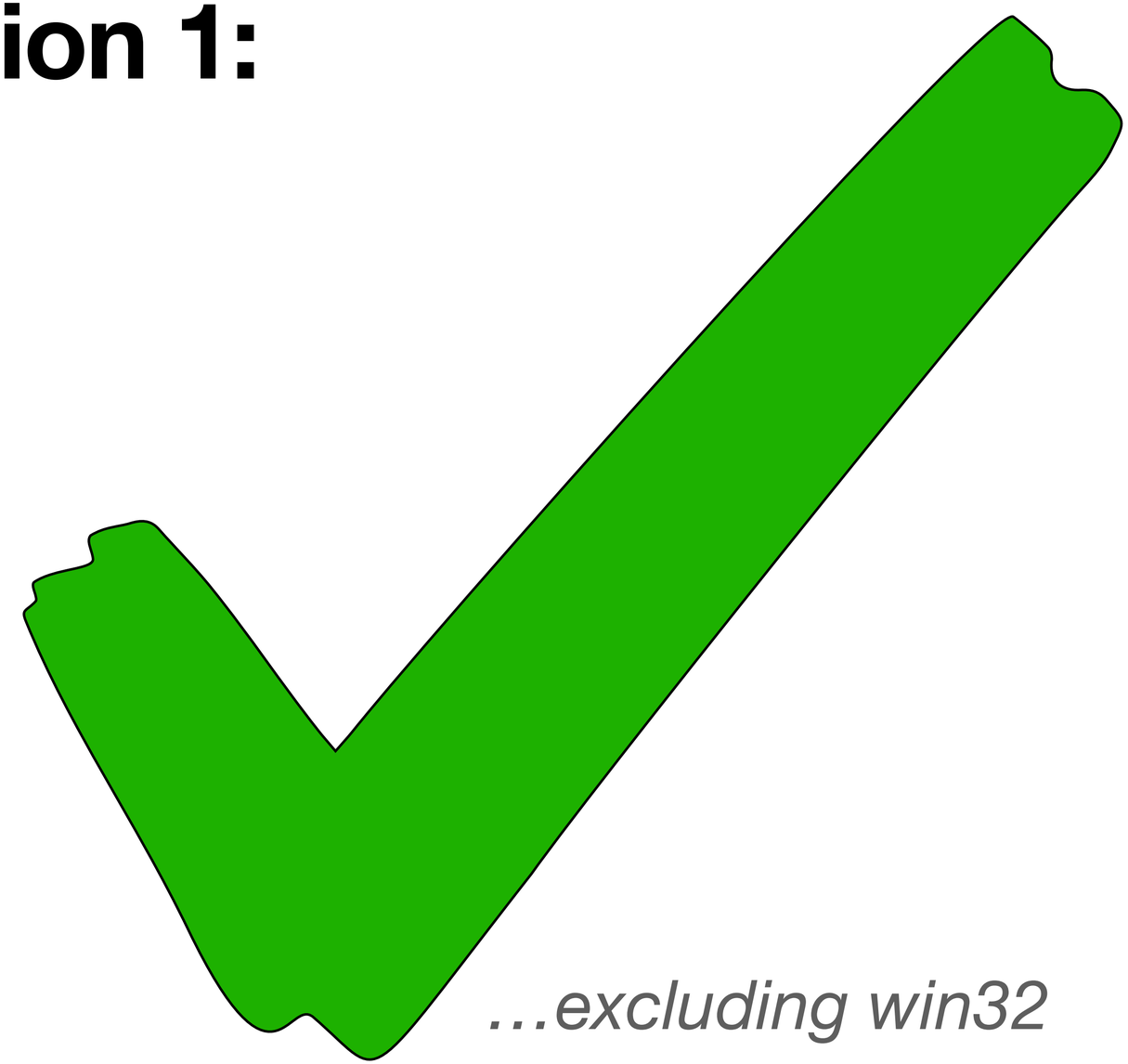
- lock/unlock

`do_mutex_lock_unlock`

Real-time Quiz Question 1:

Is this real-time safe?

```
try
{
}
catch (std::runtime_error)
{
}
```

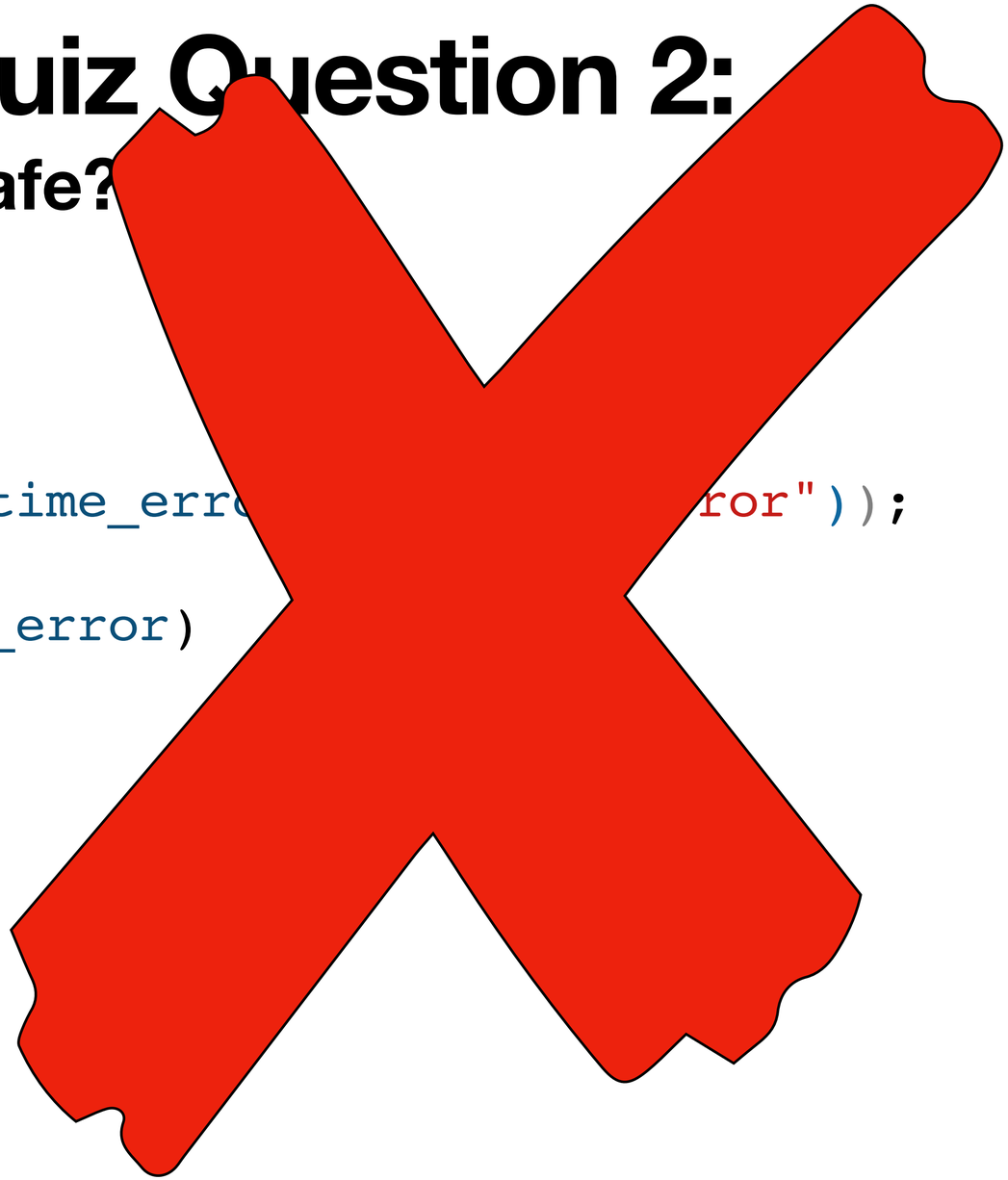


…excluding win32

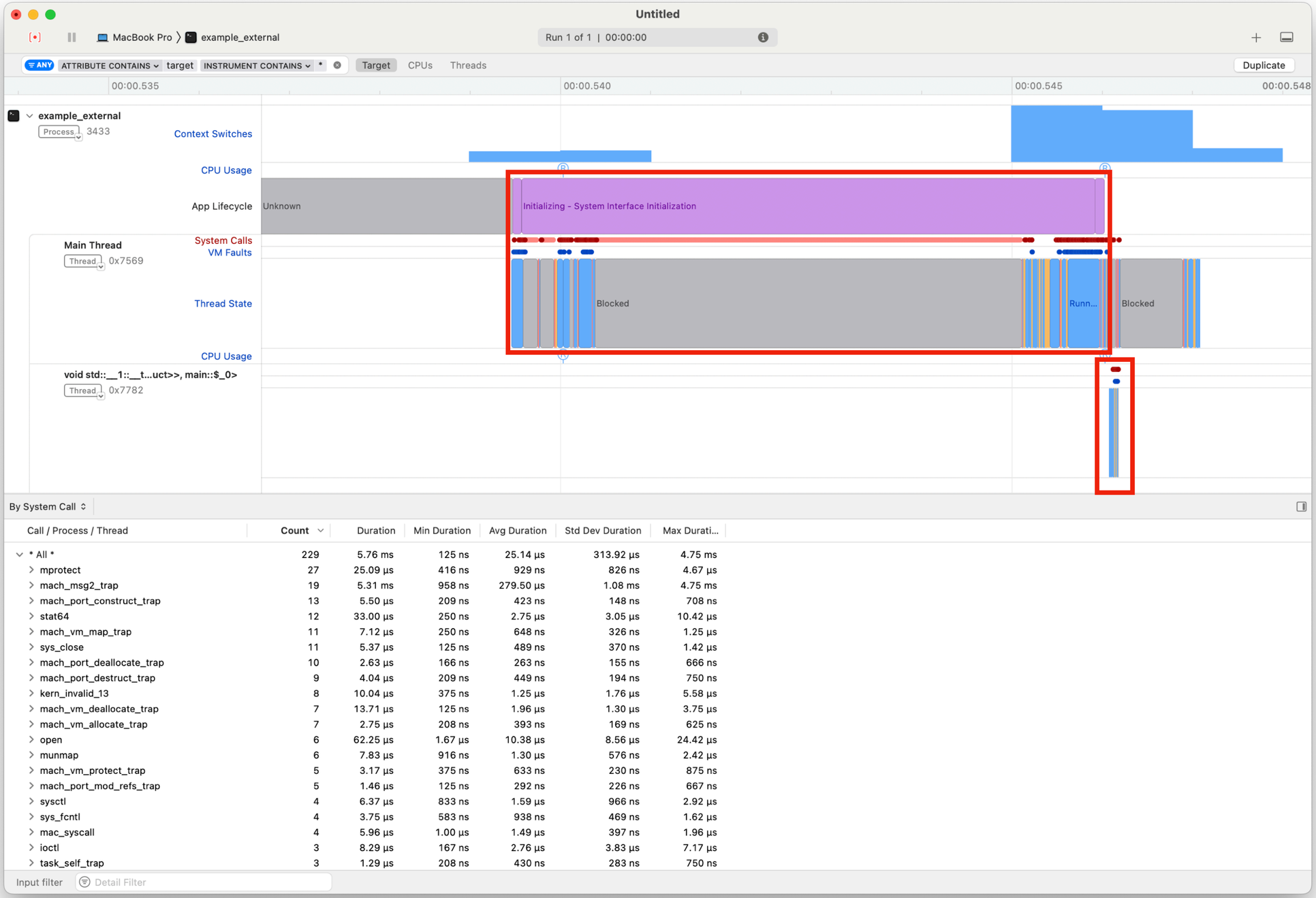
Real-time Quiz Question 2:

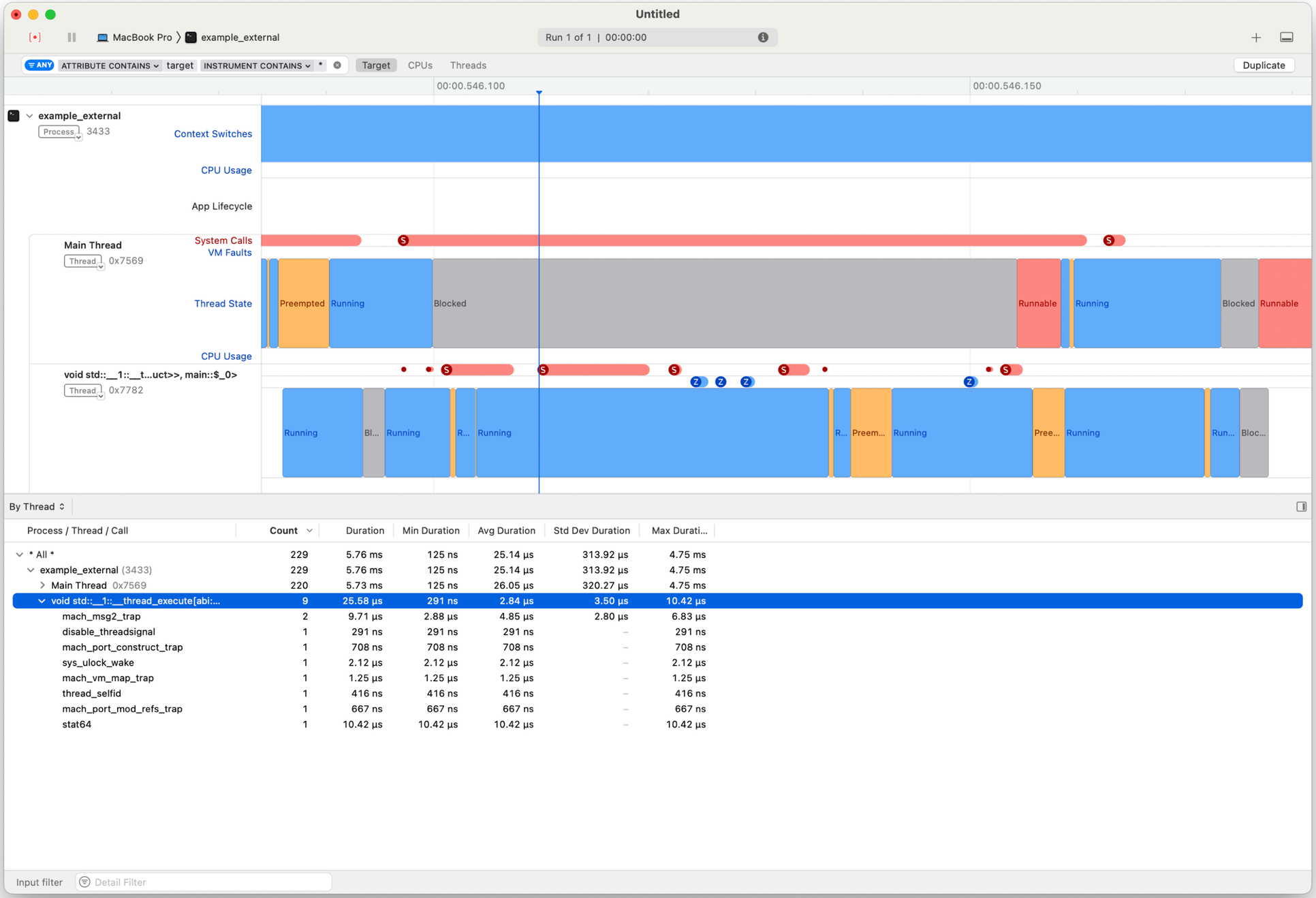
Is this real-time safe?

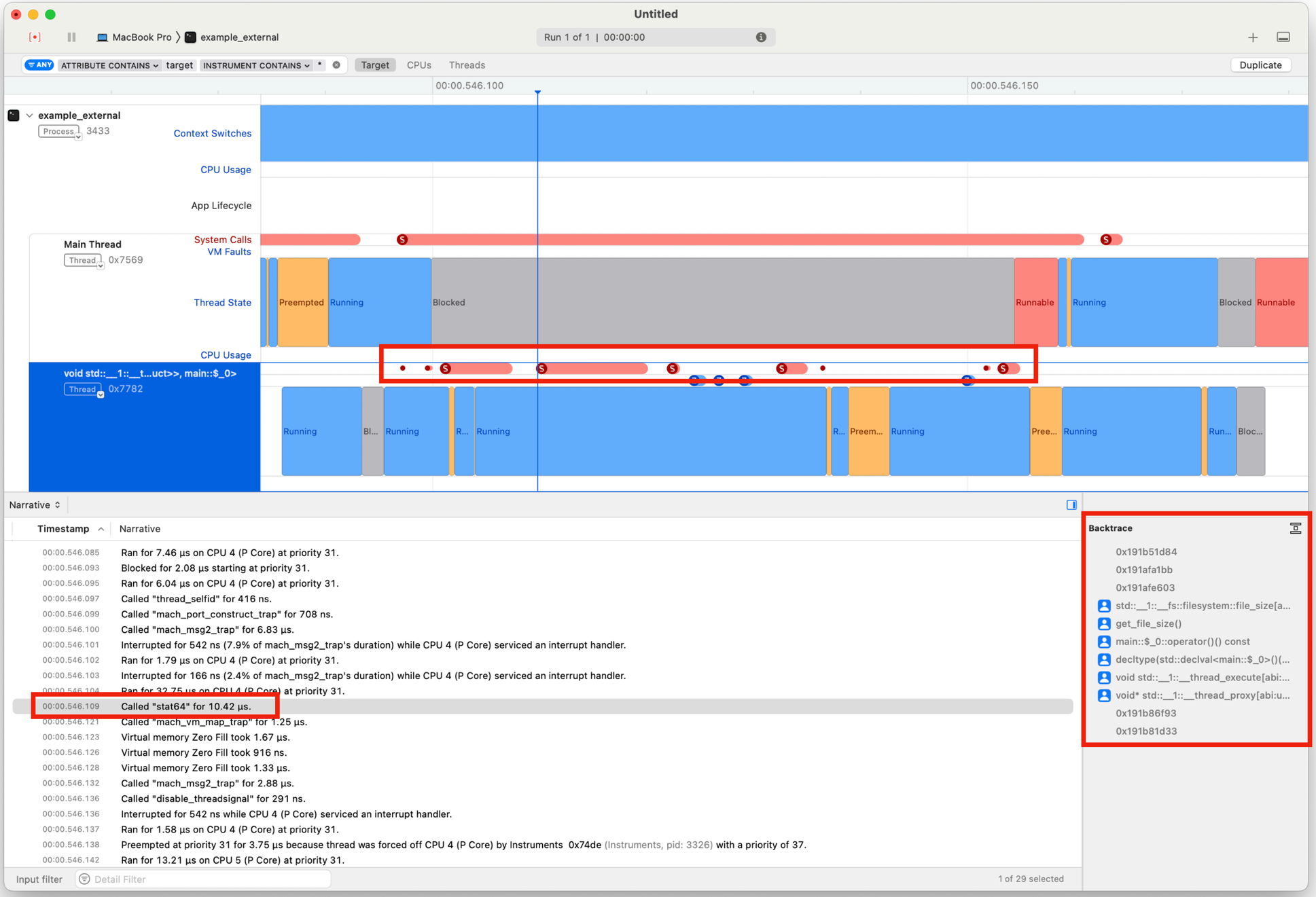
```
try
{
     throw (std::runtime_erro                    ror"));
}
catch (std::runtime_error)
{
}
```



GUI Tools: System Trace - macOS


Untitled
MacBook Pro
example_external
Run 1 of 1 | 00:00:00
ANY
ATTRIBUTE CONTAINS target
INSTRUMENT CONTAINS *
Target
CPUs
Threads
Duplicate
00:00.535
00:00.540
00:00.545
00:00.548
example_external
Process 3433
Context Switches
CPU Usage
App Lifecycle
Unknown
Initializing - System Interface Initialization
Main Thread
Thread 0x7569
System Calls
VM Faults
Thread State
Blocked
Runn...
Blocked
CPU Usage
void std::__1::__t...uct>>, main::$_0>
Thread 0x7782
By System Call
Call / Process / Thread
Count
Duration
Min Duration
Avg Duration
Std Dev Duration
Max Durati...
* All * 229 5.76 ms 125 ns 25.14 µs 313.92 µs 4.75 ms
mprotect 27 25.09 µs 416 ns 929 ns 826 ns 4.67 µs
mach_msg2_trap 19 5.31 ms 958 ns 279.50 µs 1.08 ms 4.75 ms
mach_port_construct_trap 13 5.50 µs 209 ns 423 ns 148 ns 708 ns
stat64 12 33.00 µs 250 ns 2.75 µs 3.05 µs 10.42 µs
mach_vm_map_trap 11 7.12 µs 250 ns 648 ns 326 ns 1.25 µs
sys_close 11 5.37 µs 125 ns 489 ns 370 ns 1.42 µs
mach_port_deallocate_trap 10 2.63 µs 166 ns 263 ns 155 ns 666 ns
mach_port_destruct_trap 9 4.04 µs 209 ns 449 ns 194 ns 750 ns
kern_invalid_13 8 10.04 µs 375 ns 1.25 µs 1.76 µs 5.58 µs
mach_vm_deallocate_trap 7 13.71 µs 125 ns 1.96 µs 1.30 µs 3.75 µs
mach_vm_allocate_trap 7 2.75 µs 208 ns 393 ns 169 ns 625 ns
open 6 62.25 µs 1.67 µs 10.38 µs 8.56 µs 24.42 µs
munmap 6 7.83 µs 916 ns 1.30 µs 576 ns 2.42 µs
mach_vm_protect_trap 5 3.17 µs 375 ns 633 ns 230 ns 875 ns
mach_port_mod_refs_trap 5 1.46 µs 125 ns 292 ns 226 ns 667 ns
sysctl 4 6.37 µs 833 ns 1.59 µs 966 ns 2.92 µs
sys_fcntl 4 3.75 µs 583 ns 938 ns 469 ns 1.62 µs
mac_syscall 4 5.96 µs 1.00 µs 1.49 µs 397 ns 1.96 µs
ioctl 3 8.29 µs 167 ns 2.76 µs 3.83 µs 7.17 µs
task_self_trap 3 1.29 µs 208 ns 430 ns 283 ns 750 ns
Input filter
Detail Filter


Untitled
MacBook Pro
example_external
Run 1 of 1 | 00:00:00
ANY
ATTRIBUTE CONTAINS target
INSTRUMENT CONTAINS *
Target
CPUs
Threads
Duplicate
00:00.546.100
00:00.546.150
example_external
Process 3433
Context Switches
CPU Usage
App Lifecycle
Main Thread
Thread 0x7569
System Calls
VM Faults
Thread State
CPU Usage
Preempted
Running
Blocked
Runnable
Running
Blocked
Runnable
void std::__1::__t...uct>>, main::$_0>
Thread 0x7782
Running
Bl...
Running
R...
Running
R...
Preem...
Running
Pree...
Running
Run...
Bloc...
By Thread
Process / Thread / Call
Count
Duration
Min Duration
Avg Duration
Std Dev Duration
Max Durati...
* All * 229 5.76 ms 125 ns 25.14 µs 313.92 µs 4.75 ms
example_external (3433) 229 5.76 ms 125 ns 25.14 µs 313.92 µs 4.75 ms
Main Thread 0x7569 220 5.73 ms 125 ns 26.05 µs 320.27 µs 4.75 ms
void std::__1::__thread_execute[abi:... 9 25.58 µs 291 ns 2.84 µs 3.50 µs 10.42 µs
mach_msg2_trap 2 9.71 µs 2.88 µs 4.85 µs 2.80 µs 6.83 µs
disable_threadsignal 1 291 ns 291 ns 291 ns – 291 ns
mach_port_construct_trap 1 708 ns 708 ns 708 ns – 708 ns
sys_ulock_wake 1 2.12 µs 2.12 µs 2.12 µs – 2.12 µs
mach_vm_map_trap 1 1.25 µs 1.25 µs 1.25 µs – 1.25 µs
thread_selfid 1 416 ns 416 ns 416 ns – 416 ns
mach_port_mod_refs_trap 1 667 ns 667 ns 667 ns – 667 ns
stat64 1 10.42 µs 10.42 µs 10.42 µs – 10.42 µs
Input filter
Detail Filter


Untitled
MacBook Pro
example_external
Run 1 of 1 | 00:00:00
ANY
ATTRIBUTE CONTAINS target
INSTRUMENT CONTAINS *
Target
CPUs
Threads
Duplicate
00:00.546.100
00:00.546.150
example_external
Process 3433
Context Switches
CPU Usage
App Lifecycle
Main Thread
Thread 0x7569
System Calls
VM Faults
Thread State
Preempted
Running
Blocked
Runnable
Running
Blocked
Runnable
CPU Usage
void std::__1::__t...uct>>, main::$_0>
Thread 0x7782
Running
Bl...
Running
R...
Running
R...
Preem...
Running
Pree...
Running
Run...
Bloc...
Narrative
Timestamp
Narrative
00:00.546.085 Ran for 7.46 µs on CPU 4 (P Core) at priority 31.
00:00.546.093 Blocked for 2.08 µs starting at priority 31.
00:00.546.095 Ran for 6.04 µs on CPU 4 (P Core) at priority 31.
00:00.546.097 Called "thread_selfid" for 416 ns.
00:00.546.099 Called "mach_port_construct_trap" for 708 ns.
00:00.546.100 Called "mach_msg2_trap" for 6.83 µs.
00:00.546.101 Interrupted for 542 ns (7.9% of mach_msg2_trap's duration) while CPU 4 (P Core) serviced an interrupt handler.
00:00.546.102 Ran for 1.79 µs on CPU 4 (P Core) at priority 31.
00:00.546.103 Interrupted for 166 ns (2.4% of mach_msg2_trap's duration) while CPU 4 (P Core) serviced an interrupt handler.
00:00.546.109 Called "stat64" for 10.42 µs.
00:00.546.121 Called "mach_vm_map_trap" for 1.25 µs.
00:00.546.123 Virtual memory Zero Fill took 1.67 µs.
00:00.546.126 Virtual memory Zero Fill took 916 ns.
00:00.546.128 Virtual memory Zero Fill took 1.33 µs.
00:00.546.132 Called "mach_msg2_trap" for 2.88 µs.
00:00.546.136 Called "disable_threadsignal" for 291 ns.
00:00.546.136 Interrupted for 542 ns while CPU 4 (P Core) serviced an interrupt handler.
00:00.546.137 Ran for 1.58 µs on CPU 4 (P Core) at priority 31.
00:00.546.138 Preempted at priority 31 for 3.75 µs because thread was forced off CPU 4 (P Core) by Instruments 0x74de (Instruments, pid: 3326) with a priority of 37.
00:00.546.142 Ran for 13.21 µs on CPU 5 (P Core) at priority 31.
Backtrace
0x191b51d84
0x191afa1bb
0x191afe603
std::__1::__fs::filesystem::file_size[a...
get_file_size()
main::$_0::operator()() const
decltype(std::declval<main::$_0>()(...
void std::__1::__thread_execute[abi:...
void* std::__1::__thread_proxy[abi:u...
0x191b86f93
0x191b81d33
Input filter
Detail Filter
1 of 29 selected

GUI Tools

✓ Easy to run interactively

✓ Easy to view by thread

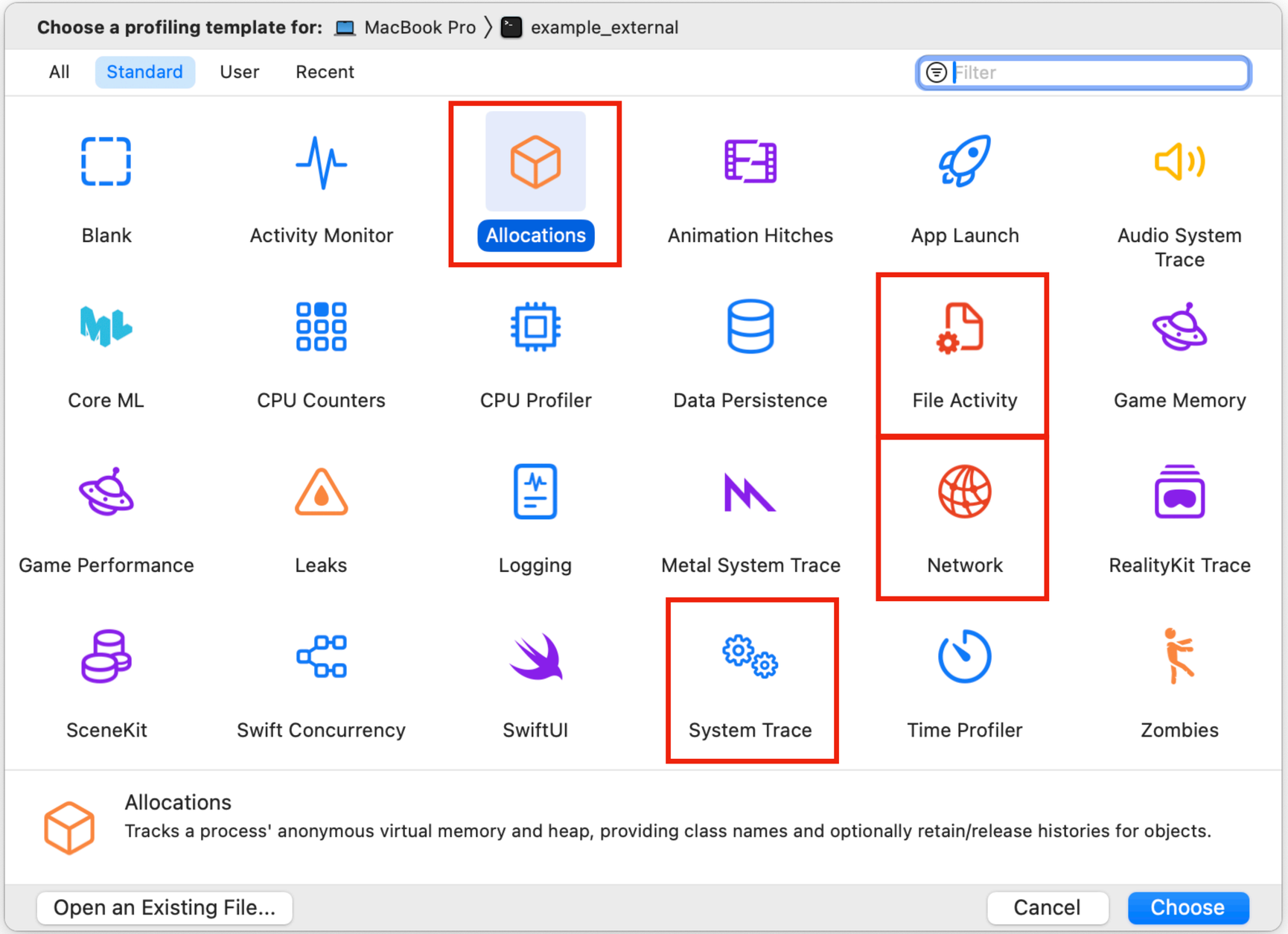
✓ Lots of information
E.g. stack-trace

x Difficult to parse

x No simple filtering

x Can’t run on CI

x Different tools for different probes

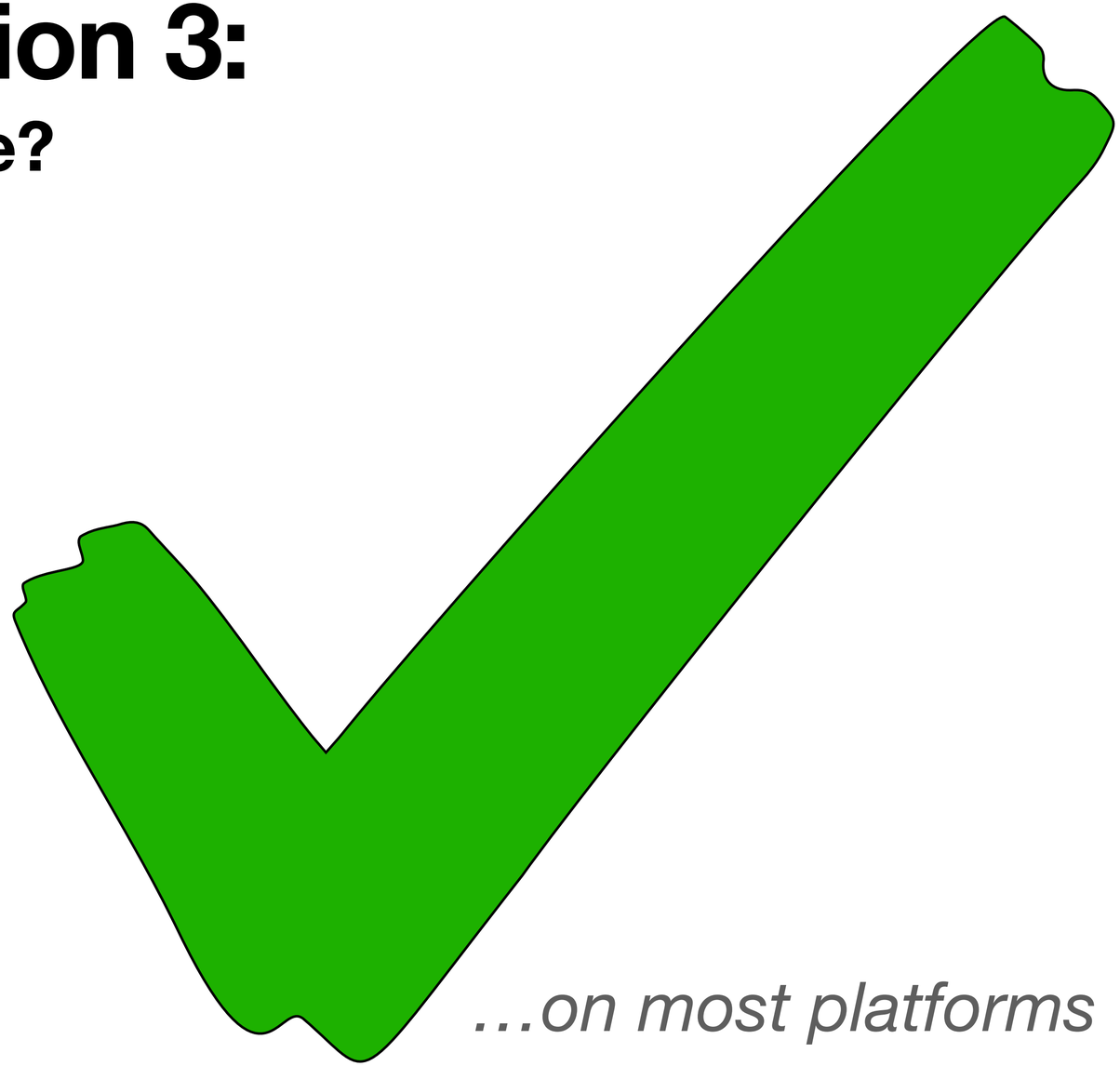

Choose a profiling template for: MacBook Pro example_external
All
Standard
User
Recent
Filter
Blank
Activity Monitor
Allocations
Animation Hitches
App Launch
Audio System Trace
Core ML
CPU Counters
CPU Profiler
Data Persistence
File Activity
Game Memory
Game Performance
Leaks
Logging
Metal System Trace
Network
RealityKit Trace
SceneKit
Swift Concurrency
SwiftUI
System Trace
Time Profiler
Zombies
Allocations
Tracks a process' anonymous virtual memory and heap, providing class names and optionally retain/release histories for objects.
Open an Existing File...
Cancel
Choose

	GUI (System Trace)				
Easy to use?	✅/⚠️				
Clear?	❌				
🧵Filterable?	⚠️				
CI?	❌				
Portable?	❌				
System calls?	✅				
Malloc/free?	⚠️				
Lock/unlock?	✅				
3rd party code?	✅				
Notes	Different tools for different tasks				

Real-time Quiz Question 3:

Is this assignment real-time safe?

```
std::atomic<float> a;
a = 42.0f;
```

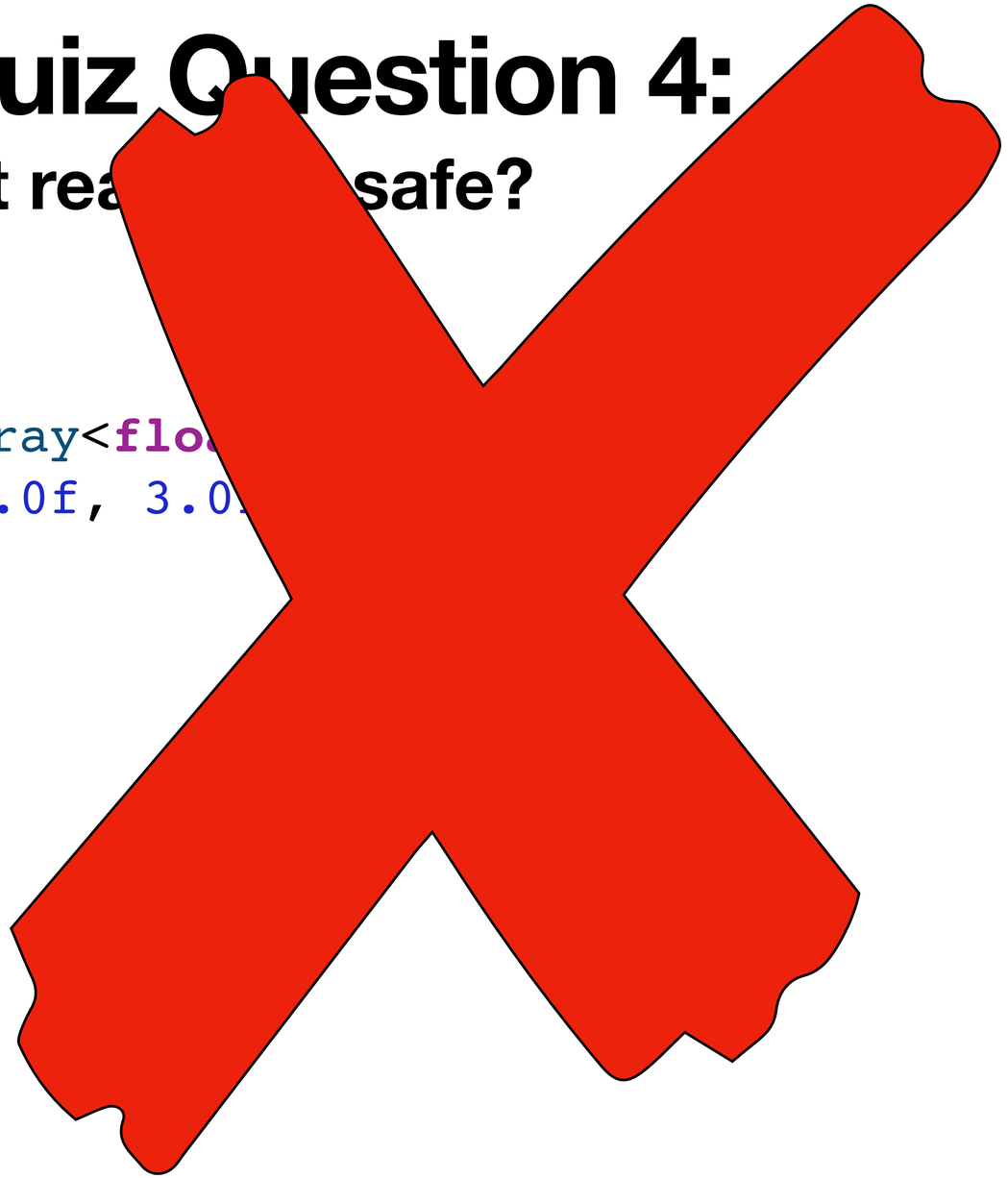


…on most platforms

Real-time Quiz Question 4:

Is this assignment rea safe?

```
std::atomic<std::array<flo
a = { 0.0f, 1.0f, 2.0f, 3.0
```



CLI: strace/ltrace/perf trace (linux)

perf trace

```
     ? (         ): example_extern/25315  ... [continued]: execve())                                    = 0
 0.021 ( 0.003 ms): example_extern/25315 brk()                                                          = 0xaaaaeb303000
 0.092 ( 0.006 ms): example_extern/25315 faccessat(dfd: -100, filename: 0x9ffdb660, mode: 4)            = -1 ENOENT (No such file or directory)
 0.171 ( 0.005 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffd9480, flags: RDONLY|CLOEXEC)  = 3
 0.192 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                   = 0
 0.202 ( 0.005 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffeb140, flags: RDONLY|CLOEXEC)  = 3
 0.210 ( 0.004 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59feaa0, count: 832)                   = 832
 0.239 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fd6b000, len: 20480)                       = 0
 0.245 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9ff9a000, len: 43040)                       = 0
 0.251 ( 0.006 ms): example_extern/25315 mprotect(start: 0xffff9ff7a000, len: 61440)                    = 0
 0.279 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                   = 0
 0.288 ( 0.004 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffeb680, flags: RDONLY|CLOEXEC)  = 3
 0.294 ( 0.004 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59fea80, count: 832)                   = 832
 0.320 ( 0.009 ms): example_extern/25315 munmap(addr: 0xffff9fd3b000, len: 20480)                       = 0
 0.332 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fd65000, len: 41672)                       = 0
 0.338 ( 0.009 ms): example_extern/25315 mprotect(start: 0xffff9fd54000, len: 61440)                    = 0
 0.361 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                   = 0
 0.367 ( 0.004 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffebbc0, flags: RDONLY|CLOEXEC)  = 3
 0.374 ( 0.003 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59fea60, count: 832)                   = 832
 0.402 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fb87000, len: 36864)                       = 0
 0.407 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fd39000, len: 28264)                       = 0
 0.413 ( 0.005 ms): example_extern/25315 mprotect(start: 0xffff9fd18000, len: 61440)                    = 0
 0.490 ( 0.006 ms): example_extern/25315 close(fd: 3)                                                   = 0
 0.521 ( 0.009 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffec100, flags: RDONLY|CLOEXEC)  = 3
 0.541 ( 0.003 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59fea20, count: 832)                   = 832
 0.576 ( 0.004 ms): example_extern/25315 munmap(addr: 0xffff9fae9000, len: 28672)                       = 0
 0.590 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fb87000, len: 32880)                       = 0
 0.602 ( 0.011 ms): example_extern/25315 mprotect(start: 0xffff9fb76000, len: 61440)                    = 0
 0.666 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                   = 0
 0.697 ( 0.005 ms): example_extern/25315 set_tid_address(tidptr: 0xffff9ffe9af0)                        = 25315 (example_externa)
 0.710 ( 0.003 ms): example_extern/25315 set_robust_list(head: 0xffff9ffe9b00, len: 24)                 = 0
 0.727 ( 0.003 ms): example_extern/25315 rseq(rseq: 0xffff9ffea1c0, rseq_len: 32, sig: 3559439360)      = 0
 0.781 ( 0.005 ms): example_extern/25315 mprotect(start: 0xffff9fd27000, len: 16384, prot: READ)        = 0
 0.791 ( 0.004 ms): example_extern/25315 mprotect(start: 0xffff9fb85000, len: 4096, prot: READ)         = 0
 0.799 ( 0.004 ms): example_extern/25315 mprotect(start: 0xffff9fd63000, len: 4096, prot: READ)         = 0
 1.794 ( 0.023 ms): example_extern/25315 mprotect(start: 0xffff9ff89000, len: 45056, prot: READ)        = 0
 1.874 ( 0.012 ms): example_extern/25315 mprotect(start: 0xaaaaae426000, len: 4096, prot: READ)         = 0
 1.897 ( 0.012 ms): example_extern/25315 mprotect(start: 0xffff9fff0000, len: 8192, prot: READ)         = 0
 1.953 ( 0.003 ms): example_extern/25315 prlimit64(resource: STACK, old_rlim: 0xffffd59ff4b8)           = 0
 1.983 ( 0.013 ms): example_extern/25315 munmap(addr: 0xffff9ffa5000, len: 67899)                       = 0
 2.053 ( 0.009 ms): example_extern/25315 getrandom(ubuf: 0xffff9fd31930, len: 8, flags: NONBLOCK)       = 8
 2.071 ( 0.003 ms): example_extern/25315 brk()                                                          = 0xaaaaeb303000
 2.083 ( 0.009 ms): example_extern/25315 brk(brk: 0xaaaaeb324000)                                       = 0xaaaaeb324000
 2.321 ( 0.004 ms): example_extern/25315 futex(uaddr: 0xffff9ff977a4, op: WAKE|PRIVATE_FLAG, val: 2147483647)  = 0
 2.489 ( 0.003 ms): example_extern/25315 rt_sigaction(sig: 0x21, act: 0xffffd59ff478, sigsetsize: 8)        = 0
 2.495 ( 0.003 ms): example_extern/25315 rt_sigprocmask(how: UNBLOCK, nset: 0xffffd59ff6a8, sigsetsize: 8)     = 0
 2.507 ( 0.004 ms): example_extern/25315 mprotect(start: 0xffff9f2f0000, len: 8388608, prot: READ|WRITE)      = 0
 2.525 ( 0.004 ms): example_extern/25315 rt_sigprocmask(how: BLOCK, nset: 0xffff9fcd82c8, oset: 0xffffd59ff6a0, sigsetsize: 8) = 0
 2.544 ( 0.096 ms): example_extern/25315 clone(clone_flags: VM|FS|FILES|SIGHAND|THREAD|SYSVSEM|SETTLS|PARENT_SETTID|CHILD_CLEARTID, newsp: 0xffff9faee940, parent_tidptr: 0xffff9faef1d0, tls: 0xffff9faef8c0, chil
 2.647 ( 0.003 ms): example_extern/25315 rt_sigprocmask(how: SETMASK, nset: 0xffffd59ff6a0, sigsetsize: 8)     = 0
 2.676 (         ): example_extern/25315 futex(uaddr: 0xffff9faef1d0, op: WAIT_BITSET|CLOCK_REALTIME, val: 25316, val3: MATCH_ANY) ...
 2.689 ( 0.005 ms): example_extern/25316 rseq(rseq: 0xffff9faef8a0, rseq_len: 32, sig: 3559439360)      = 0
 2.702 ( 0.003 ms): example_extern/25316 set_robust_list(head: 0xffff9faef1e0, len: 24)                 = 0
 2.713 ( 0.006 ms): example_extern/25316 rt_sigprocmask(how: SETMASK, nset: 0xffff9faef7f0, sigsetsize: 8)     = 0
 2.728 ( 0.006 ms): example_extern/25316 sched_get_priority_max(policy: 1)                              = 99
 2.742 ( 0.008 ms): example_extern/25316 sched_setscheduler(pid: 25316 (example_externa), policy: FIFO, param: 0xffff9faee6c0) = 0
 2.772 ( 0.004 ms): example_extern/25316 munmap(addr: 0xffff972e0000, len: 13762560)                    = 0
 2.779 ( 0.003 ms): example_extern/25316 munmap(addr: 0xffff9c000000, len: 53346304)                    = 0
 2.785 ( 0.003 ms): example_extern/25316 mprotect(start: 0xffff98000000, len: 135168, prot: READ|WRITE)       = 0
 2.808 ( 0.003 ms): example_extern/25316 sched_get_priority_min(policy: 1)                              = 1
 2.814 ( 0.004 ms): example_extern/25316 sched_setscheduler(pid: 25316 (example_externa), policy: FIFO, param: 0xffff9faee6c0) = 0
 2.823 ( 0.003 ms): example_extern/25316 rt_sigprocmask(how: BLOCK, nset: 0xffff9faef7f0, sigsetsize: 8)       = 0
 2.829 ( 0.005 ms): example_extern/25316 madvise(start: 0xffff9f2e0000, len_in: 8314880, behavior: MADV_DONTNEED) = 0
 2.837 (         ): example_extern/25316 exit()                                                         = ?
 2.676 ( 0.177 ms): example_extern/25315  ... [continued]: futex())                                     = 0
 2.937 (         ): example_extern/25315 exit_group()                                                   = ?
```

perf trace

```
     ? (         ): example_extern/25315  ... [continued]: execve())                                    = 0
 0.021 ( 0.003 ms): example_extern/25315 brk()                                                          = 0xaaaaeb303000
 0.092 ( 0.006 ms): example_extern/25315 faccessat(dfd: -100, filename: 0x9ffdb660, mode: 4)            = -1 ENOENT (No such file or directory)
 0.171 ( 0.005 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffd9480, flags: RDONLY|CLOEXEC)   = 3
 0.192 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                   = 0
 0.202 ( 0.005 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffeb140, flags: RDONLY|CLOEXEC)   = 3
 0.210 ( 0.004 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59feaa0, count: 832)                   = 832
 0.239 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fd6b000, len: 20480)                       = 0
 0.245 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9ff9a000, len: 43040)                       = 0
 0.251 ( 0.006 ms): example_extern/25315 mprotect(start: 0xffff9ff7a000, len: 61440)                    = 0
 0.279 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                   = 0
 0.288 ( 0.004 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffeb680, flags: RDONLY|CLOEXEC)   = 3
 0.294 ( 0.004 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59fea80, count: 832)                   = 832
 0.320 ( 0.009 ms): example_extern/25315 munmap(addr: 0xffff9fd3b000, len: 20480)                       = 0
 0.332 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fd65000, len: 41672)                       = 0
 0.338 ( 0.009 ms): example_extern/25315 mprotect(start: 0xffff9fd54000, len: 61440)                    = 0
 0.361 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                   = 0
 0.367 ( 0.004 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffebbc0, flags: RDONLY|CLOEXEC)   = 3
 0.374 ( 0.003 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59fea60, count: 832)                   = 832
 0.402 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fb87000, len: 36864)                       = 0
 0.407 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fd39000, len: 28264)                       = 0
 0.413 ( 0.005 ms): example_extern/25315 mprotect(start: 0xffff9fd18000, len: 61440)                    = 0
 0.490 ( 0.006 ms): example_extern/25315 close(fd: 3)                                                   = 0
 0.521 ( 0.009 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffec100, flags: RDONLY|CLOEXEC)   = 3
 0.541 ( 0.003 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59fea20, count: 832)                   = 832
 0.576 ( 0.004 ms): example_extern/25315 munmap(addr: 0xffff9fae9000, len: 28672)                       = 0
 0.590 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fb87000, len: 32880)                       = 0
 0.602 ( 0.011 ms): example_extern/25315 mprotect(start: 0xffff9fb76000, len: 61440)                    = 0
 0.666 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                   = 0
 0.697 ( 0.005 ms): example_extern/25315 set_tid_address(tidptr: 0xffff9ffe9af0)                        = 25315 (example_externa)
 0.710 ( 0.003 ms): example_extern/25315 set_robust_list(head: 0xffff9ffe9b00, len: 24)                 = 0
 0.727 ( 0.003 ms): example_extern/25315 rseq(rseq: 0xffff9ffea1c0, rseq_len: 32, sig: 3559439360)      = 0
 0.781 ( 0.005 ms): example_extern/25315 mprotect(start: 0xffff9fd27000, len: 16384, prot: READ)        = 0
 0.791 ( 0.004 ms): example_extern/25315 mprotect(start: 0xffff9fb85000, len: 4096, prot: READ)         = 0
 0.799 ( 0.004 ms): example_extern/25315 mprotect(start: 0xffff9fd63000, len: 4096, prot: READ)         = 0
 1.794 ( 0.023 ms): example_extern/25315 mprotect(start: 0xffff9ff89000, len: 45056, prot: READ)        = 0
 1.874 ( 0.012 ms): example_extern/25315 mprotect(start: 0xaaaaae426000, len: 4096, prot: READ)         = 0
 1.897 ( 0.012 ms): example_extern/25315 mprotect(start: 0xffff9fff0000, len: 8192, prot: READ)         = 0
 1.953 ( 0.003 ms): example_extern/25315 prlimit64(resource: STACK, old_rlim: 0xffffd59ff4b8)           = 0
 1.983 ( 0.013 ms): example_extern/25315 munmap(addr: 0xffff9ffa5000, len: 67899)                       = 0
 2.053 ( 0.009 ms): example_extern/25315 getrandom(ubuf: 0xffff9fd31930, len: 8, flags: NONBLOCK)       = 8
 2.071 ( 0.003 ms): example_extern/25315 brk()                                                          = 0xaaaaeb303000
 2.083 ( 0.009 ms): example_extern/25315 brk(brk: 0xaaaaeb324000)                                       = 0xaaaaeb324000
 2.321 ( 0.004 ms): example_extern/25315 futex(uaddr: 0xffff9ff977a4, op: WAKE|PRIVATE_FLAG, val: 2147483647)  = 0
 2.489 ( 0.003 ms): example_extern/25315 rt_sigaction(sig: 0x21, act: 0xffffd59ff478, sigsetsize: 8)    = 0
 2.495 ( 0.003 ms): example_extern/25315 rt_sigprocmask(how: UNBLOCK, nset: 0xffffd59ff6a8, sigsetsize: 8)  = 0
 2.507 ( 0.004 ms): example_extern/25315 mprotect(start: 0xffff9f2f0000, len: 8388608, prot: READ|WRITE)  = 0
 2.525 ( 0.004 ms): example_extern/25315 rt_sigprocmask(how: BLOCK, nset: 0xffff9fcd82c8, oset: 0xffffd59ff6a0, sigsetsize: 8) = 0
 2.544 ( 0.096 ms): example_extern/25315 clone(clone_flags: VM|FS|FILES|SIGHAND|THREAD|SYSVSEM|SETTLS|PARENT_SETTID|CHILD_CLEARTID, newsp: 0xffff9faee940, parent_tidptr: 0xffff9faef1d0, tls: 0xffff9faef8c0, chil
 2.647 ( 0.003 ms): example_extern/25315 rt_sigprocmask(how: SETMASK, nset: 0xffffd59ff6a0, sigsetsize: 8)  = 0
 2.676 (         ): example_extern/25315 futex(uaddr: 0xffff9faef1d0, op: WAIT_BITSET|CLOCK_REALTIME, val: 25316, val3: MATCH_ANY) ...
 2.689 ( 0.005 ms): example_extern/25316 rseq(rseq: 0xffff9faef8a0, rseq_len: 32, sig: 3559439360)      = 0
 2.702 ( 0.003 ms): example_extern/25316 set_robust_list(head: 0xffff9faef1e0, len: 24)                 = 0
 2.713 ( 0.006 ms): example_extern/25316 rt_sigprocmask(how: SETMASK, nset: 0xffff9faef7f0, sigsetsize: 8)  = 0
 2.728 ( 0.006 ms): example_extern/25316 sched_get_priority_max(policy: 1)                              = 99
 2.742 ( 0.008 ms): example_extern/25316 sched_setscheduler(pid: 25316 (example_externa), policy: FIFO, param: 0xffff9faee6c0) = 0
 2.772 ( 0.004 ms): example_extern/25316 munmap(addr: 0xffff972e0000, len: 13762560)                    = 0
 2.779 ( 0.003 ms): example_extern/25316 munmap(addr: 0xffff9c000000, len: 53346304)                    = 0
 2.785 ( 0.003 ms): example_extern/25316 mprotect(start: 0xffff98000000, len: 135168, prot: READ|WRITE)  = 0
 2.808 ( 0.003 ms): example_extern/25316 sched_get_priority_min(policy: 1)                              = 1
 2.814 ( 0.004 ms): example_extern/25316 sched_setscheduler(pid: 25316 (example_externa), policy: FIFO, param: 0xffff9faee6c0) = 0
 2.823 ( 0.003 ms): example_extern/25316 rt_sigprocmask(how: BLOCK, nset: 0xffff9faef7f0, sigsetsize: 8)  = 0
 2.829 ( 0.005 ms): example_extern/25316 madvise(start: 0xffff9f2e0000, len_in: 8314880, behavior: MADV_DONTNEED) = 0
 2.837 (         ): example_extern/25316 exit()                                                         = ?
 2.676 ( 0.177 ms): example_extern/25315  ... [continued]: futex())                                     = 0
 2.937 (         ): example_extern/25315 exit_group()                                                   = ?
```

strace

```
execve("./example_external", ["./example_external"], 0xfffff35bb480 /* 17 vars */) = 0
brk(NULL)                               = 0xaaaad04d8000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffaed86000
faccessat(AT_FDCWD, "/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=67899, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 67899, PROT_READ, MAP_PRIVATE, 3, 0) = 0xffffaed40000
close(3)                                = 0
openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libstdc++.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0\267\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2190752, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 2332704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffaeb06000
mmap(0xffffaeb10000, 2267168, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffffaeb10000
munmap(0xffffaeb06000, 40960)           = 0
munmap(0xffffaed3a000, 22560)           = 0
mprotect(0xffffaed1a000, 61440, PROT_NONE) = 0
mmap(0xffffaed29000, 57344, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x209000) = 0xffffaed29000
mmap(0xffffaed37000, 10272, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0xffffaed37000
close(3)                                = 0
openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libgcc_s.so.1", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\3\0\267\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=84296, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 213704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffaeadb000
mmap(0xffffaeae0000, 148168, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffffaeae0000
munmap(0xffffaeadb000, 20480)           = 0
munmap(0xffffaeb05000, 41672)           = 0
mprotect(0xffffaeaf4000, 61440, PROT_NONE) = 0
mmap(0xffffaeb03000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x13000) = 0xffffaeb03000
close(3)                                = 0
openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0\267\0\1\0\0\0\340u\2\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=1637400, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 1805928, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffae927000
mmap(0xffffae930000, 1740392, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffffae930000
munmap(0xffffae927000, 36864)           = 0
munmap(0xffffaead9000, 28264)           = 0
mprotect(0xffffaeab8000, 61440, PROT_NONE) = 0
mmap(0xffffaeac7000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x187000) = 0xffffaeac7000
mmap(0xffffaeacd000, 48744, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0xffffaeacd000
close(3)                                = 0
openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libm.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\3\0\267\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=551064, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 680048, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffae889000
mmap(0xffffae890000, 614512, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffffae890000
munmap(0xffffae889000, 28672)           = 0
munmap(0xffffae927000, 32880)           = 0
mprotect(0xffffae916000, 61440, PROT_NONE) = 0
mmap(0xffffae925000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x85000) = 0xffffae925000
close(3)                                = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffaed84000
set_tid_address(0xffffaed84af0)         = 25322
set_robust_list(0xffffaed84b00, 24)     = 0
rseq(0xffffaed851c0, 0x20, 0, 0xd428bc00) = 0
mprotect(0xffffaeac7000, 16384, PROT_READ) = 0
mprotect(0xffffae925000, 4096, PROT_READ) = 0
mprotect(0xffffaeb03000, 4096, PROT_READ) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffaed82000
mprotect(0xffffaed29000, 45056, PROT_READ) = 0
mprotect(0xaaaac3096000, 4096, PROT_READ) = 0
mprotect(0xffffaed8b000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0xffffaed40000, 67899)           = 0
getrandom("\x58\x27\x16\xb8\xfd\xe5\x40\x03", 8, GRND_NONBLOCK) = 8
brk(NULL)                               = 0xaaaad04d8000
brk(0xaaaad04f9000)                     = 0xaaaad04f9000
futex(0xffffaed377a4, FUTEX_WAKE_PRIVATE, 2147483647) = 0
rt_sigaction(SIGRT_1, {sa_handler=0xffffae9aa700, sa_mask=[], sa_flags=SA_ONSTACK|SA_RESTART|SA_SIGINFO}, NULL, 8) = 0
rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8) = 0
mmap(NULL, 8454144, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0xffffae080000
mprotect(0xffffae090000, 8388608, PROT_READ|PROT_WRITE) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)  = 0
clone(child_stack=0xffffae88e940, flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[25323], tls=0xffffae88f8c0, child_tidptr=0xffffae88f1d0) = 25323
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
exit_group(0)                           = ?
```

ltrace -e pthread*+malloc+free

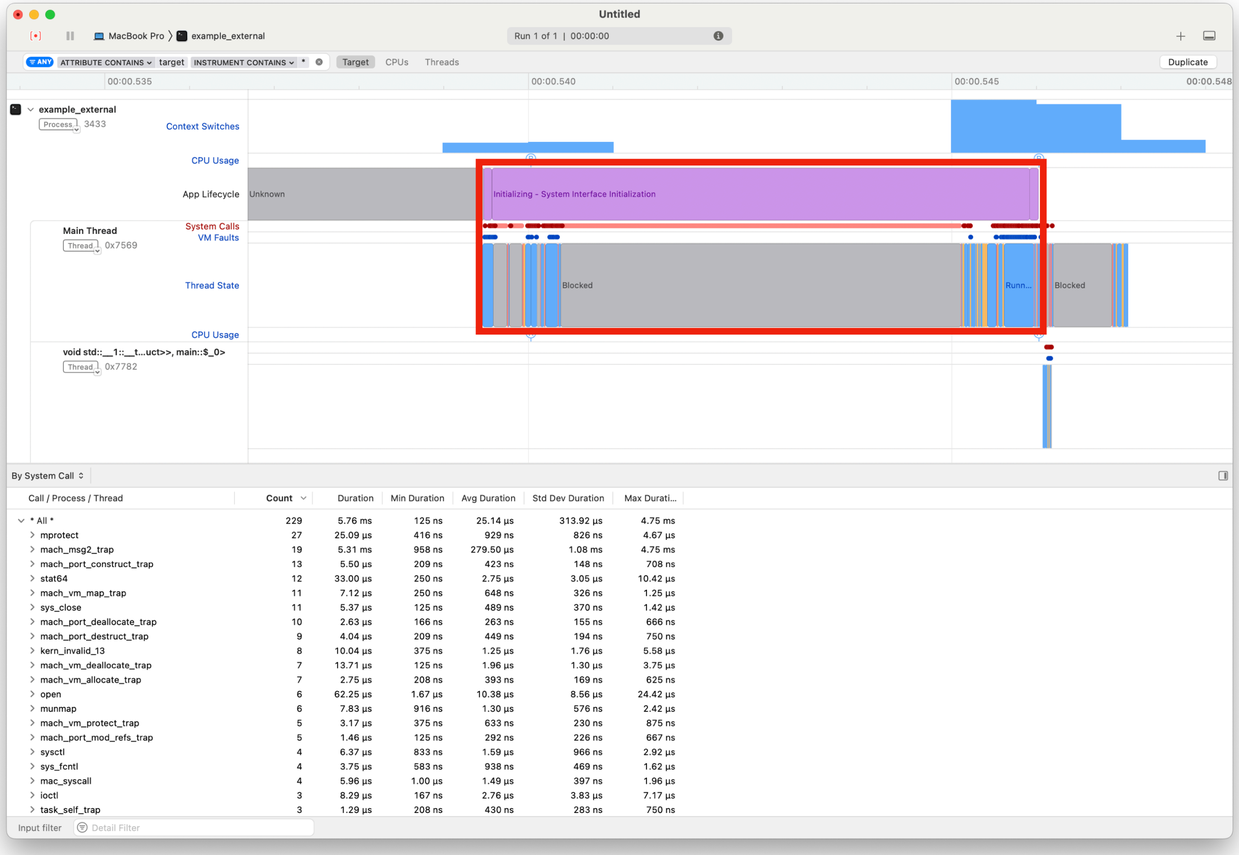
```
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0, 1)                    = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8878740, 0xffffb8878779) = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 1, 0xffffb8870af0)       = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 1, 1)                    = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8878610, 0xffffb8870e78) = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb88785f0, 0xffffb8870e78) = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb88780b0, 0xffffb860a9d0) = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 1, 0xffffb8871fb0)       = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 1, 1)                    = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8877f80, 0xffffb8872338) = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8879540, 5)       = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 38, 1)                   = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8879470, 4)       = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8878b00, 46)      = 0
libstdc++.so.6->malloc(16)                                                            = 0xaaaadda66eb0
libstdc++.so.6->pthread_create(0xffffd90c9760, 0, 0xffffb87231e0, 0xaaaadda66eb0)     = 0
libstdc++.so.6->pthread_join(0xffffb83cf100, 0, 0, 8)
```

ltrace -e pthread*+malloc+free

```
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0, 1)                    = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8878740, 0xffffb8878779) = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 1, 0xffffb8870af0)       = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 1, 1)                    = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8878610, 0xffffb8870e78) = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb88785f0, 0xffffb8870e78) = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb88780b0, 0xffffb860a9d0) = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 1, 0xffffb8871fb0)       = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 1, 1)                    = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8877f80, 0xffffb8872338) = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8879540, 5)       = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 38, 1)                   = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8879470, 4)       = 0
libstdc++.so.6->pthread_once(0xffffb88777a4, 0xffffb8705c30, 0xffffb8878b00, 46)      = 0
libstdc++.so.6->malloc(16)                                                            = 0xaaaadda66eb0
libstdc++.so.6->pthread_create(0xffffd90c9760, 0, 0xffffb87231e0, 0xaaaadda66eb0)     = 0
libstdc++.so.6->pthread_join(0xffffb83cf100, 0, 0, 8)
```

strace

```
execve("./example_external", ["./example_external"], 0xfffff35bb480 /* 17 vars */) = 0
brk(NULL)                               = 0xaaaad04d8000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffaed86000
faccessat(AT_FDCWD, "/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=67899, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 67899, PROT_READ, MAP_PRIVATE, 3, 0) = 0xffffaed40000
close(3)                                = 0
openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libstdc++.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0\267\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2190752, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 2332704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffaeb06000
mmap(0xffffaeb10000, 2267168, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffffaeb10000
munmap(0xffffaeb06000, 40960)           = 0
munmap(0xffffaed3a000, 22560)           = 0
mprotect(0xffffaed1a000, 61440, PROT_NONE) = 0
mmap(0xffffaed29000, 57344, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x209000) = 0xffffaed29000
mmap(0xffffaed37000, 10272, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0xffffaed37000
close(3)                                = 0
openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libgcc_s.so.1", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\3\0\267\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=84296, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 213704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffaeadb000
mmap(0xffffaeae0000, 148168, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffffaeae0000
munmap(0xffffaeadb000, 20480)           = 0
munmap(0xffffaeb05000, 41672)           = 0
mprotect(0xffffaeaf4000, 61440, PROT_NONE) = 0
mmap(0xffffaeb03000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x13000) = 0xffffaeb03000
close(3)                                = 0
openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0\267\0\1\0\0\0\340u\2\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=1637400, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 1805928, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffae927000
mmap(0xffffae930000, 1740392, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffffae930000
munmap(0xffffae927000, 36864)           = 0
munmap(0xffffaead9000, 28264)           = 0
mprotect(0xffffaeab8000, 61440, PROT_NONE) = 0
mmap(0xffffaeac7000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x187000) = 0xffffaeac7000
mmap(0xffffaeacd000, 48744, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0xffffaeacd000
close(3)                                = 0
openat(AT_FDCWD, "/lib/aarch64-linux-gnu/libm.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\3\0\267\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=551064, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 680048, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffae889000
mmap(0xffffae890000, 614512, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0) = 0xffffae890000
munmap(0xffffae889000, 28672)           = 0
munmap(0xffffae927000, 32880)           = 0
mprotect(0xffffae916000, 61440, PROT_NONE) = 0
mmap(0xffffae925000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x85000) = 0xffffae925000
close(3)                                = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffaed84000
set_tid_address(0xffffaed84af0)         = 25322
set_robust_list(0xffffaed84b00, 24)     = 0
rseq(0xffffaed851c0, 0x20, 0, 0xd428bc00) = 0
mprotect(0xffffaeac7000, 16384, PROT_READ) = 0
mprotect(0xffffae925000, 4096, PROT_READ) = 0
mprotect(0xffffaeb03000, 4096, PROT_READ) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xffffaed82000
mprotect(0xffffaed29000, 45056, PROT_READ) = 0
mprotect(0xaaaac3096000, 4096, PROT_READ) = 0
mprotect(0xffffaed8b000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0xffffaed40000, 67899)           = 0
getrandom("\x58\x27\x16\xb8\xfd\xe5\x40\x03", 8, GRND_NONBLOCK) = 8
brk(NULL)                               = 0xaaaad04d8000
brk(0xaaaad04f9000)                     = 0xaaaad04f9000
futex(0xffffaed377a4, FUTEX_WAKE_PRIVATE, 2147483647) = 0
rt_sigaction(SIGRT_1, {sa_handler=0xffffae9aa700, sa_mask=[], sa_flags=SA_ONSTACK|SA_RESTART|SA_SIGINFO}, NULL, 8) = 0
rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8) = 0
mmap(NULL, 8454144, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0xffffae080000
mprotect(0xffffae090000, 8388608, PROT_READ|PROT_WRITE) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0xffffae88e940, flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[25323], tls=0xffffae88f8c0, child_tidptr=0xffffae88f1d0) = 25323
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
exit_group(0)                           = ?
```



CLI Tools: strace/ltrace/perf (linux)

✓ Easy to run

✓ Lots of information

x No simple thread separation

x Verbose defaults

x Different tools for different probes

CLI: dtrace (macOS)

```
sudo dtrace -c $BUILD_DIR/example -s rt_check.d
```

rt_check.d

```
pid$target::syscall:entry,

pid$target::malloc:entry,
pid$target::free:entry,

pid$target::pthread_*:entry,

pid$target::stat*:entry,
pid$target::stat64*:entry,
pid$target::fstat*:entry,
pid$target::fstat64*:entry,
pid$target::*write*:entry,
pid$target::read*:entry
{
}
```

```
CPU     ID                    FUNCTION:NAME
  4 472662         mach_msg_overwrite:entry
  4 472441                     malloc:entry
  4 472441                     malloc:entry
  4 472441                     malloc:entry
  4 472480              pthread_create:entry
  4 472519                pthread_join:entry
  4 472484         pthread_testcancel:entry
  6 472453         pthread_mutex_lock:entry
  6 472461               pthread_self:entry
  6 472462     pthread_mach_thread_np:entry
  6 472454       pthread_mutex_unlock:entry
  6 472452         pthread_key_create:entry
  6 472453         pthread_mutex_lock:entry
  6 472454       pthread_mutex_unlock:entry
  6 472456        pthread_setspecific:entry
  6 472453         pthread_mutex_lock:entry
  6 472461               pthread_self:entry
  6 472462     pthread_mach_thread_np:entry
  6 472454       pthread_mutex_unlock:entry
  6 472453         pthread_mutex_lock:entry
  6 472454       pthread_mutex_unlock:entry
  6 472453         pthread_mutex_lock:entry
  6 472454       pthread_mutex_unlock:entry
  6 472641                     stat:entry
  6 472645                   stat64:entry
  6 472645                   stat64:entry
  6 472441                   malloc:entry
  6 472443                     free:entry
  6 472441                   malloc:entry
  6 472443                     free:entry
  6 472453         pthread_mutex_lock:entry
  6 472454       pthread_mutex_unlock:entry
  6 472458      pthread_mutex_destroy:entry
  6 472443                     free:entry
  6 472443                     free:entry
  6 472443                     free:entry
```

```
  5 472455        pthread_getspecific:entry
  5 472453         pthread_mutex_lock:entry
  5 472454       pthread_mutex_unlock:entry
  5 472453         pthread_mutex_lock:entry
  5 472454       pthread_mutex_unlock:entry
  5 472453         pthread_mutex_lock:entry
  5 472454       pthread_mutex_unlock:entry
  5 472710                    fwrite:entry
  5 472453         pthread_mutex_lock:entry
  5 472454       pthread_mutex_unlock:entry
  5 472710                    fwrite:entry
  5 472453         pthread_mutex_lock:entry
  5 472454       pthread_mutex_unlock:entry
  5 472710                    fwrite:entry
  5 472453         pthread_mutex_lock:entry
  5 472454       pthread_mutex_unlock:entry
  5 472710                    fwrite:entry
  5 472453         pthread_mutex_lock:entry
  5 472454       pthread_mutex_unlock:entry
  5 472453         pthread_mutex_lock:entry
  5 472454       pthread_mutex_unlock:entry
  5 472453         pthread_mutex_lock:entry
  5 472454       pthread_mutex_unlock:entry
```

```
void run_rt_thread()
{
    std::mutex m:


    get_file_size();
    do_malloc_free();
    do_vector_reserve();
    do_mutex_lock_unlock (m);


}
```

```
const auto policy = SCHED_FIFO;
pthread_t thread = pthread_self();

// Set the real-time scheduling policy and priority
struct sched_param param;
param.sched_priority = 47;

pthread_setschedparam(thread, policy, &param);
```

```
sudo dtrace -c $BUILD_DIR/example -s rt_check.d
```

rt_check.d

```
pid$target::syscall:entry,

pid$target::malloc:entry,
pid$target::free:entry,

pid$target::pthread_*:entry,

pid$target::stat*:entry,
pid$target::stat64*:entry,
pid$target::fstat*:entry,
pid$target::fstat64*:entry,
pid$target::*write*:entry,
pid$target::read*:entry

/curthread->sched_pri>=47/
{
}
```

```
CPU     ID                    FUNCTION:NAME
  5 475325          pthread_mutex_lock:entry
  5 475333                pthread_self:entry
  5 475334      pthread_mach_thread_np:entry
  5 475326        pthread_mutex_unlock:entry
  5 475325          pthread_mutex_lock:entry
  5 475326        pthread_mutex_unlock:entry
  5 475325          pthread_mutex_lock:entry
  5 475326        pthread_mutex_unlock:entry
  5 475513                       stat:entry
  5 475517                     stat64:entry
  5 475517                     stat64:entry
  5 475313                     malloc:entry
  5 475315                       free:entry
  5 475313                     malloc:entry
  5 475315                       free:entry
  5 475325          pthread_mutex_lock:entry
  5 475326        pthread_mutex_unlock:entry
  5 475330       pthread_mutex_destroy:entry
  5 475315                       free:entry
  5 475315                       free:entry
  5 475315                       free:entry
```

```
sudo dtrace -c $BUILD_DIR/example -s rt_check.d
```

rt_check.d

```
pid$target::syscall:entry,

pid$target::malloc:entry,
pid$target::free:entry,

pid$target::pthread_*:entry,

pid$target::stat*:entry,
pid$target::stat64*:entry,
pid$target::fstat*:entry,
pid$target::fstat64*:entry,
pid$target::*write*:entry,
pid$target::read*:entry

/curthread->sched_pri>=47/
{
    ustack();
}
```

```
7 474480      pthread_mutex_destroy:entry
libsystem_pthread.dylib`pthread_mutex_destroy
libc++.1.dylib`std::__1::mutex::~mutex()+0x18
example_external`main::$_0::operator()() const+0x5c
example_external`decltype(std::declval<main::$_0>()()) std::__1::__invoke[abi:ue170006]<main::$_0>(main::$_0&&)+0x18
example_external`void std::__1::__thread_execute[abi:ue170006]<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_delete
example_external`void* std::__1::__thread_proxy[abi:ue170006]<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1:
libsystem_pthread.dylib`_pthread_start+0x88
libsystem_pthread.dylib`thread_start+0x8

7 474465                    free:entry
libsystem_malloc.dylib`free
example_external`std::__1::default_delete<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_delete<std:
example_external`std::__1::unique_ptr<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_delete<std::__1
example_external`std::__1::unique_ptr<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_delete<std::__1
example_external`std::__1::unique_ptr<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_delete<std::__1
example_external`void* std::__1::__thread_proxy[abi:ue170006]<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__1:
libsystem_pthread.dylib`_pthread_start+0x88
libsystem_pthread.dylib`thread_start+0x8

7 474465                    free:entry
libsystem_malloc.dylib`free
libc++.1.dylib`std::__1::__thread_struct::~__thread_struct()+0x24
libc++.1.dylib`std::__1::__thread_specific_ptr<std::__1::__thread_struct>::__at_thread_exit(void*)+0x14
libsystem_pthread.dylib`_pthread_tsd_cleanup+0x1e8
libsystem_pthread.dylib`_pthread_exit+0x54
libsystem_pthread.dylib`_pthread_start+0x94
libsystem_pthread.dylib`thread_start+0x8

7 474465                    free:entry
libsystem_malloc.dylib`free
libsystem_pthread.dylib`_pthread_tsd_cleanup+0x1e8
libsystem_pthread.dylib`_pthread_exit+0x54
libsystem_pthread.dylib`_pthread_start+0x94
libsystem_pthread.dylib`thread_start+0x8
```

```
void run_rt_thread()
{
    std::mutex m;
    set_realtime_max_priority();

    get_file_size();
    do_malloc_free();
    do_vector_reserve();
    do_mutex_lock_unlock (m);


}
```

```
int main()
{
    std::thread t1 ([&]
                    {
                        run_rt_thread();

                        // Reduce priority to avoid thread
                        // clean-up in dtrace output
                        set_realtime_min_priority();
                    });
    t1.join();
```

```
4 472970              pthread_self:entry
          libsystem_pthread.dylib`pthread_self
          example_external`set_realtime_priority(int)+0x2c
          example_external`set_realtime_min_priority()+0x14
          example_external`main::$_0::operator()() const+0x58
          example_external`decltype(std::declval<main::$_0>()()) std::__1::__invoke[abi:ue170006]<main::$_0>(main::$_0&&)+0x18
          example_external`void std::__1::__thread_execute[abi:ue170006]<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_dele
          example_external`void* std::__1::__thread_proxy[abi:ue170006]<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__
          libsystem_pthread.dylib`_pthread_start+0x88
          libsystem_pthread.dylib`thread_start+0x8

4 473006      pthread_setschedparam:entry
          libsystem_pthread.dylib`pthread_setschedparam
          example_external`set_realtime_priority(int)+0x48
          example_external`set_realtime_min_priority()+0x14
          example_external`main::$_0::operator()() const+0x58
          example_external`decltype(std::declval<main::$_0>()()) std::__1::__invoke[abi:ue170006]<main::$_0>(main::$_0&&)+0x18
          example_external`void std::__1::__thread_execute[abi:ue170006]<std::__1::unique_ptr<std::__1::__thread_struct, std::__1::default_dele
          example_external`void* std::__1::__thread_proxy[abi:ue170006]<std::__1::tuple<std::__1::unique_ptr<std::__1::__thread_struct, std::__
          libsystem_pthread.dylib`_pthread_start+0x88
          libsystem_pthread.dylib`thread_start+0x8
```

CLI Tools: dtrace

✓ Easy to run on CI

✓ Can filter by thread priority

✓ Good level of detail

X Difficult to learn

X Explicit list (may not be exhaustive)

X Inlining can remove functions making them untraceable (false negatives*)

X May require code behaviour changes to suit tool

X Requires disabling SIP

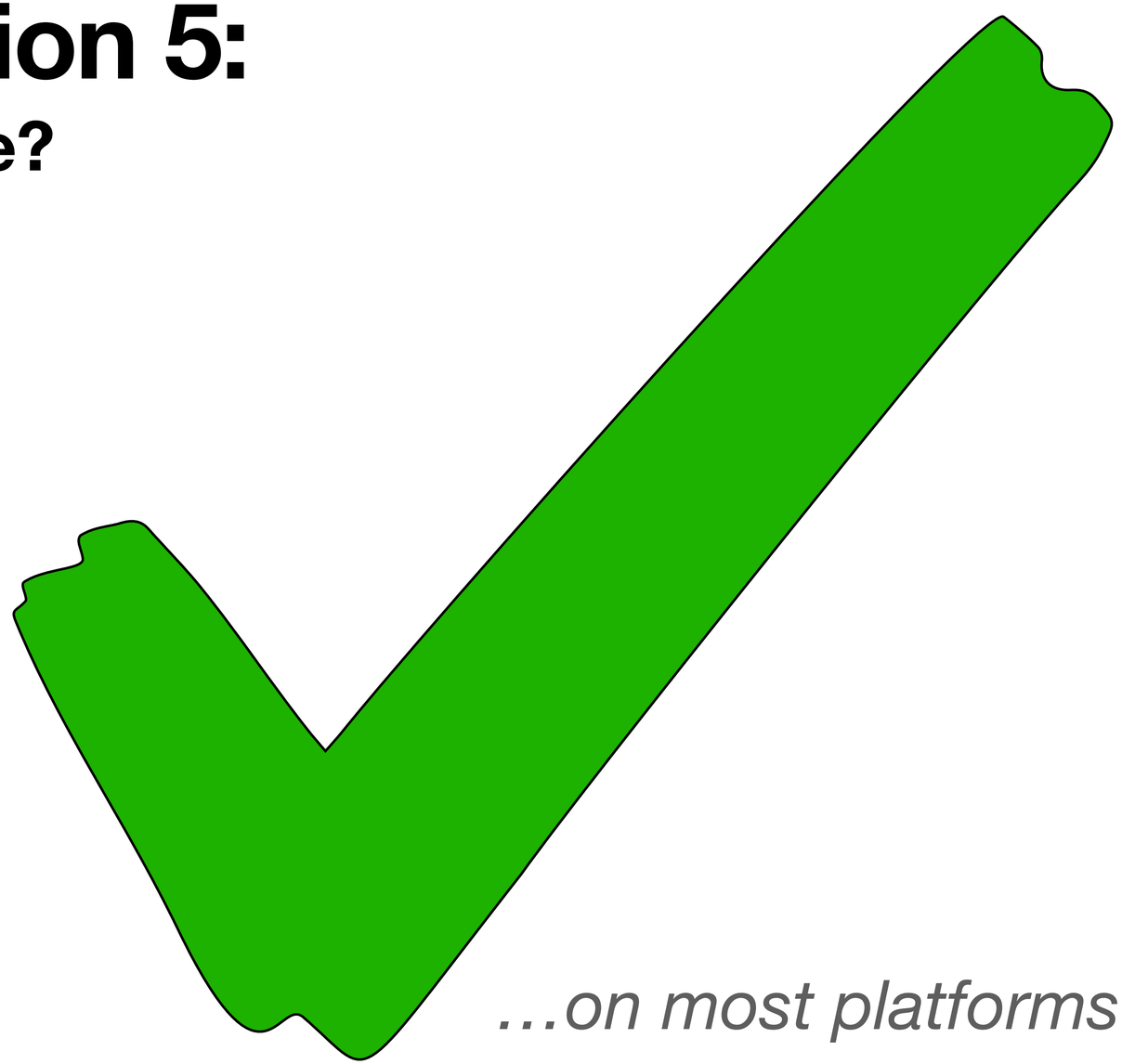
X Can lock-up machine

	GUI (System Trace)	cli (dtrace)			
Easy to use?	✅/⚠️	⚠️			
Clear?	❌	✅			
🧵Filterable?	⚠️	⚠️			
CI?	❌	⚠️			
Portable?	❌	❌			
System calls?	✅	✅			
Malloc/free?	⚠️	✅			
Lock/unlock?	✅	✅			
3rd party code?	✅	✅			
Notes	Different tools for different tasks	Requires disabling SIP			

Real-time Quiz Question 5:

Is this assignment real-time safe?

```
std::array<double, 2> d;
std::function<void()> fn = [d] { };
```

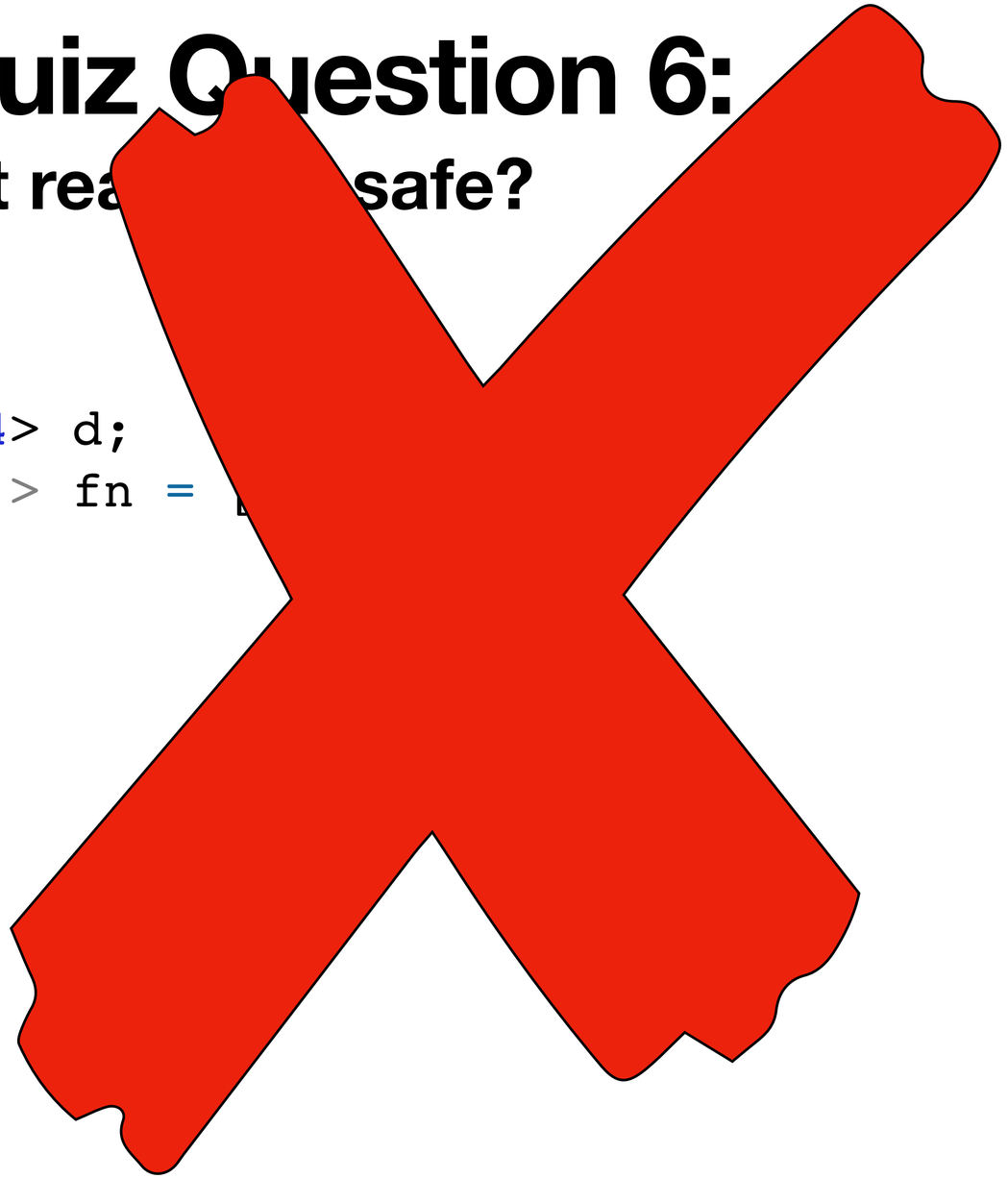


…on most platforms

Real-time Quiz Question 6:

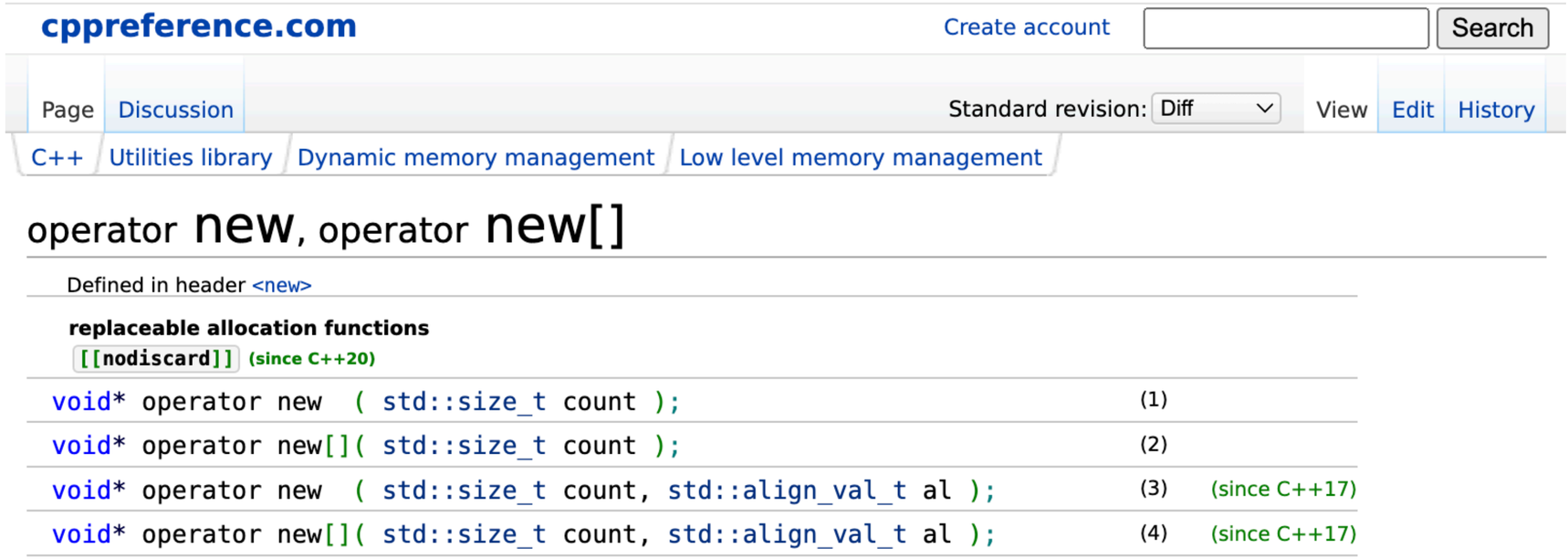
Is this assignment rea safe?

```
std::array<double, 4> d;
std::function<void()> fn =
```



Code

operator new/delete



"The versions (1-4) are implicitly declared in each translation unit even if the <new> header is not included. ***Versions (1-8) are replaceable: a user-provided non-member function with the same signature defined anywhere in the program, in any source file, replaces the default version****. Its declaration does not need to be visible."*

```
void* operator new (std::size_t sz)
{
   std::cout << “new called\n”;
   return std::malloc (sz);
}

void* operator new[] (std::size_t sz)
{
    std::cout << “new[] called\n”;
    return std::malloc (sz);
}

void operator delete (void* ptr) noexcept
{
    std::cout << “delete called\n”;
    std::free (ptr);
}

void operator delete[] (void* ptr) noexcept
{
    std::cout << “delete[] called\n”;
    std::free (ptr);
}
```

```
$./run

new called
new called
delete called
new called
delete called
delete called
```

```
void* operator new (std::size_t sz)
{
    if (is_real_time_context())
    {
        // Log violation
        ...
    }

    return std::malloc (sz);
}
```

```
struct realtime_context_state
{
    realtime_context_state() = default;

    void realtime_enter()               { realtime_flag.store (true); }

    void realtime_exit()                { realtime_flag.store (false); }

    /// Returns true if this is in a real-time state
    bool is_realtime_context() const    { return realtime_flag.load(); }

private:
    std::atomic<bool> realtime_flag { false };
};


inline realtime_context_state& get_realtime_context_state()
{
    thread_local realtime_context_state rcs;
    return rcs;
}
```

```
inline bool is_real_time_context()
{
    return get_realtime_context_state().is_realtime_context();
}
```

```
struct realtime_context
{
    realtime_context()
    {
        get_realtime_context_state().realtime_enter();
    }

    ~realtime_context()
    {
        get_realtime_context_state().realtime_exit();
    }
};
```

```
void run_rt_thread()
{
    std::mutex m;

    realtime_context rc;

    get_file_size();
    do_malloc_free();
    do_vector_reserve();
    do_mutex_lock_unlock (m);
}


void* operator new (std::size_t sz)
{
    if (is_real_time_context())
    {

        std::cerr << "!!! WARNING: Illegal allocation of " << sz << " bytes\n";
        std::cerr << get_stacktrace();
    }

    return std::malloc (sz);
}
```

```
!!! WARNING: Illegal allocation of 200 bytes
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaad6f8a1d8]
 ./example_external(_Znwm+0x78) [0xaaaad6f89ae4]
  /lib/aarch64-linux-gnu/libstdc++.so.6(+0x17a44c) [0xffff919da44c]
   /lib/aarch64-linux-gnu/libstdc++.so.6(_ZNSt10filesystem7__cxx114path14_M_split_cmptsEv+0x3a8) [0xffff919da1f8]
    ./example_external(_ZNSt10filesystem7__cxx114pathC1IA13_cS1_EERKT_NS1_6formatE+0x84) [0xaaaad6f880dc]
     ./example_external(_Z13get_file_sizev+0x38) [0xaaaad6f872f4]
      ...

!!! WARNING: Illegal deletion
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaad6f8a1d8]
 ./example_external(_ZdlPvm+0x5c) [0xaaaad6f89e28]
  ./example_external(_ZNSt10unique_ptrINSt10filesystem7__cxx114path5_List5_ImplENS3_13_Impl_deleterEED2Ev+0x50) [0xaaaad6f87f84]
   ./example_external(_ZNSt10filesystem7__cxx114path5_ListD2Ev+0x14) [0xaaaad6f87bc8]
    ./example_external(_ZNSt10filesystem7__cxx114pathD1Ev+0x18) [0xaaaad6f87bec]
     ./example_external(_Z13get_file_sizev+0x50) [0xaaaad6f8730c]
      ...

!!! WARNING: Illegal allocation of 168 bytes
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaad6f8a1d8]
 ./example_external(_Znwm+0x78) [0xaaaad6f89ae4]
  ./example_external(_ZN9__gnu_cxx13new_allocatorIiE8allocateEmPKv+0x6c) [0xaaaad6f898c8]
   ./example_external(_ZNSt16allocator_traitsISaIiEE8allocateERS0_m+0x50) [0xaaaad6f89338]
    ./example_external(_ZNSt12_Vector_baseIiSaIiEE11_M_allocateEm+0x28) [0xaaaad6f88bfc]
     ./example_external(_ZNSt6vectorIiSaIiEE7reserveEm+0x80) [0xaaaad6f882a4]
      ./example_external(_Z17do_vector_reservev+0x34) [0xaaaad6f873b4]
       ...

!!! WARNING: Illegal deletion
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaad6f8a1d8]
 ./example_external(_ZdlPvm+0x5c) [0xaaaad6f89e28]
  ./example_external(_ZN9__gnu_cxx13new_allocatorIiE10deallocateEPim+0x28) [0xaaaad6f89820]
   ./example_external(_ZNSt16allocator_traitsISaIiEE10deallocateERS0_Pim+0x58) [0xaaaad6f891f8]
    ./example_external(_ZNSt12_Vector_baseIiSaIiEE13_M_deallocateEPim+0x30) [0xaaaad6f88b18]
     ./example_external(_ZNSt12_Vector_baseIiSaIiEED1Ev+0x3c) [0xaaaad6f881c0]
      ./example_external(_ZNSt6vectorIiSaIiEED2Ev+0x40) [0xaaaad6f88214]
       ./example_external(_Z17do_vector_reservev+0x3c) [0xaaaad6f873bc]
        ...
```

```
!!! WARNING: Illegal allocation of 200 bytes
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaad6f8a1d8]
 ./example_external(_Znwm+0x78) [0xaaaad6f89ae4]
  /lib/aarch64-linux-gnu/libstdc++.so.6(+0x17a44c) [0xffff919da44c]
   /lib/aarch64-linux-gnu/libstdc++.so.6(_ZNSt10filesystem7__cxx114path14_M_split_cmptsEv+0x3a8) [0xffff919da1f8]
    ./example_external(_ZNSt10filesystem7__cxx114pathC1IA13_cS1_EERKT_NS1_6formatE+0x84) [0xaaaad6f880dc]
     ./example_external(_Z13get_file_sizev+0x38) [0xaaaad6f872f4]
      ...

!!! WARNING: Illegal deletion
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaad6f8a1d8]
 ./example_external(_ZdlPvm+0x5c) [0xaaaad6f89e28]
  ./example_external(_ZNSt10unique_ptrINSt10filesystem7__cxx114path5_List5_ImplENS3_13_Impl_deleterEED2Ev+0x50) [0xaaaad6f87f84]
   ./example_external(_ZNSt10filesystem7__cxx114path5_ListD2Ev+0x14) [0xaaaad6f87bc8]
    ./example_external(_ZNSt10filesystem7__cxx114pathD1Ev+0x18) [0xaaaad6f87bec]
     ./example_external(_Z13get_file_sizev+0x50) [0xaaaad6f8730c]
      ...
```

```
!!! WARNING: Illegal allocation of 168 bytes
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaad6f8a1d8]
 ./example_external(_Znwm+0x78) [0xaaaad6f89ae4]
  ./example_external(_ZN9__gnu_cxx13new_allocatorIiE8allocateEmPKv+0x6c) [0xaaaad6f898c8]
   ./example_external(_ZNSt16allocator_traitsISaIiEE8allocateERS0_m+0x50) [0xaaaad6f89338]
    ./example_external(_ZNSt12_Vector_baseIiSaIiEE11_M_allocateEm+0x28) [0xaaaad6f88bfc]
     ./example_external(_ZNSt6vectorIiSaIiEE7reserveEm+0x80) [0xaaaad6f882a4]
      ./example_external(_Z17do_vector_reservev+0x34) [0xaaaad6f873b4]
       ...

!!! WARNING: Illegal deletion
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaad6f8a1d8]
 ./example_external(_ZdlPvm+0x5c) [0xaaaad6f89e28]
  ./example_external(_ZN9__gnu_cxx13new_allocatorIiE10deallocateEPim+0x28) [0xaaaad6f89820]
   ./example_external(_ZNSt16allocator_traitsISaIiEE10deallocateERS0_Pim+0x58) [0xaaaad6f891f8]
    ./example_external(_ZNSt12_Vector_baseIiSaIiEE13_M_deallocateEPim+0x30) [0xaaaad6f88b18]
     ./example_external(_ZNSt12_Vector_baseIiSaIiEED1Ev+0x3c) [0xaaaad6f881c0]
      ./example_external(_ZNSt6vectorIiSaIiEED2Ev+0x40) [0xaaaad6f88214]
       ./example_external(_Z17do_vector_reservev+0x3c) [0xaaaad6f873bc]
        ...
```

```
{
    realtime_context rc;

    get_file_size();
    do_malloc_free();
    do_vector_reserve();
    do_mutex_lock_unlock (m);
});

void do_malloc_free()
{
    auto m = malloc (1024);
    free (m);
}
```

```
template<typename mutex_type>
struct checked_lock
{
    checked_lock (mutex_type& m)
        : lock (m)
    {
        if (is_real_time_context())
        {
            non_realtime_context nrtc;
            std::cerr << "!!! WARNING: Locking in real-time context\n";
            std::cerr << get_stacktrace();
        }
    }

    ~checked_lock()
    {
        if (is_real_time_context())
        {
            non_realtime_context nrtc;
            std::cerr << "!!! WARNING: Unlocking in real-time context\n";
            std::cerr << get_stacktrace();
        }
    }

private:
    std::unique_lock<mutex_type> lock;
};
```

```
void do_mutex_lock_unlock (std::mutex& m)
{
    checked_lock l (m);
}
```

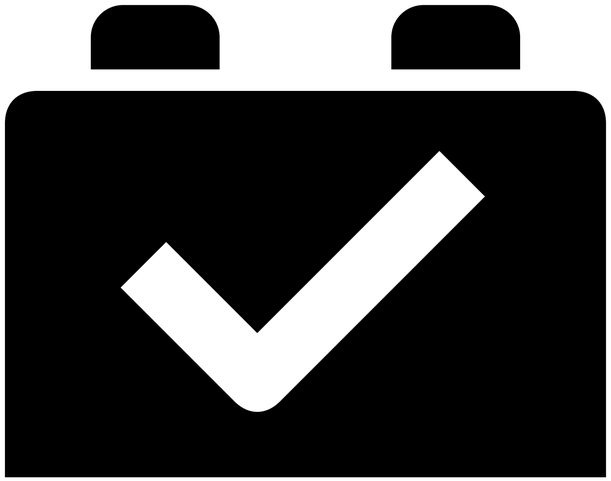
```
!!! WARNING: Locking in real-time context
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabdfc80c4
 ./example_external(_ZN12checked_lockISt5mutexEC2ERS0_+0x6c) [0xaaa
  ./example_external(_Z20do_mutex_lock_unlockRSt5mutex+0x2c) [0
   ...

!!! WARNING: Unlocking in real-time context
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabdfc80c4
 ./example_external(_ZN12checked_lockISt5mutexED1Ev+0x58) [0xaaaabd
  ./example_external(_Z20do_mutex_lock_unlockRSt5mutex+0x34) [0
   ...
```

```cpp
void do_read_file()
{
    namespace fs = std::filesystem;

    if (fs::path file_path ("/usr/bin/awk");
        fs::exists (file_path))
    {
        // Open the file in binary mode
        const auto file_size = fs::file_size (file_path);
        std::vector<char> file_data (file_size);

        // Read the file into the vector
        std::ifstream file (file_path, std::ios::binary);
        file.read (file_data.data(), file_size);

        // Check if the file was read successfully
        if (file)
            std::cout << "File read successfully." << std::endl;
        else
            std::cout << "Error reading the file." << std::endl;

        file.close();
    }
    else
    {
        std::cout << "File does not exist." << std::endl;
    }
}
```

Code

- ✓ Easy to use
- ✓ Explicit real-time scopes
- ✓ Easy to run on CI
- ✓ Good level of detail
- ✓ Can be easily disabled
- ✓ Portable

- x Not exhaustive
  - ✓ new/delete
  - ⚠ locks (limited)
  - x No malloc/free
  - x No system calls
- x Only owned code (no 3rd party libs)



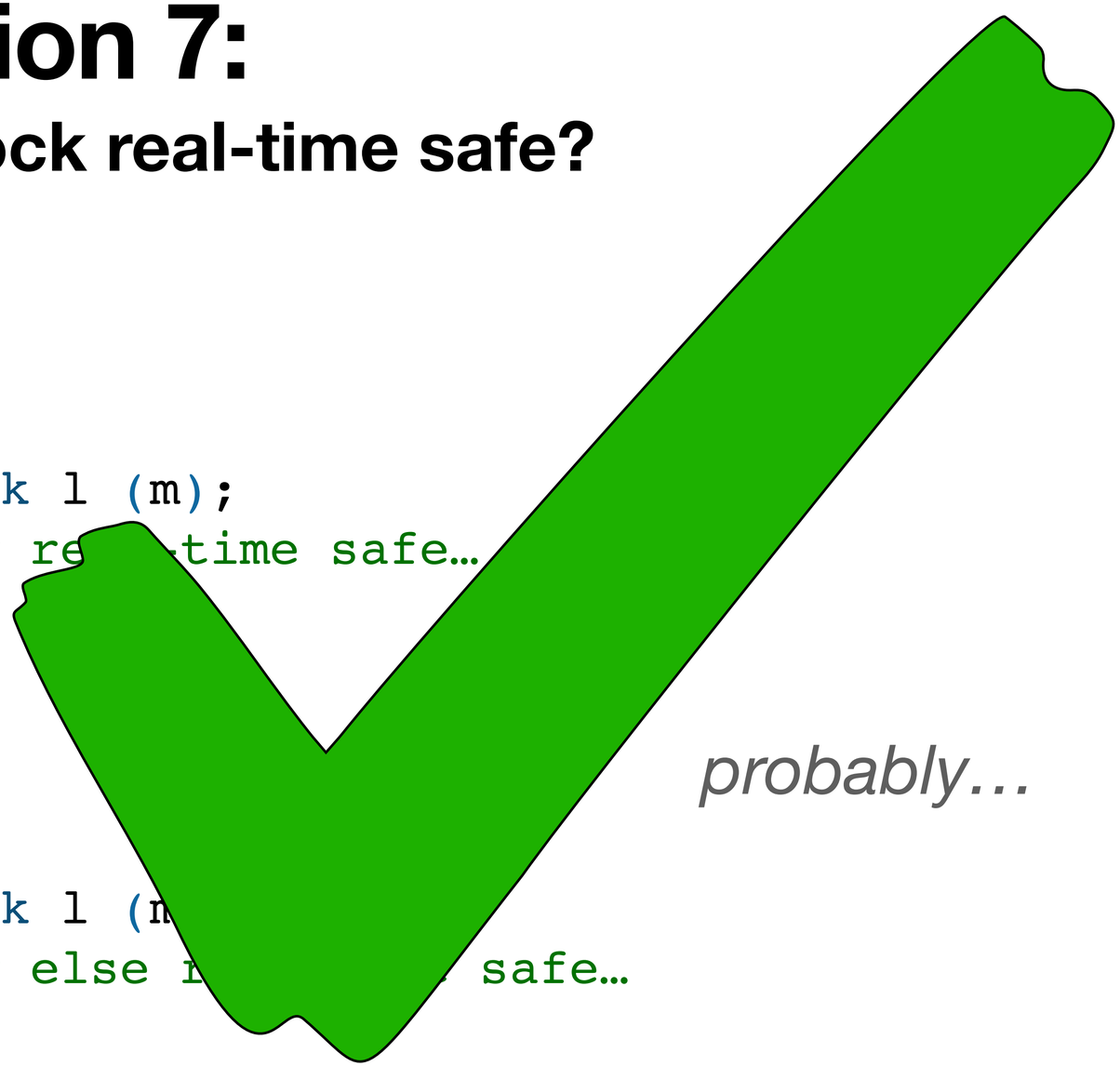
github.com/Tracktion/pluginval

	GUI (System Trace)	cli (dtrace)	code		
Easy to use?	✅/⚠️	⚠️	✅		
Clear?	❌	✅	✅		
🧵Filterable?	⚠️	⚠️	✅		
CI?	❌	⚠️	✅		
Portable?	❌	❌	✅		
System calls?	✅	✅	❌		
Malloc/free?	⚠️	✅	❌		
Lock/unlock?	✅	✅	⚠️		
3rd party code?	✅	✅	❌		
Notes	Different tools for different tasks	Requires disabling SIP			

Real-time Quiz Question 7:

Is this un-contended mutex unlock real-time safe?

```
std::mutex m;

std::thread t1 ([&]
                {
                    std::unique_lock l (m);
                    // Do something re   -time safe…
                });
t1.join();

std::thread t2 ([&]
                {
                    std::unique_lock l (m
                    // Do something else r      safe…
                });
t2.join();
```

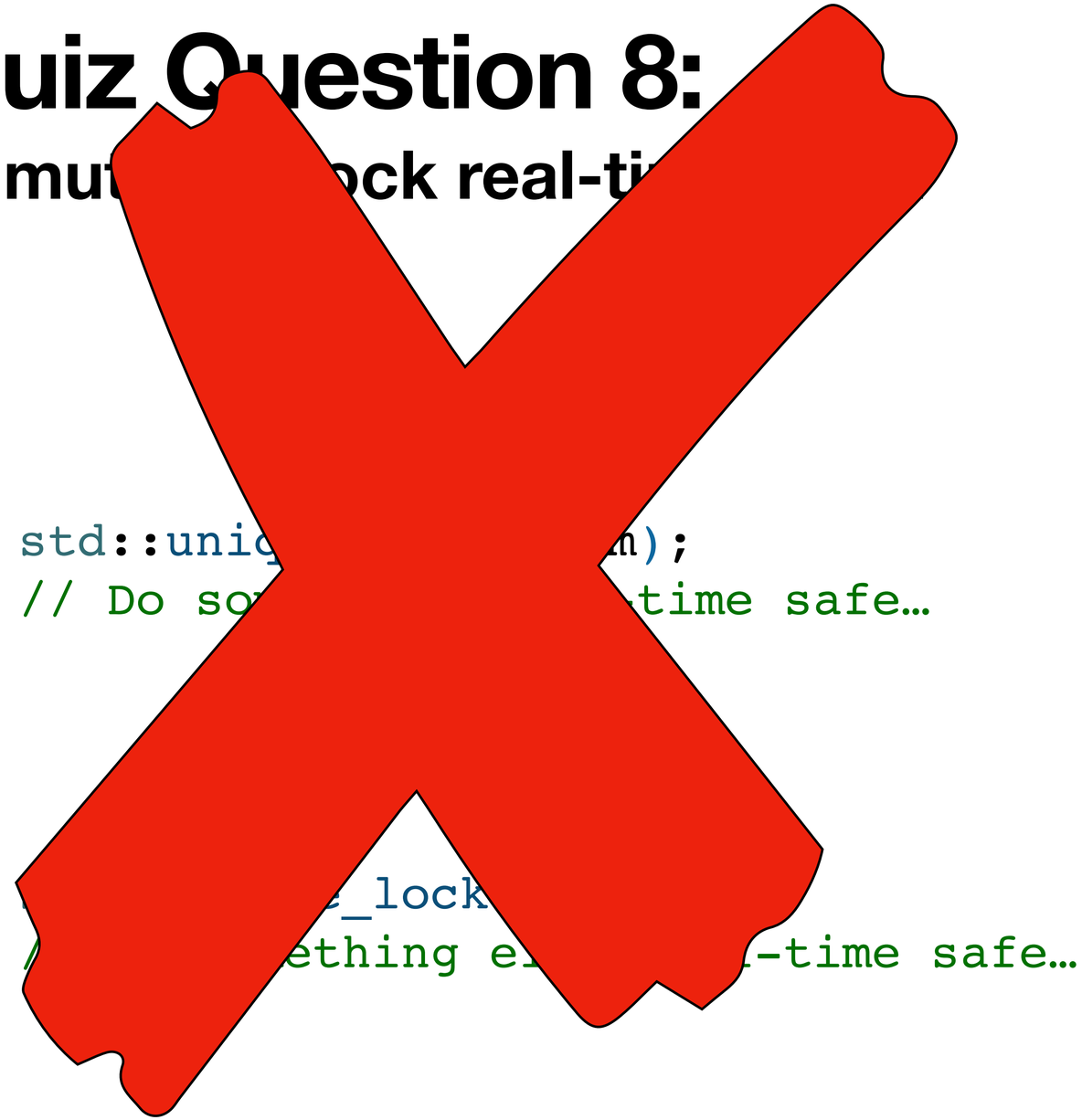


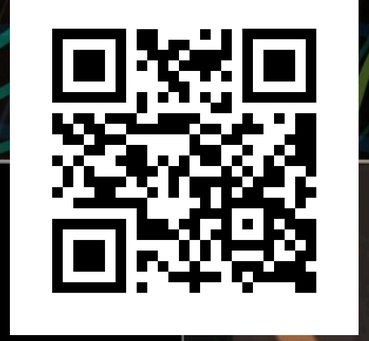
probably…

Real-time Quiz Question 8:

Is this contended mut ock real-ti

```
std::mutex m;

std::thread t1 ([&]
                {
                    std::unig m);
                    // Do so -time safe…
                });

std::thread t2 ([&]
                {
                    e_lock
                    ething e -time safe…
                });
t1.join();
t2.join();
```





Sin #4a

You may never take a lock in real-time code

Normal lock: `std::mutex::lock()`

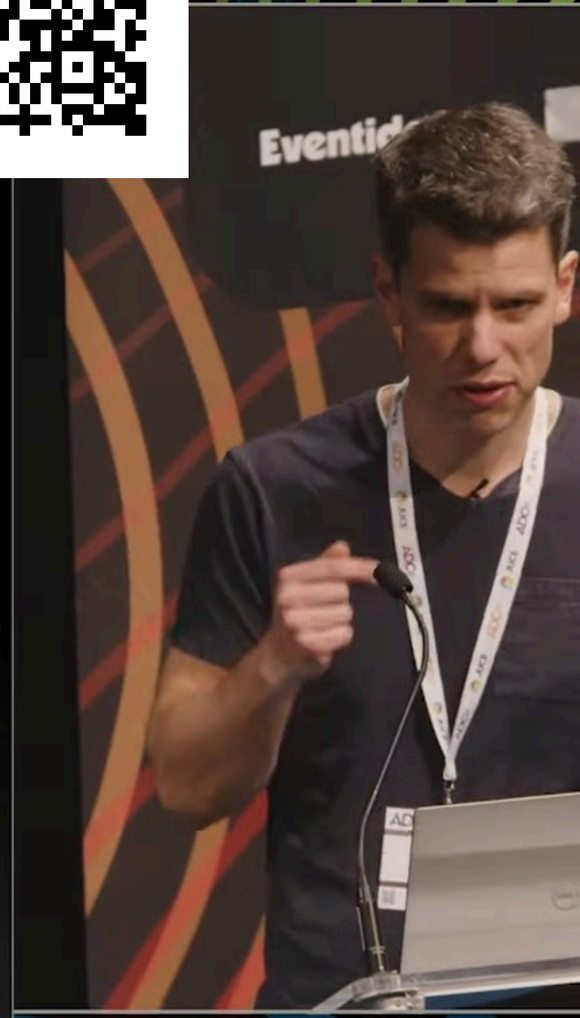
std::mutex::lock is wrapper around `pthread_mutex_lock` (linux source code) - edited for brevity

```
while (1) {
   /* Try to acquire the lock through a CAS from 0 (not acquired) to our TID  */
   oldval = atomic_compare_and_exchange_val_acq (&mutex->__data.__lock,
                                                 tid, 0);
   if (__glibc_likely (oldval == 0))
       break;

    ...
    /* Block using the futex and reload current lock value.  */
     futex_wait ((unsigned int *) &mutex->__data.__lock, oldval,
                 PTHREAD_ROBUST_MUTEX_PSHARED (mutex));
     oldval = mutex->__data.__lock;
}
return;
```

Real-time safe

OS call which can block thread

➔ Lock is real-time safe as long as it's never contended



Sin #4a

You may never take a lock in real-time code

Normal lock: `std::mutex::lock()`

std::mutex::lock is wrapper around `pthread_mutex_lock` (linux source code) - edited for brevity

```
while (1) {
   /* Try to acquire the lock through a CAS from 0 (not acquired) to our TID  */
   oldval = atomic_compare_and_exchange_val_acq (&mutex->__data.__lock,
                                                 tid, 0);
   if (__glibc_likely (oldval == 0))
      break;

   ...
   /* Block using the futex and reload current lock value.  */
    futex_wait ((unsigned int *) &mutex->__data.__lock, oldval,
                PTHREAD_ROBUST_MUTEX_PSHARED (mutex));
    oldval = mutex->__data.__lock;
}
return;
```

Real-time safe

OS call which can block thread

→ Lock is real-time safe as long as it's never contended

Interposing

Interposing

- Define our own versions of functions
- Ensure these get called

CMakeLists.txt:

```
add_library(rtcheck SHARED
    rtcheck/rtcheck.cpp
)

target_link_libraries(example_app
    rtcheck
)
```

rtcheck.cpp:

```
extern "C" void* malloc(size_t size)
{
    log_function_if_realtime_context(__func__);

    static auto real_malloc = (void* (*)(size_t))dlsym(RTLD_NEXT, "malloc");
    return real_malloc(size);
}
```

```
static auto real_malloc
    = (void* (*)(size_t))dlsym(RTLD_NEXT, "malloc");
```

CMakeLists.txt:

```
add_library(rtcheck SHARED
    rtcheck/rtcheck.cpp
)

target_link_libraries(example
    rtcheck
)
```

```
static void* (*real_malloc)(size_t);

__attribute__((constructor))
void init()
{
    real_malloc = (void* (*)(size_t))
                        dlsym(RTLD_NEXT, "malloc");
}
```

rtcheck.cpp:

```
extern "C" void* malloc(size_t size)
{
    log_function_if_realtime_context(__func__);


    return real_malloc(size);
}
```

```
Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(_Z32log_function_if_realtime_contextPKc+0x58)
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(malloc+0x18) [0xffff976e3d9c]
   /lib/aarch64-linux-gnu/libstdc++.so.6(_Znwm+0x1c) [0xffff975443ac]
    /lib/aarch64-linux-gnu/libstdc++.so.6(+0x17a44c) [0xffff9761a44c]
     /lib/aarch64-linux-gnu/libstdc++.so.6(_ZNSt10filesystem7__cxx114path14_M_split_cmptsEv+0x3a8) [0xffff9761a1f8]
      ./example_external(_ZNSt10filesystem7__cxx114pathC2IA13_cS1_EERKT_NS1_6formatE+0x84) [0xaaaabc289c04]
       ./example_external(_Z13get_file_sizev+0x38) [0xaaaabc2885a8]

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(_Z32log_function_if_realtime_contextPKc+0x58)
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(free+0x18) [0xffff976e3e40]
   ./example_external(_ZNSt10unique_ptrINSt10filesystem7__cxx114path5_List5_ImplENS3_13_Impl_deleterEED1Ev+0x50) [0xaaaabc2899b8]
    ./example_external(_ZNSt10filesystem7__cxx114path5_ListD2Ev+0x14) [0xaaaabc2891d0]
     ./example_external(_ZNSt10filesystem7__cxx114pathD1Ev+0x18) [0xaaaabc2891f4]
      ./example_external(_Z13get_file_sizev+0x50) [0xaaaabc2885c0]

Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(_Z32log_function_if_realtime_contextPKc+0x58)
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(malloc+0x18) [0xffff976e3d9c]
   ./example_external(_Z14do_malloc_freev+0x10) [0xaaaabc28861c]

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(_Z32log_function_if_realtime_contextPKc+0x58)
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(free+0x18) [0xffff976e3e40]
   ./example_external(_Z14do_malloc_freev+0x1c) [0xaaaabc288628]

Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(_Z32log_function_if_realtime_contextPKc+0x58)
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(malloc+0x18) [0xffff976e3d9c]
   /lib/aarch64-linux-gnu/libstdc++.so.6(_Znwm+0x1c) [0xffff975443ac]
    ./example_external(_ZN9__gnu_cxx13new_allocatorIiE8allocateEmPKv+0x6c) [0xaaaabc28d210]
     ./example_external(_ZNSt16allocator_traitsISaIiEE8allocateERS0_m+0x50) [0xaaaabc28c87c]
      ./example_external(_ZNSt12_Vector_baseIiSaIiEE11_M_allocateEm+0x28) [0xaaaabc28bbf8]
       ./example_external(_ZNSt6vectorIiSaIiEE7reserveEm+0x80) [0xaaaabc289dcc]
        ./example_external(_Z17do_vector_reservev+0x34) [0xaaaabc288668]

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(_Z32log_function_if_realtime_contextPKc+0x58)
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external/build_linux-gnu/librt_checked.so(free+0x18) [0xffff976e3e40]
   ./example_external(_ZN9__gnu_cxx13new_allocatorIiE10deallocateEPim+0x28) [0xaaaabc28d168]
    ./example_external(_ZNSt16allocator_traitsISaIiEE10deallocateERS0_Pim+0x58) [0xaaaabc28c73c]
     ./example_external(_ZNSt12_Vector_baseIiSaIiEE13_M_deallocateEPim+0x30) [0xaaaabc28bb14]
      ./example_external(_ZNSt12_Vector_baseIiSaIiEED1Ev+0x3c) [0xaaaabc289ce8]
       ./example_external(_ZNSt6vectorIiSaIiEED1Ev+0x40) [0xaaaabc289d3c]
        ./example_external(_Z17do_vector_reservev+0x3c) [0xaaaabc288670]
```

```
Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_externa
   /lib/aarch64-linux-gnu/libstdc++.so.6(_Znwm+0x1c) [0xffff975443ac]
    /lib/aarch64-linux-gnu/libstdc++.so.6(+0x17a44c) [0xffff9761a44c]
     /lib/aarch64-linux-gnu/libstdc++.so.6(_ZNSt10filesystem7__cxx114path14_M_split_cmptsEv+0x3a8) [0xffff9761a1f8]
      ./example_external(_ZNSt10filesystem7__cxx114pathC2IA13_cS1_EERKT_NS1_6formatE+0x84) [0xaaaabc289c04]
       ./example_external(_Z13get_file_sizev+0x38) [0xaaaabc2885a8]

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_externa
   ./example_external(_ZNSt10unique_ptrINSt10filesystem7__cxx114path5_List5_ImplENS3_13_Impl_deleterEED1Ev+0x50) [0xaa
    ./example_external(_ZNSt10filesystem7__cxx114path5_ListD2Ev+0x14) [0xaaaabc2891d0]
     ./example_external(_ZNSt10filesystem7__cxx114pathD1Ev+0x18) [0xaaaabc2891f4]
      ./example_external(_Z13get_file_sizev+0x50) [0xaaaabc2885c0]
```

```
Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_externa
   ./example_external(_Z14do_malloc_freev+0x10) [0xaaaabc28861c]

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_externa
   ./example_external(_Z14do_malloc_freev+0x1c) [0xaaaabc288628]
```

```
Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_externa
   /lib/aarch64-linux-gnu/libstdc++.so.6(_Znwm+0x1c) [0xffff975443ac]
    ./example_external(_ZN9__gnu_cxx13new_allocatorIiE8allocateEmPKv+0x6c) [0xaaaabc28d210]
     ./example_external(_ZNSt16allocator_traitsISaIiEE8allocateERS0_m+0x50) [0xaaaabc28c87c]
      ./example_external(_ZNSt12_Vector_baseIiSaIiEE11_M_allocateEm+0x28) [0xaaaabc28bbf8]
       ./example_external(_ZNSt6vectorIiSaIiEE7reserveEm+0x80) [0xaaaabc289dcc]
        ./example_external(_Z17do_vector_reservev+0x34) [0xaaaabc288668]

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaabc289464]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_external
[0xffff976e46e0]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/example_externa
   ./example_external(_ZN9__gnu_cxx13new_allocatorIiE10deallocateEPim+0x28) [0xaaaabc28d168]
    ./example_external(_ZNSt16allocator_traitsISaIiEE10deallocateERS0_Pim+0x58) [0xaaaabc28c73c]
     ./example_external(_ZNSt12_Vector_baseIiSaIiEE13_M_deallocateEPim+0x30) [0xaaaabc28bb14]
      ./example_external(_ZNSt12_Vector_baseIiSaIiEED1Ev+0x3c) [0xaaaabc289ce8]
       ./example_external(_ZNSt6vectorIiSaIiEED1Ev+0x40) [0xaaaabc289d3c]
        ./example_external(_Z17do_vector_reservev+0x3c) [0xaaaabc288670]
```

```
{
    realtime_context rc;

    get_file_size();
    do_malloc_free();
    do_vector_reserve();
});
```

```cpp
extern "C" int pthread_mutex_lock(pthread_mutex_t *mutex)
{
    log_function_if_realtime_context (__func__);

    static auto real_pthread_mutex_lock = (int (*)(pthread_mutex_t *))dlsym(RTLD_NEXT, "pthread_mutex_lock");
    return real_pthread_mutex_lock(mutex);
}

extern "C" int pthread_mutex_unlock(pthread_mutex_t *mutex)
{
    log_function_if_realtime_context (__func__);

    static auto real_pthread_mutex_unlock = (int (*)(pthread_mutex_t *))dlsym(RTLD_NEXT, "pthread_mutex_unlock");
    return real_pthread_mutex_unlock(mutex);
}
```

```
Real-time violation: intercepted call to real-time unsafe function `pthread_mutex_lock` in real-time
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaadc5c986c]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/exampl
librt_checked.so(_Z32log_function_if_realtime_contextPKc+0x58) [0xffff974956dc]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/examp
[0xffff97494838]
   ./example_external(+0x9278) [0xaaaadc5c9278]
    ./example_external(_ZNSt5mutex4lockEv+0x14) [0xaaaadc5c92d8]
     ./example_external(_ZNSt11unique_lockISt5mutexE4lockEv+0x48) [0xaaaadc5cbfbc]
      ./example_external(_ZNSt11unique_lockISt5mutexEC2ERS0_+0x34) [0xaaaadc5cb310]
       ./example_external(_Z20do_mutex_lock_unlockRSt5mutex+0x2c) [0xaaaadc5c89f0]
```

```
Real-time violation: intercepted call to real-time unsafe function `pthread_mutex_unlock` in real-tim
./example_external(_Z14get_stacktraceB5cxx11v+0x3c) [0xaaaadc5c986c]
 /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/exampl
librt_checked.so(_Z32log_function_if_realtime_contextPKc+0x58) [0xffff974956dc]
  /home/parallels/Desktop/Parallels Shared Folders/Home/Documents/Developement/Lectures/CppOnSea 2024/examp
[0xffff974948dc]
   ./example_external(+0x92b4) [0xaaaadc5c92b4]
    ./example_external(_ZNSt5mutex6unlockEv+0x14) [0xaaaadc5c9310]
     ./example_external(_ZNSt11unique_lockISt5mutexE6unlockEv+0x48) [0xaaaadc5cc01c]
      ./example_external(_ZNSt11unique_lockISt5mutexED1Ev+0x24) [0xaaaadc5cb34c]
       ./example_external(_Z20do_mutex_lock_unlockRSt5mutex+0x34) [0xaaaadc5c89f8]
```

```cpp
{
    std::mutex m;

    realtime_context rc;

    get_file_size();
    do_malloc_free();
    do_vector_reserve();
    do_mutex_lock_unlock (m);
});
```

```cpp
void do_mutex_lock_unlock (std::mutex& m)
{
    std::unique_lock l (m);
}
```

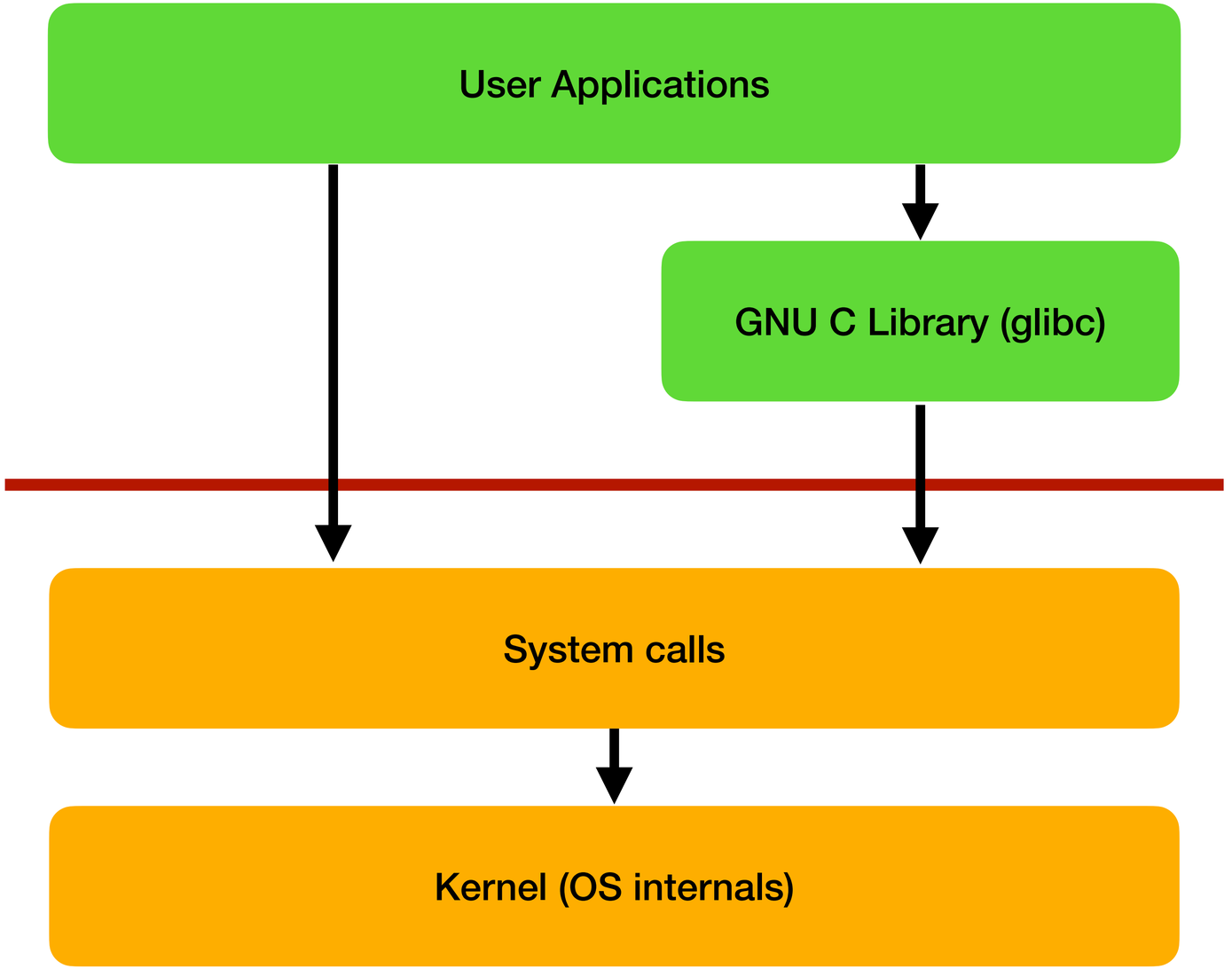
False Positives

Quiz question 8 rev

```
Real-time violation: intercepted call to real-time unsafe function `pthread_mutex_lock`
  in real-time context! Stack trace:
<source>/src/librt_check.so(_Z14get_stacktraceB5cxx11v+0x3c) [ff89d86338]
 <source>/src/librt_check.so(_Z32log_function_if_realtime_contextPKc+0x58) [0xffff89d864f4]
  <source>/src/librt_check.so(pthread_mutex_lock+0x18) [0xffff89d85268]
   <source>/tests/pass_mutex_unique_lock_uncontended(+0x4168) [aac2444168]
    ...
```

```
Real-time violation: intercepted call to real-time unsafe function `pthread_mutex_unlock`
  in real-time context! Stack trace:
<source>/src/librt_check.so(_Z14get_stacktraceB5cxx11v+0x3c) [ff89d86338]
 <source>/src/librt_check.so(_Z32log_function_if_realtime_contextPKc+0x58) [0xffff89d864f4]
  <source>/src/librt_check.so(pthread_mutex_unlock+0x18) [0xffff89d85268]
   <source>/tests/pass_mutex_unique_lock_uncontended(+0x4168) [aac2444168]
    ...
```

```
std::mutex m;

std::thread t1 ([&]
                {
                    std::unique_lock l (m);
                    // Do something real-time safe…
                });
t1.join();

std::thread t2 ([&]
                {
                    std::unique_lock l (m);
                    // Do something else real-time safe…
                });
t2.join();
```



User space
(restricted)

Kernel space
(privileged)

Sin #4a

You may never take a lock in real-time code

Normal lock: `std::mutex::lock()`

std::mutex::lock is wrapper around `pthread_mutex_lock` (linux source code) - edited for brevity

```
while (1) {
   /* Try to acquire the lock through a CAS from 0 (not acquired) to our TID  */
   oldval = atomic_compare_and_exchange_val_acq (&mutex->__data.__lock,
                                                 tid, 0);
   if (__glibc_likely (oldval == 0))
      break;

   ...
   /* Block using the futex and reload current lock value.  */
    futex_wait ((unsigned int *) &mutex->__data.__lock, oldval,
                PTHREAD_ROBUST_MUTEX_PSHARED (mutex));
    oldval = mutex->__data.__lock;
}
return;
```

Real-time safe

OS call which can block thread

→ Lock is real-time safe as long as it's never contended

Intercept futex_wait?

```
#define internal_syscall1(number, arg1)                         \
({                                                              \
    unsigned long int resultvar;                                \
    TYPEFY (arg1, __arg1) = ARGIFY (arg1);                      \
    register TYPEFY (arg1, _a1) asm ("rdi") = __arg1;           \
    asm volatile (                                              \
    "syscall\n\t"                                               \
    : "=a" (resultvar)                                          \
    : "0" (number), "r" (_a1)                                   \
    : "memory", REGISTERS_CLOBBERED_BY_SYSCALL);                \
    (long int) resultvar;                                       \
})
```

perf

```
     ? (         ): example_extern/25315  ... [continued]: execve())                                  = 0
 0.021 ( 0.003 ms): example_extern/25315 brk()                                                        = 0xaaaaeb303000
 0.092 ( 0.006 ms): example_extern/25315 faccessat(dfd: -100, filename: 0x9ffdb660, mode: 4)          = -1 ENOENT (No such file or directory)
 0.171 ( 0.005 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffd9480, flags: RDONLY|CLOEXEC)        = 3
 0.192 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                 = 0
 0.202 ( 0.005 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffeb140, flags: RDONLY|CLOEXEC)        = 3
 0.210 ( 0.004 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59feaa0, count: 832)                 = 832
 0.239 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fd6b000, len: 20480)                     = 0
 0.245 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9ff9a000, len: 43040)                     = 0
 0.251 ( 0.006 ms): example_extern/25315 mprotect(start: 0xffff9ff7a000, len: 61440)                  = 0
 0.279 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                 = 0
 0.288 ( 0.004 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffeb680, flags: RDONLY|CLOEXEC)        = 3
 0.294 ( 0.004 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59fea80, count: 832)                 = 832
 0.320 ( 0.009 ms): example_extern/25315 munmap(addr: 0xffff9fd3b000, len: 20480)                     = 0
 0.332 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fd65000, len: 41672)                     = 0
 0.338 ( 0.009 ms): example_extern/25315 mprotect(start: 0xffff9fd54000, len: 61440)                  = 0
 0.361 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                 = 0
 0.367 ( 0.004 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffebbc0, flags: RDONLY|CLOEXEC)        = 3
 0.374 ( 0.003 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59fea60, count: 832)                 = 832
 0.402 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fb87000, len: 36864)                     = 0
 0.407 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fd39000, len: 28264)                     = 0
 0.413 ( 0.005 ms): example_extern/25315 mprotect(start: 0xffff9fd18000, len: 61440)                  = 0
 0.490 ( 0.006 ms): example_extern/25315 close(fd: 3)                                                 = 0
 0.521 ( 0.009 ms): example_extern/25315 openat(dfd: CWD, filename: 0x9ffec100, flags: RDONLY|CLOEXEC)        = 3
 0.541 ( 0.003 ms): example_extern/25315 read(fd: 3, buf: 0xffffd59fea20, count: 832)                 = 832
 0.576 ( 0.004 ms): example_extern/25315 munmap(addr: 0xffff9fae9000, len: 28672)                     = 0
 0.590 ( 0.003 ms): example_extern/25315 munmap(addr: 0xffff9fb87000, len: 32880)                     = 0
 0.602 ( 0.011 ms): example_extern/25315 mprotect(start: 0xffff9fb76000, len: 61440)                  = 0
 0.666 ( 0.003 ms): example_extern/25315 close(fd: 3)                                                 = 0
 0.697 ( 0.005 ms): example_extern/25315 set_tid_address(tidptr: 0xffff9ffe9af0)                      = 25315 (example_externa)
 0.710 ( 0.003 ms): example_extern/25315 set_robust_list(head: 0xffff9ffe9b00, len: 24)               = 0
 0.727 ( 0.003 ms): example_extern/25315 rseq(rseq: 0xffff9ffea1c0, rseq_len: 32, sig: 3559439360)    = 0
 0.781 ( 0.005 ms): example_extern/25315 mprotect(start: 0xffff9fd27000, len: 16384, prot: READ)      = 0
 0.791 ( 0.004 ms): example_extern/25315 mprotect(start: 0xffff9fb85000, len: 4096, prot: READ)       = 0
 0.799 ( 0.004 ms): example_extern/25315 mprotect(start: 0xffff9fd63000, len: 4096, prot: READ)       = 0
 1.794 ( 0.023 ms): example_extern/25315 mprotect(start: 0xffff9ff89000, len: 45056, prot: READ)      = 0
 1.874 ( 0.012 ms): example_extern/25315 mprotect(start: 0xaaaaae426000, len: 4096, prot: READ)       = 0
 1.897 ( 0.012 ms): example_extern/25315 mprotect(start: 0xffff9fff0000, len: 8192, prot: READ)       = 0
 1.953 ( 0.003 ms): example_extern/25315 prlimit64(resource: STACK, old_rlim: 0xffffd59ff4b8)         = 0
 1.983 ( 0.013 ms): example_extern/25315 munmap(addr: 0xffff9ffa5000, len: 67899)                     = 0
 2.053 ( 0.009 ms): example_extern/25315 getrandom(ubuf: 0xffff9fd31930, len: 8, flags: NONBLOCK)     = 8
 2.071 ( 0.003 ms): example_extern/25315 brk()                                                        = 0xaaaaeb303000
 2.083 ( 0.009 ms): example_extern/25315 brk(brk: 0xaaaaeb324000)                                     = 0xaaaaeb324000
 2.321 ( 0.004 ms): example_extern/25315 futex(uaddr: 0xffff9ff977a4, op: WAKE|PRIVATE_FLAG, val: 2147483647)  = 0
 2.489 ( 0.003 ms): example_extern/25315 rt_sigaction(sig: 0x21, act: 0xffffd59ff478, sigsetsize: 8)        = 0
 2.495 ( 0.003 ms): example_extern/25315 rt_sigprocmask(how: UNBLOCK, nset: 0xffffd59ff6a8, sigsetsize: 8)     = 0
 2.507 ( 0.004 ms): example_extern/25315 mprotect(start: 0xffff9f2f0000, len: 8388608, prot: READ|WRITE)       = 0
 2.525 ( 0.004 ms): example_extern/25315 rt_sigprocmask(how: BLOCK, nset: 0xffff9fcd82c8, oset: 0xffffd59ff6a0, sigsetsize: 8) = 0
 2.544 ( 0.096 ms): example_extern/25315 clone(clone_flags: VM|FS|FILES|SIGHAND|THREAD|SYSVSEM|SETTLS|PARENT_SETTID|CHILD_CLEARTID, newsp: 0xffff9faee940, parent_tidptr: 0xffff9faef1d0, tls: 0xffff9faef8c0, chil
 2.647 ( 0.003 ms): example_extern/25315 rt_sigprocmask(how: SETMASK, nset: 0xffffd59ff6a0, sigsetsize: 8)     = 0
 2.676 (         ): example_extern/25315 futex(uaddr: 0xffff9faef1d0, op: WAIT_BITSET|CLOCK_REALTIME, val: 25316, val3: MATCH_ANY) ...
 2.689 ( 0.005 ms): example_extern/25316 rseq(rseq: 0xffff9faef8a0, rseq_len: 32, sig: 3559439360)    = 0
 2.702 ( 0.003 ms): example_extern/25316 set_robust_list(head: 0xffff9faef1e0, len: 24)               = 0
 2.713 ( 0.006 ms): example_extern/25316 rt_sigprocmask(how: SETMASK, nset: 0xffff9faef7f0, sigsetsize: 8)     = 0
 2.728 ( 0.006 ms): example_extern/25316 sched_get_priority_max(policy: 1)                            = 99
 2.742 ( 0.008 ms): example_extern/25316 sched_setscheduler(pid: 25316 (example_externa), policy: FIFO, param: 0xffff9faee6c0) = 0
 2.772 ( 0.004 ms): example_extern/25316 munmap(addr: 0xffff972e0000, len: 13762560)                  = 0
 2.779 ( 0.003 ms): example_extern/25316 munmap(addr: 0xffff9c000000, len: 53346304)                  = 0
 2.785 ( 0.003 ms): example_extern/25316 mprotect(start: 0xffff98000000, len: 135168, prot: READ|WRITE)        = 0
 2.808 ( 0.003 ms): example_extern/25316 sched_get_priority_min(policy: 1)                            = 1
 2.814 ( 0.004 ms): example_extern/25316 sched_setscheduler(pid: 25316 (example_externa), policy: FIFO, param: 0xffff9faee6c0) = 0
 2.823 ( 0.003 ms): example_extern/25316 rt_sigprocmask(how: BLOCK, nset: 0xffff9faef7f0, sigsetsize: 8)       = 0
 2.829 ( 0.005 ms): example_extern/25316 madvise(start: 0xffff9f2e0000, len_in: 8314880, behavior: MADV_DONTNEED) = 0
 2.837 (         ): example_extern/25316 exit()                                                       = ?
 2.676 ( 0.177 ms): example_extern/25315  ... [continued]: futex())                                   = 0
 2.937 (         ): example_extern/25315 exit_group()                                                 = ?
```

perf

```
2.071 ( 0.003 ms): example_extern/25315 brk()                                                                = 0xaaaaeb303000
 2.083 ( 0.009 ms): example_extern/25315 brk(brk: 0xaaaaeb324000)                                             = 0xaaaaeb324000

  2.321 ( 0.004 ms): example_extern/25315 futex(uaddr: 0xffff9ff977a4, op: WAKE|PRIVATE_FLAG, val: 2147483647)  = 0
 2.489 ( 0.003 ms): example_extern/25315 rt_sigaction(sig: 0x21, act: 0xffffd59ff478, sigsetsize: 8)           = 0
 2.495 ( 0.003 ms): example_extern/25315 rt_sigprocmask(how: UNBLOCK, nset: 0xffffd59ff6a8, sigsetsize: 8)     = 0
 2.507 ( 0.004 ms): example_extern/25315 mprotect(start: 0xffff9f2f0000, len: 8388608, prot: READ|WRITE)       = 0
```

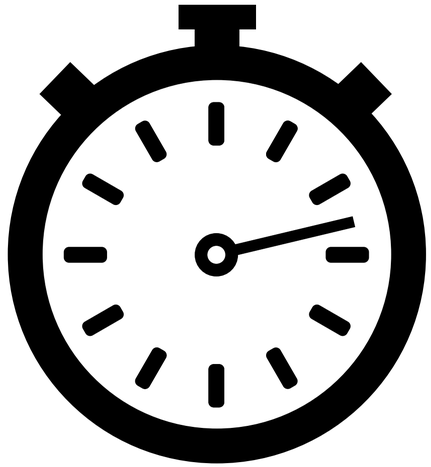
- False positives:
  - libc function calls will be flagged even if they only set an atomic and not result in a system call
- False negatives:
  - If the function call gets inlined so only the asm system call happens, it won’t get flagged

Interposing

- ✓ Relatively easy to use
- ✓ Explicit real-time scopes
- ✓ Easy to run on CI
- ✓ Good level of detail
- ✓ Can be easily disabled
- ✓ Fairly complete
- ✓ Includes 3rd party libs

- x Not portable
  - x No Windows
  - x Different for linux/macOS
- x No direct syscall/kernel interception

	GUI (System Trace)	cli (dtrace)	code	interpose	
Easy to use?	✅/⚠️	⚠️	✅	⚠️	
Clear?	❌	✅	✅	✅	
🧵Filterable?	⚠️	⚠️	✅	✅	
CI?	❌	⚠️	✅	✅	
Portable?	❌	❌	✅	⚠️	
System calls?	✅	✅	❌	⚠️	
Malloc/free?	⚠️	✅	❌	✅	
Lock/unlock?	✅	✅	⚠️	✅	
3rd party code?	✅	✅	❌	✅	
Notes	Different tools for different tasks	Requires disabling SIP		No raw/inline context switches/syscalls	



github.com/Tracktion/rtcheck

Cppcon 2021 The C++ Conference | October 24-29

Timur Doumler

Real-Time programming with the C++ standard library

```
struct random_sample_gen
{
    // returns a random float in the interval [0, 1)
    float operator()()
    {
        auto x = float (rng() - rng.min()) / float (rng.max() + 1);
        if (x == 1.0f) x -= std::numeric_limits<float>::epsilon();
        return x;
    }

private:
    xorshift_rand rng { std::random_device{}() };
};

void process(buffer& b)
{
    std::ranges::fill(b, random_sample_gen{});
}
```

```
void run_rt_thread()
{
    realtime_context rc;

    get_file_size();
    do_malloc_free();
    do_vector_reserve();
    do_mutex_lock_unlock (m);
    do_read_file();
};
```

[[realtime_safe]]

```
void run_rt_thread()
{
    realtime_context rc;

    get_file_size();
    do_malloc_free();
    do_vector_reserve();
    do_mutex_lock_unlock (m);
    do_read_file();
};
```

[[clang::realtime]]

```
[[clang::realtime]] void run_rt_thread()
{
    get_file_size();
    do_malloc_free();
    do_vector_reserve();
    do_mutex_lock_unlock (m);
    do_read_file();
};
```

CMakeLists.txt:

```
target_compile_options(example PUBLIC
    -fsanitize=realtime)

target_link_options(example PUBLIC
    -fsanitize=realtime)
```

RTSan - Realtime Sanitizer

github.com/realtime-sanitizer/radsan

RADSan:
A REAL TIME-SAFETY SANITIZER

ADC23

ALI BARKER
DAVID TREVELYAN

Memory Allocation

malloc, calloc

realloc, reallocf,

valloc, alligned_alloc

free

posix_memalign

Threads & Sleep

pthread_create, pthread_join

pthread_mutex_lock, pthread_mutex_unlock

pthread_cond_signal, pthread_cond_broadcast etc…

pthread_rwlock_rdlock, pthread_rwlock_unlock etc…

OSSpinLockLock, os_unfair_lock_lock

sleep, usleep, nanosleep

Filesystem, Streams

open, openat, creat, close,

fopen, fopenat, fclose, fread, fwrite

puts, fputs

Sockets

socket

send, sendto, sendmsg

recv, recvfrom, recvmsg

shutdown

```
[[clang::realtime]] void run_rt_thread()
{
    get_file_size();
    do_malloc_free();
    do_vector_reserve();
    do_mutex_lock_unlock (m);
    do_read_file();
};
```

CMakeLists.txt:

```
target_compile_options(example PUBLIC
    -fsanitize=realtime
)

target_link_options(example PUBLIC
    -fsanitize=realtime
)
```

```
Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
    #0 0xaaaad2a1b56c in radsan::printStackTrace() /llvm-project/compiler-rt/lib/radsan/radsan_stack.cpp:36:3
    #1 0xaaaad2a1b2ec in printDiagnostics /llvm-project/compiler-rt/lib/radsan/radsan_context.cpp:57:3
    #2 0xaaaad2a1b2ec in radsan::Context::expectNotRealtime(char const*) /llvm-project/compiler-rt/lib/radsan/radsan_context.cpp:40:5
    #3 0xaaaad2a1bde4 in expectNotRealtime /llvm-project/compiler-rt/lib/radsan/radsan_interceptors.cpp:29:29
    #4 0xaaaad2a1bde4 in free /llvm-project/compiler-rt/lib/radsan/radsan_interceptors.cpp:231:5
    #5 0xaaaad2a404f0 in std::unique_ptr<std::filesystem::__cxx11::path::_List::_Impl, std::filesystem::__cxx11::path::_List::_Impl_deleter>::~unique_ptr() /usr/lib/gcc/aarch64-linux
    #6 0xaaaad2a40460 in std::filesystem::__cxx11::path::_List::~_List() /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/c++/12/bits/fs_path.h:688:24
    #7 0xaaaad2a3fe6c in std::filesystem::__cxx11::path::~path() /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/c++/12/bits/fs_path.h:354:21
    #8 0xaaaad2a3f6c8 in do_read_file() /my_repo/main.cpp:143:30
    #9 0xaaaad2a3f32c in run_rt_thread() /my_repo/main.cpp:35:5
    #10 0xaaaad2a3fba8 in main::$_0::operator()() const /my_repo/main.cpp:43:25
    #11 0xaaaad2a3fb88 in void std::__invoke_impl<void, main::$_0>(std::__invoke_other, main::$_0&&) /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/c++/12/bits/invoke.h:61:14
    #12 0xaaaad2a3fb3c in std::__invoke_result<main::$_0>::type std::__invoke<main::$_0>(main::$_0&&) /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/c++/12/bits/invoke.h:96:14
    #13 0xaaaad2a3fb14 in void std::thread::_Invoker<std::tuple<main::$_0>>::_M_invoke<0ul>(std::_Index_tuple<0ul>) /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/c++/12/bits/
    #14 0xaaaad2a3fae8 in std::thread::_Invoker<std::tuple<main::$_0>>::operator()() /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/c++/12/bits/std_thread.h:286:11
    #15 0xaaaad2a3fa4c in std::thread::_State_impl<std::thread::_Invoker<std::tuple<main::$_0>>>::_M_run() /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/c++/12/bits/std_threa
    #16 0xffffa66631f8  (/lib/aarch64-linux-gnu/libstdc++.so.6+0xd31f8) (BuildId: a012b2bb77110e84b266cd7425b50e57427abb02)
    #17 0xffffa635d5c4  (/lib/aarch64-linux-gnu/libc.so.6+0x7d5c4) (BuildId: 317350dd9c806d5dfe9358afcdca6bf56f2b0a54)
    #18 0xffffa63c5d98  (/lib/aarch64-linux-gnu/libc.so.6+0xe5d98) (BuildId: 317350dd9c806d5dfe9358afcdca6bf56f2b0a54)
```

```
Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
    #0 0xaaaad2a1b56c in radsan::printStackTrace() /llvm-project/compiler-rt/lib/radsan/radsan_stack.cpp:36:3
    #1 0xaaaad2a1b2ec in printDiagnostics /llvm-project/compiler-rt/lib/radsan/radsan_context.cpp:57:3
    #2 0xaaaad2a1b2ec in radsan::Context::expectNotRealtime(char const*) /llvm-project/compiler-rt/lib/radsan/radsan_context.cpp:40
    #3 0xaaaad2a1bde4 in expectNotRealtime /llvm-project/compiler-rt/lib/radsan/radsan_interceptors.cpp:29:29
    #4 0xaaaad2a1bde4 in free /llvm-project/compiler-rt/lib/radsan/radsan_interceptors.cpp:231:5
    #5 0xaaaad2a404f0 in std::unique_ptr<std::filesystem::__cxx11::path::_List::_Impl, std::filesystem::__cxx11::path::_List::_Impl_deleter>::~unique_ptr()
    #6 0xaaaad2a40460 in std::filesystem::__cxx11::path::_List::~_List() /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/c++/12/bits/fs_path.h:688:24
    #7 0xaaaad2a3fe6c in std::filesystem::__cxx11::path::~path() /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/c++/12/bits/fs_path.h:354:21
    #8 0xaaaad2a3f6c8 in do_read_file() /my_repo/main.cpp:143:30
    #9 0xaaaad2a3f32c in run_rt_thread() /my_repo/main.cpp:35:5
   #10 0xaaaad2a3fba8 in main::$_0::operator()() const /my_repo/main.cpp:43:25
   #11 0xaaaad2a3fb88 in void std::__invoke_impl<void, main::$_0>(std::__invoke_other, main::$_0&&) /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/
   #12 0xaaaad2a3fb3c in std::__invoke_result<main::$_0>::type std::__invoke<main::$_0>(main::$_0&&) /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include
   #13 0xaaaad2a3fb14 in void std::thread::_Invoker<std::tuple<main::$_0>>::_M_invoke<0ul>(std::_Index_tuple<0ul>) /usr/lib/gcc/aarch64-linux-gnu/12/../..
   #14 0xaaaad2a3fae8 in std::thread::_Invoker<std::tuple<main::$_0>>::operator()() /usr/lib/gcc/aarch64-linux-gnu/12/../../../../include/c++/12/bits/std_
   #15 0xaaaad2a3fa4c in std::thread::_State_impl<std::thread::_Invoker<std::tuple<main::$_0>>>::_M_run() /usr/lib/gcc/aarch64-linux-gnu/12/../../../../in
   #16 0xffffa66631f8  (/lib/aarch64-linux-gnu/libstdc++.so.6+0xd31f8) (BuildId: a012b2bb77110e84b266cd7425b50e57427abb02)
   #17 0xffffa635d5c4  (/lib/aarch64-linux-gnu/libc.so.6+0x7d5c4) (BuildId: 317350dd9c806d5dfe9358afcdca6bf56f2b0a54)
   #18 0xffffa63c5d98  (/lib/aarch64-linux-gnu/libc.so.6+0xe5d98) (BuildId: 317350dd9c806d5dfe9358afcdca6bf56f2b0a54)
```

```
Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
    #9 0xaaaad2a3f3b0 in get_file_size() /my_repo/main.cpp:115:27

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
    #8 0xaaaad2a3f3d8 in get_file_size() /my_repo/main.cpp:118:1

Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
    #5 0xaaaad2a3f41c in do_malloc_free() /my_repo/main.cpp:122:14

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
    #5 0xaaaad2a3f428 in do_malloc_free() /my_repo/main.cpp:123:5

Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
    #11 0xaaaad2a3f458 in do_vector_reserve() /my_repo/main.cpp:129:7

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
    #11 0xaaaad2a3f464 in do_vector_reserve() /my_repo/main.cpp:130:1

Real-time violation: intercepted call to real-time unsafe function `pthread_mutex_lock` in real-time context! Stack trace:
    #9 0xaaaad2a3f4b0 in do_mutex_lock_unlock(std::mutex&) /my_repo/main.cpp:135:22

Real-time violation: intercepted call to real-time unsafe function `pthread_mutex_unlock` in real-time context! Stack trace:
    #9 0xaaaad2a3f4b8 in do_mutex_lock_unlock(std::mutex&) /my_repo/main.cpp:136:1

Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
    #9 0xaaaad2a3f4ec in do_read_file() /my_repo/main.cpp:142:18

Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
    #13 0xaaaad2a3f544 in do_read_file() /my_repo/main.cpp:147:27

Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
    #10 0xaaaad2a3f560 in do_read_file() /my_repo/main.cpp:150:23

Real-time violation: intercepted call to real-time unsafe function `malloc` in real-time context! Stack trace:
    #10 0xaaaad2a3f560 in do_read_file() /my_repo/main.cpp:150:23

Real-time violation: intercepted call to real-time unsafe function `fwrite` in real-time context! Stack trace:
    #7 0xaaaad2a3f5bc in do_read_file() /my_repo/main.cpp:155:23

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
    #8 0xaaaad2a3f668 in do_read_file() /my_repo/main.cpp:159:14

Real-time violation: intercepted call to real-time unsafe function `fclose` in real-time context! Stack trace:
    #8 0xaaaad2a3f668 in do_read_file() /my_repo/main.cpp:159:14

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
    #9 0xaaaad2a3f668 in do_read_file() /my_repo/main.cpp:159:14

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
    #11 0xaaaad2a3f67c in do_read_file() /my_repo/main.cpp:160:5

Real-time violation: intercepted call to real-time unsafe function `free` in real-time context! Stack trace:
    #8 0xaaaad2a3f6c8 in do_read_file() /my_repo/main.cpp:143:30
```

`malloc`
    get_file_size()

`free`
    get_file_size()

`malloc`
    do_malloc_free()

`free`
    do_malloc_free()

`malloc`
    do_vector_reserve()

`free`
    do_vector_reserve()

`pthread_mutex_lock`
    do_mutex_lock_unlock(std::mutex&)

`pthread_mutex_unlock`
    do_mutex_lock_unlock(std::mutex&)

`malloc`
    do_read_file()

`malloc`
    do_read_file()

`malloc`
    do_read_file()

`malloc`
    do_read_file()

`fwrite`
    do_read_file()

`free`
    do_read_file()

`fclose`
    do_read_file()

`free`
    do_read_file()

`free`
    do_read_file()

`free`
    do_read_file()

Disabling RTSan
(From GitHub)

```
__attribute__((no_sanitize("realtime")))
void mutex_unlock_uncontended (std::mutex& m)
{
    m.unlock();
}

[[clang::realtime]] float process (float x)
{
    ...
    mutex_unlock_uncontended(m); // I know this is always uncontended, thus real-time safe!
    ...
}
```

RTSan

✓ Easy to use

✓ Explicit real-time scopes

✓ Easy to run on CI

✓ Good level of detail

✓ Can be easily disabled

✓ Very complete

✓ Includes 3rd party libs

x Clang fork

x No Windows

x Requires code changes (attributes)

RTSan Future

- Merged to Clang 🙏
- Customisable detection lists
- Opt-in in own code:

```
[[clang::blocking]] my_spin_lock::lock() { … }
```

- Windows support
- Static checking?

Static checking?

```
[[clang::realtime]] void run_rt_thread()
{
    my_safe_function();
    [[clang::realtime_ignore]] unsafe_library_function();
    // library doesn’t have attributes added
}


[[clang::realtime]] void my_safe_function()
{
   // real-time-safe code
}
```

noalloc, nolock

https://discourse.llvm.org/t/rfc-nolock-and-noalloc-attributes

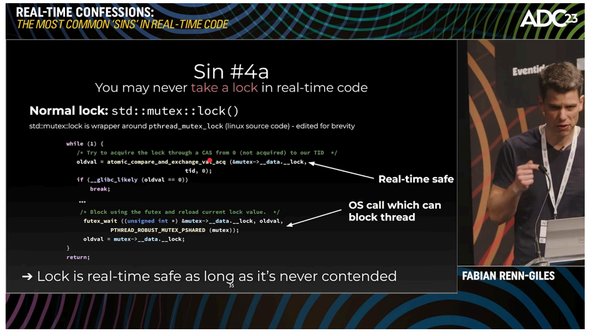
- Similar to `noexcept`
  - `[[clang::noalloc]]`
    - **`noexcept`** + never allocate memory on the heap
  - `[[clang::nolock]]`
    - **`noalloc`** + never block on a lock
- Can be inferred in some cases
  - E.g. function body is visible
- Verified statically at compile time 😎

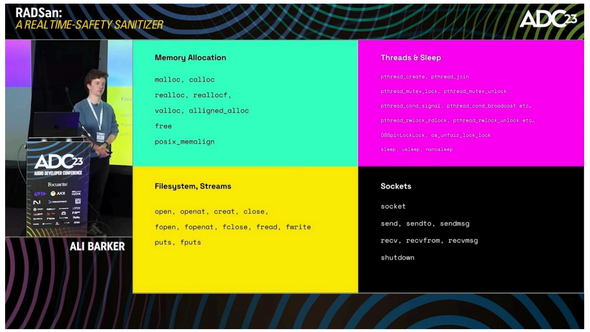
```
void noLockFunction() [[clang::nolock]];
void noAllocFunction() [[clang::noalloc]];

struct widget
{
    void noLockMethod() [[clang::nolock]];
};

void myFunction() [[clang::nolock]]
{
    noLockFunction();
    noAllocFunction(); // Error!

    widget w;
    w.noLockMethod();
}
```

	GUI (System Trace)	cli (dtrace)	code	interpose	RTSan
Easy to use?	✅/⚠️	⚠️	✅	⚠️	✅
Clear?	❌	✅	✅	✅	✅
🧵Filterable?	⚠️	⚠️	✅	✅	✅
CI?	❌	⚠️	✅	✅	✅
Portable?	❌	❌	✅	⚠️	⚠️
System calls?	✅	✅	❌	⚠️	⚠️
Malloc/free?	⚠️	✅	❌	✅	✅
Lock/unlock?	✅	✅	⚠️	✅	✅
3rd party code?	✅	✅	❌	✅	✅
Notes	Different tools for different tasks	Requires disabling SIP		No raw/inline context switches/syscalls	No raw/inline context switches/syscalls

Thanks

- Timur Doumler
  - “*Real-time programming with the standard library*”
- Fabian Renn-Giles
  - “Real-time confessions”
- David Trevelyan, Ali Barker & Chris Apple
  - “RADSan: A real-time safety sanitizer”
- Doug Wyatt
  - dtrace script, `noalloc`, `nolock`





Catching Real-Time Safety Violations

David Rowland

@drowaudio

Questions?

Slides/video:

drowaudio.github.io/presentations