SDC 20

BY Developers FOR Developers

Storage Developer Conference
September 22-23, 2020

Tracing and Visualizing FS Internals with eBPF Superpowers

Suchakra Sharma
ShiftLeft Inc.

Hani Nemati
Microsoft

Suchakra Sharma

PhD, École Polytechnique de Montréal

Staff Scientist
ShiftLeft Inc

Hani Nemati

PhD, École Polytechnique de Montréal

Software Engineer (Kernel)
Microsoft

Agenda

- **FS Internals 101**
- **Journey of a `read()`**
  - Tracing Control-Flow via `ftrace`
- **FS and Performance**
  - Probing the system via `/proc`
  - BCC and `bpftrace` Tool Collection
- **Building tools with eBPF**
  - Case Study: Read-ahead
  - DIY new BPF tool - `readaheadstat`
  - Visualizing Performance

FS Internals 101

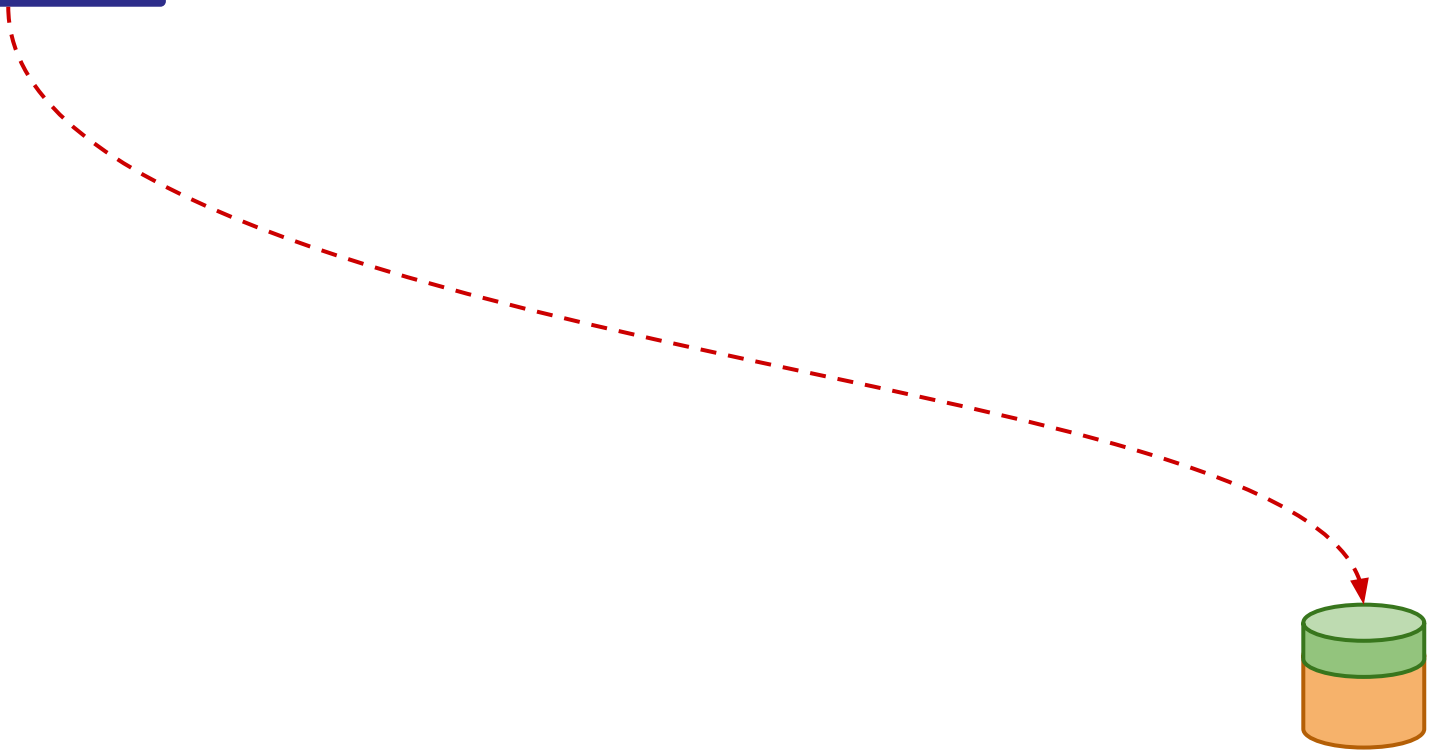
Why Filesystem?

- **Keep track of things**
  - What many files are open or being read?
  - Which user/process is operating with files?
  - How many files can be created?
  - What data is most frequently accessed
- **Structure**
  - How are they structured?
  - What happens when we move them across?
- **Abstraction and Uniformity**
  - What if some of our files are on a network storage?
  - What if we copy a file from two different storage devices?

FS Internals

APP

```
int fd = open(“foo”, R)
```



FS Internals

APP

```
int fd = open("foo", R)
```

open()

sys_open

Syscall Interface

vfs_open

Virtual File System

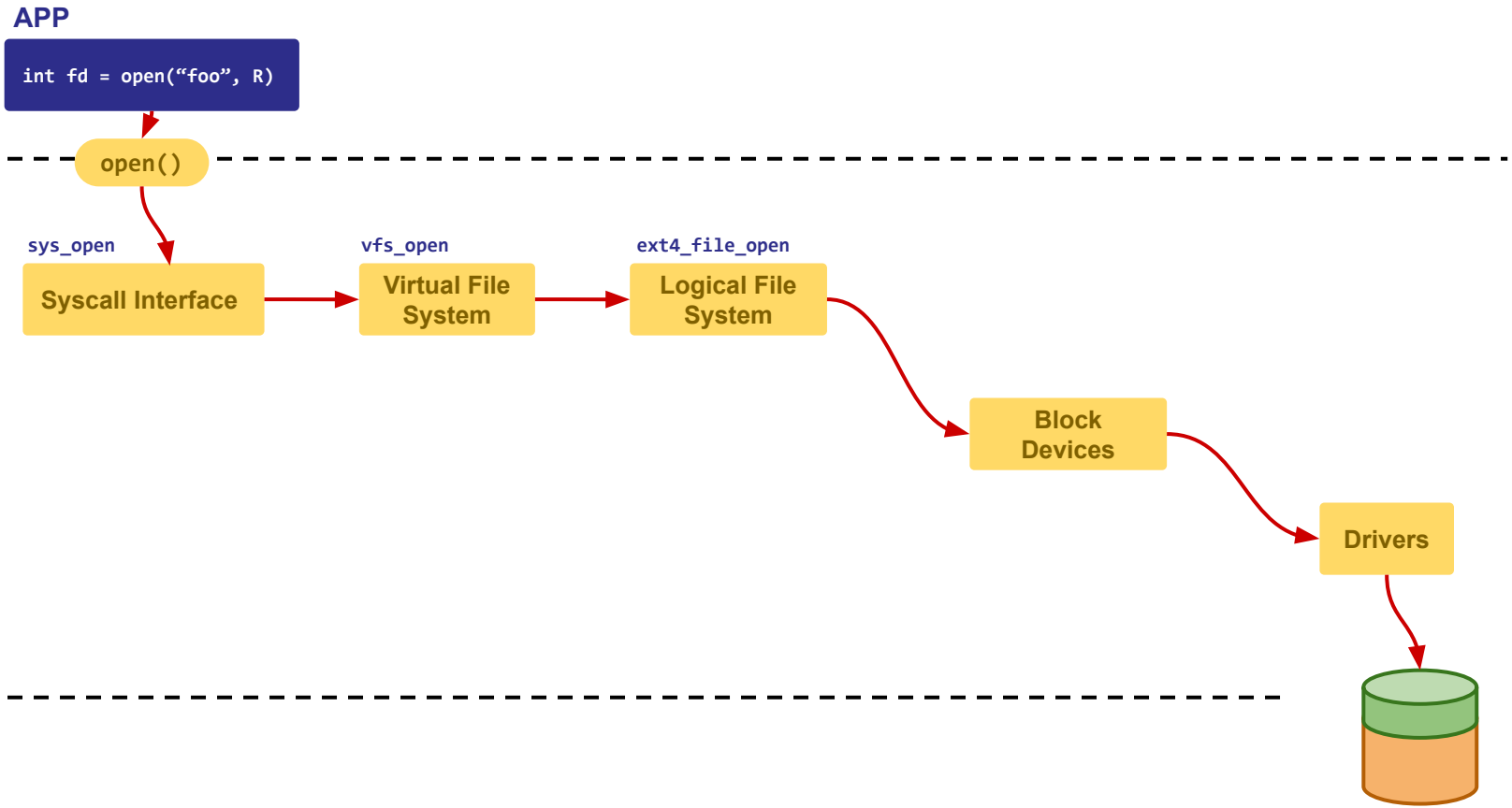
ext4_file_open

Logical File System

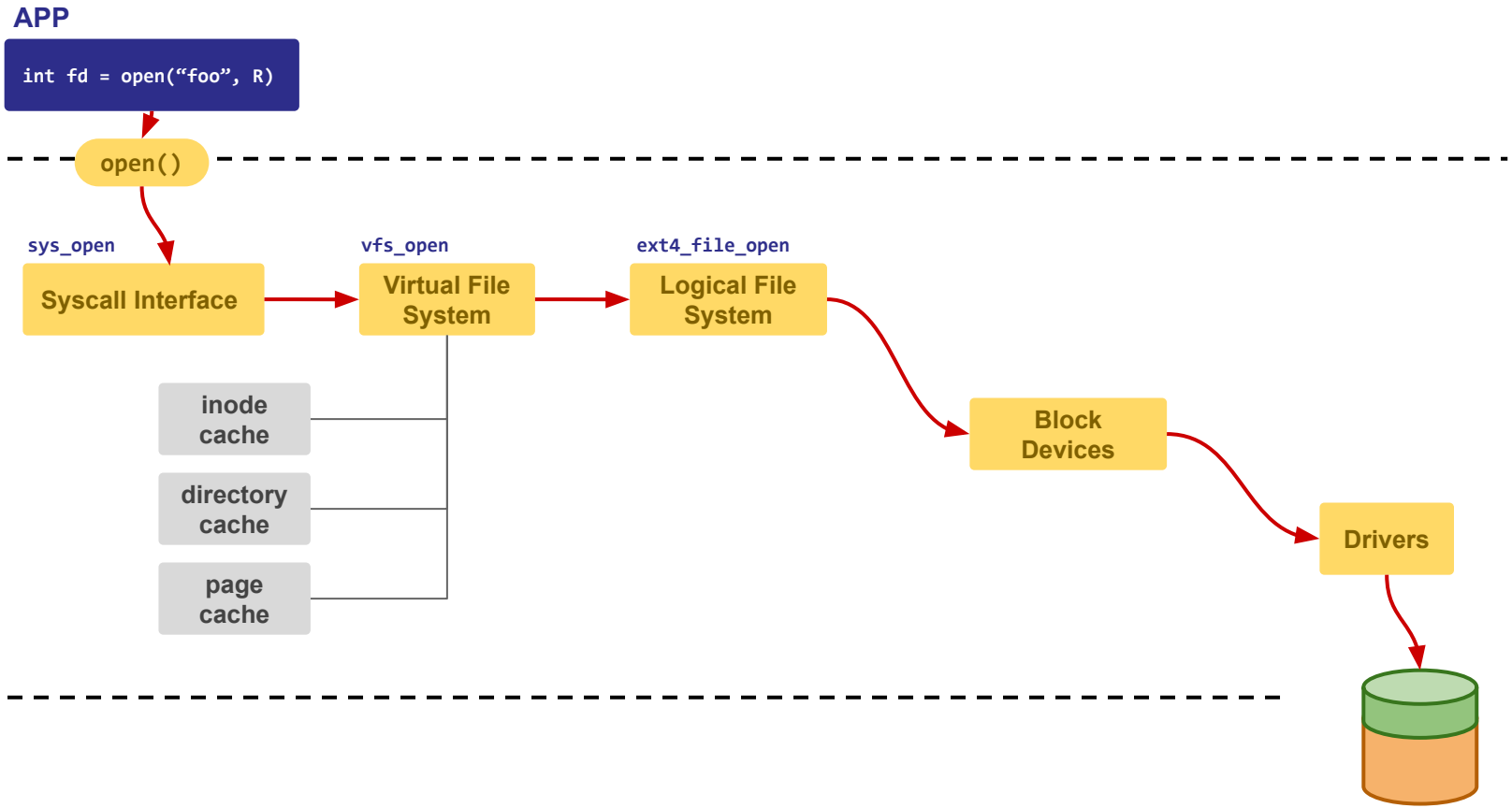
Block Devices

Drivers

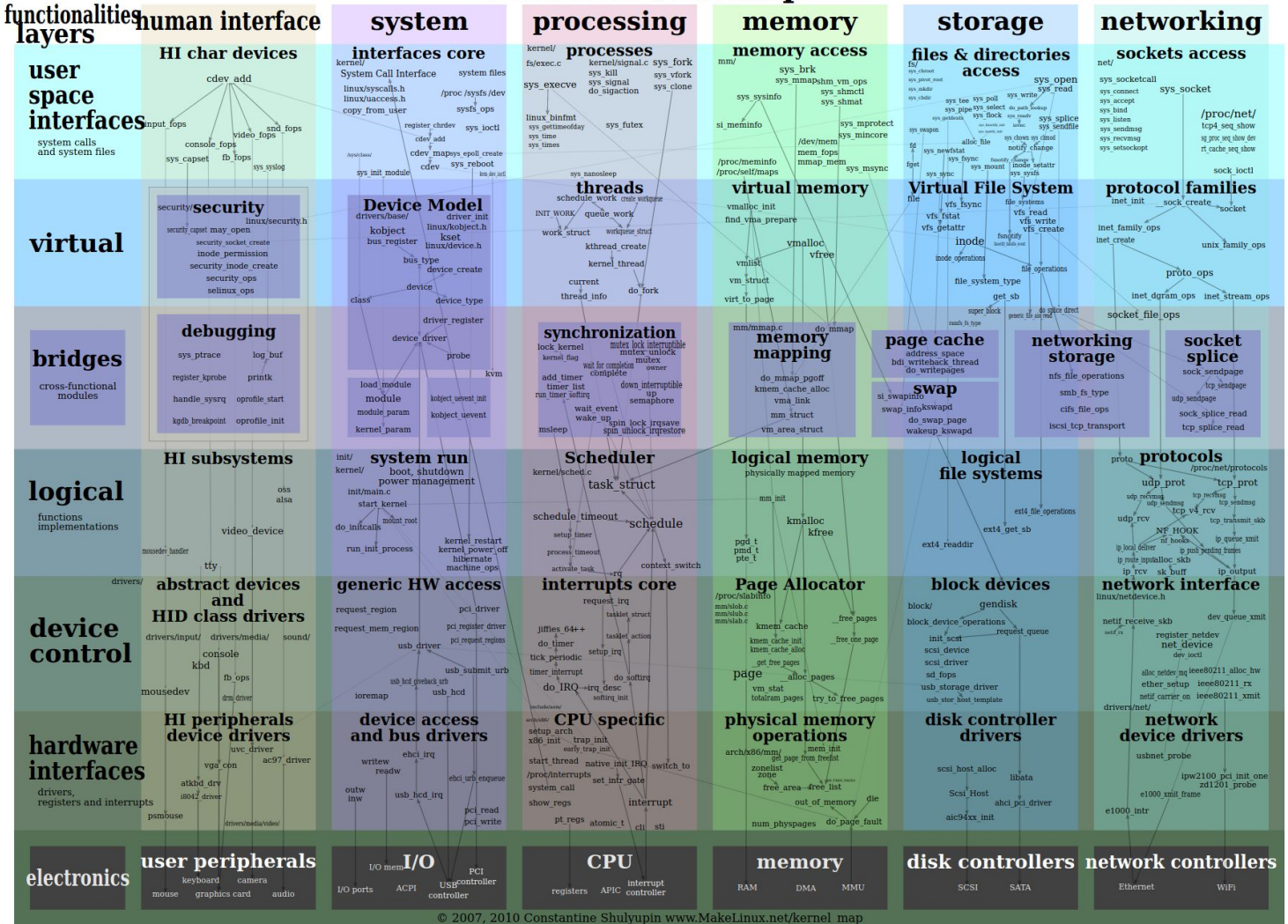
FS Internals



FS Internals

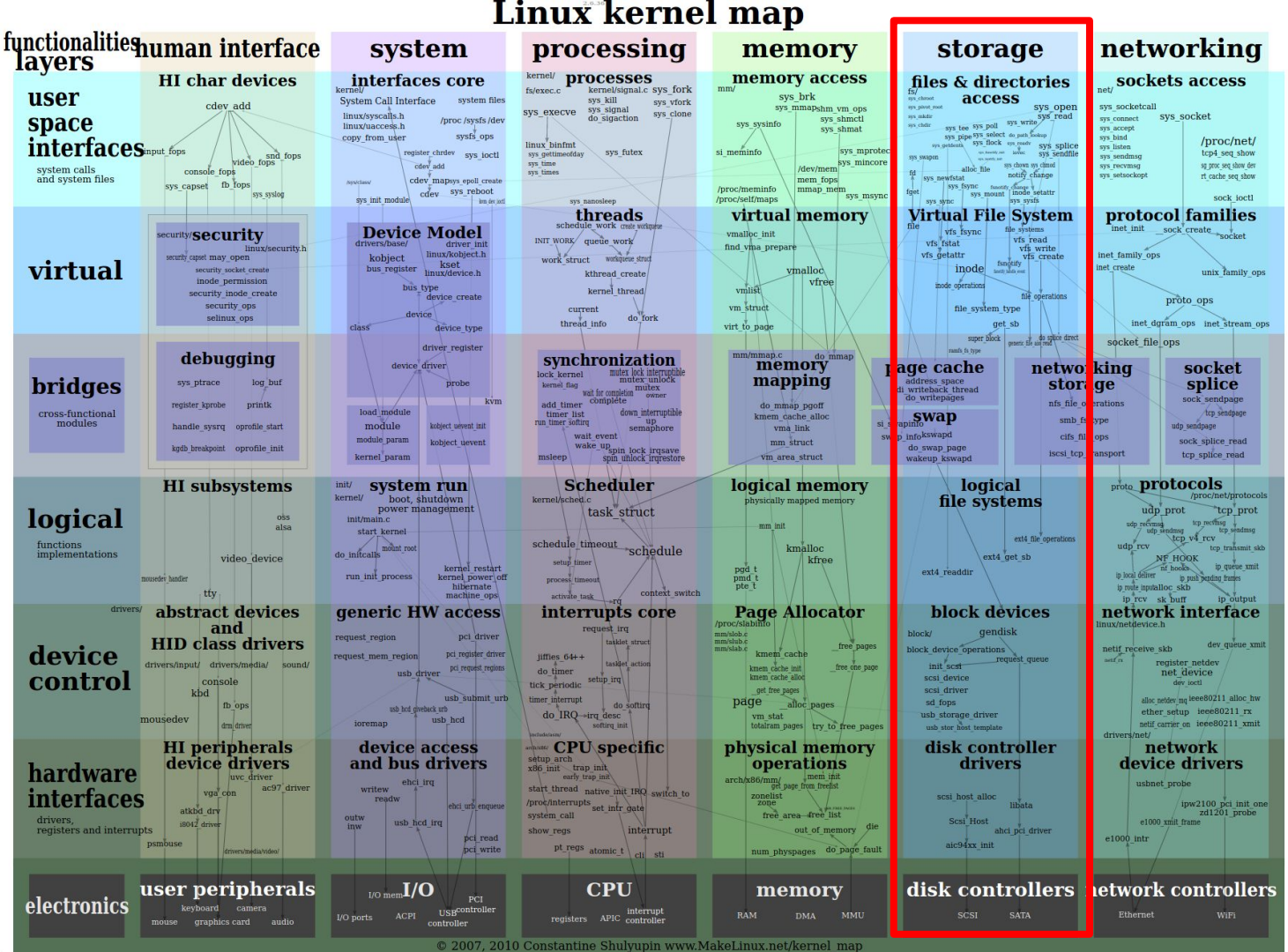


Linux kernel map



2020

SDC 20


Linux kernel map
functionalities
layers
human interface
system
processing
memory
storage
networking
user space interfaces
system calls and system files
HI char devices
interfaces core
processes
memory access
files & directories access
sockets access
virtual
security
Device Model
threads
virtual memory
Virtual File System
protocol families
bridges
cross-functional modules
debugging
synchronization
memory mapping
page cache
swap
networking storage
socket splice
logical
functions implementations
HI subsystems
system run
Scheduler
logical memory
logical file systems
protocols
device control
abstract devices and HID class drivers
generic HW access
interrupts core
Page Allocator
block devices
network interface
hardware interfaces
drivers, registers and interrupts
HI peripherals device drivers
device access and bus drivers
CPU specific
physical memory operations
disk controller drivers
network device drivers
electronics
user peripherals
I/O
CPU
memory
disk controllers
network controllers
© 2007, 2010 Constantine Shulyupin www.MakeLinux.net/kernel_map

2020

Tracing a `read()` with `ftrace`

```
$ sudo trace-cmd record -p function -P 2535
$ sudo trace-cmd report
```

Tracing a `read()` with `ftrace`

```
$ sudo trace-cmd record -p function -P 2535
$ sudo trace-cmd report
```

```
finish_task_switch
down_read
   _cond_resched
      rcu_all_qs
tty_audit_add_data
__check_object_size
   check_stack_object
   __virt_addr_valid
tty_audit_push
n_tty_kick_worker
tty_wakeup
   __wake_up
      __wake_up_common_lock
         _raw_spin_lock_irqsave
         __wake_up_common
         __lock_text_start
n_tty_kick_worker
up_read
remove_wait_queue
   _raw_spin_lock_irqsave
   __lock_text_start
mutex_unlock
tty_ldisc_deref
   ldsem_up_read
ktime_get_real_seconds
fsnotify_parent
fsnotify
fpregs_assert_state_consistent
switch_fpu_return
do_page_fault
down_read_trylock
_cond_resched
   rcu_all_qs
find_vma
   vmacache_find
handle_mm_fault
   mem_cgroup_from_task
   __count_memcg_events
   __handle_mm_fault
      pmd_devmap_trans_unstable
      do_fault
         filemap_map_pages
            PageHuge
            alloc_set_pte
               pmd_devmap_trans_unstable
               _raw_spin_lock
               page_add_file_rmap
                  lock_page_memcg
                  unlock_page_memcg
                     __unlock_page_memcg
            unlock_page
            PageHuge
            alloc_set_pte
               page_add_file_rmap
                  lock_page_memcg
                  unlock_page_memcg
                     __unlock_page_memcg
            unlock_page
            PageHuge
            alloc_set_pte
               page_add_file_rmap
                  lock_page_memcg
```

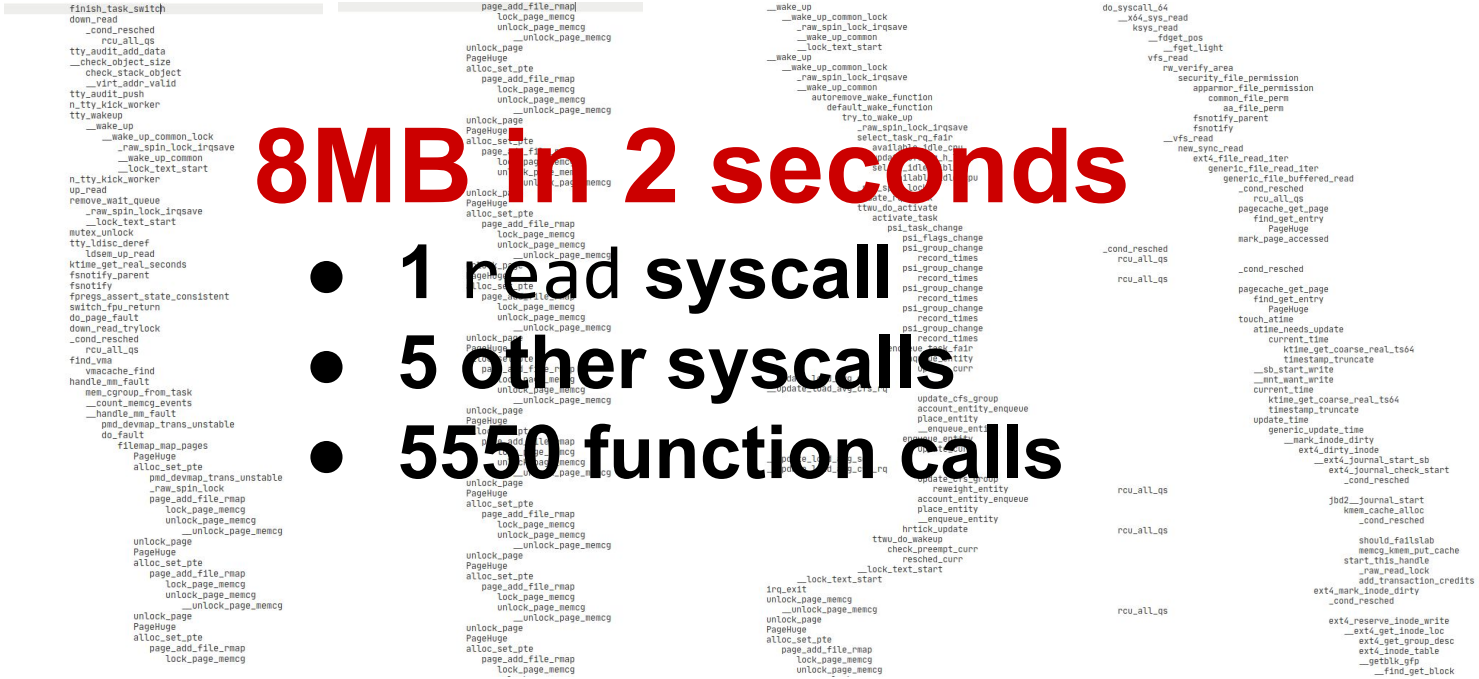
```
   page_add_file_rmap
      lock_page_memcg
      unlock_page_memcg
         __unlock_page_memcg
unlock_page
PageHuge
alloc_set_pte
   page_add_file_rmap
      lock_page_memcg
      unlock_page_memcg
         __unlock_page_memcg
unlock_page
PageHuge
alloc_set_pte
   page_add_file_rmap
      lock_page_memcg
      unlock_page_memcg
         __unlock_page_memcg
unlock_page
PageHuge
alloc_set_pte
   page_add_file_rmap
      lock_page_memcg
      unlock_page_memcg
         __unlock_page_memcg
unlock_page
PageHuge
alloc_set_pte
   page_add_file_rmap
      lock_page_memcg
      unlock_page_memcg
         __unlock_page_memcg
unlock_page
PageHuge
alloc_set_pte
   page_add_file_rmap
      lock_page_memcg
      unlock_page_memcg
         __unlock_page_memcg
unlock_page
PageHuge
alloc_set_pte
   page_add_file_rmap
      lock_page_memcg
      unlock_page_memcg
         __unlock_page_memcg
unlock_page
PageHuge
alloc_set_pte
   page_add_file_rmap
      lock_page_memcg
      unlock_page_memcg
         __unlock_page_memcg
unlock_page
PageHuge
alloc_set_pte
   page_add_file_rmap
      lock_page_memcg
      unlock_page_memcg
         __unlock_page_memcg
unlock_page
PageHuge
alloc_set_pte
   page_add_file_rmap
      lock_page_memcg
```

```
__wake_up
   __wake_up_common_lock
      _raw_spin_lock_irqsave
      __wake_up_common
      __lock_text_start
__wake_up
   __wake_up_common_lock
      _raw_spin_lock_irqsave
      __wake_up_common
         autoremove_wake_function
            default_wake_function
               try_to_wake_up
                  _raw_spin_lock_irqsave
                  select_task_rq_fair
                     available_idle_cpu
                     update_cfs_rq_h_load
                     select_idle_sibling
                        available_idle_cpu
                  _raw_spin_lock
                  update_rq_clock
                  ttwu_do_activate
                     activate_task
                        psi_task_change
                           psi_flags_change
                           psi_group_change
                              record_times
                           psi_group_change
                              record_times
                           psi_group_change
                              record_times
                           psi_group_change
                              record_times
                           psi_group_change
                              record_times
                        enqueue_task_fair
                           enqueue_entity
                              update_curr
__update_load_avg_se
__update_load_avg_cfs_rq
                              update_cfs_group
                              account_entity_enqueue
                              place_entity
                              __enqueue_entity
                           enqueue_entity
                              update_curr
__update_load_avg_se
__update_load_avg_cfs_rq
                              update_cfs_group
                                 reweight_entity
                              account_entity_enqueue
                              place_entity
                              __enqueue_entity
                           hrtick_update
                     ttwu_do_wakeup
                        check_preempt_curr
                           resched_curr
                  __lock_text_start
      __lock_text_start
irq_exit
unlock_page_memcg
   __unlock_page_memcg
unlock_page
PageHuge
alloc_set_pte
   page_add_file_rmap
      lock_page_memcg
      unlock_page_memcg
```

```
do_syscall_64
   __x64_sys_read
      ksys_read
         __fdget_pos
            __fget_light
         vfs_read
            rw_verify_area
               security_file_permission
                  apparmor_file_permission
                     common_file_perm
                        aa_file_perm
                  fsnotify_parent
                  fsnotify
            __vfs_read
               new_sync_read
                  ext4_file_read_iter
                     generic_file_read_iter
                        generic_file_buffered_read
                           _cond_resched
                              rcu_all_qs
                           pagecache_get_page
                              find_get_entry
                                 PageHuge
                           mark_page_accessed
_cond_resched
   rcu_all_qs
                           _cond_resched

   rcu_all_qs
                           pagecache_get_page
                              find_get_entry
                                 PageHuge
                           touch_atime
                              atime_needs_update
                                 current_time
                                    ktime_get_coarse_real_ts64
                                    timestamp_truncate
                              __sb_start_write
                              __mnt_want_write
                              current_time
                                 ktime_get_coarse_real_ts64
                                 timestamp_truncate
                              update_time
                                 generic_update_time
                                    __mark_inode_dirty
                                       ext4_dirty_inode
                                          __ext4_journal_start_sb
                                             ext4_journal_check_start
                                                _cond_resched
   rcu_all_qs

                                             jbd2__journal_start
                                                kmem_cache_alloc
                                                   _cond_resched
   rcu_all_qs

                                                   should_failslab
                                                   memcg_kmem_put_cache
                                                start_this_handle
                                                   _raw_read_lock
                                                   add_transaction_credits
                                          ext4_mark_inode_dirty
                                             _cond_resched
   rcu_all_qs

                                             ext4_reserve_inode_write
                                                __ext4_get_inode_loc
                                                   ext4_get_group_desc
                                                   ext4_inode_table
                                                   __getblk_gfp
                                                      __find_get_block
```

Tracing a `read()` with `ftrace`

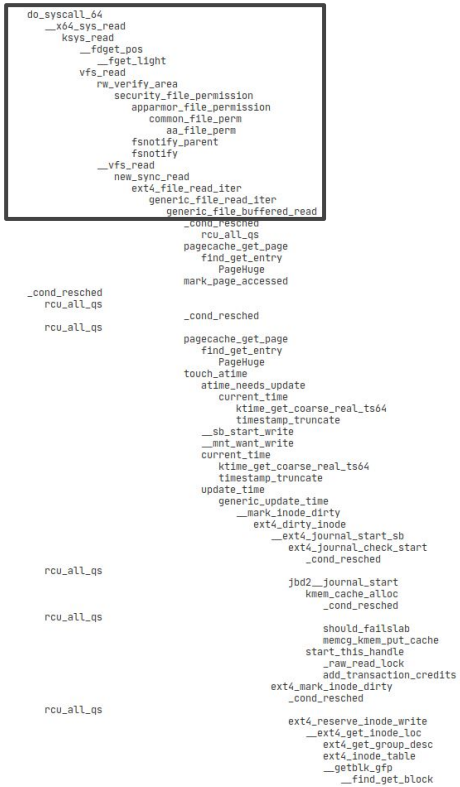
```
$ sudo trace-cmd record -p function -P 2535
$ sudo trace-cmd report
```



8MB in 2 seconds

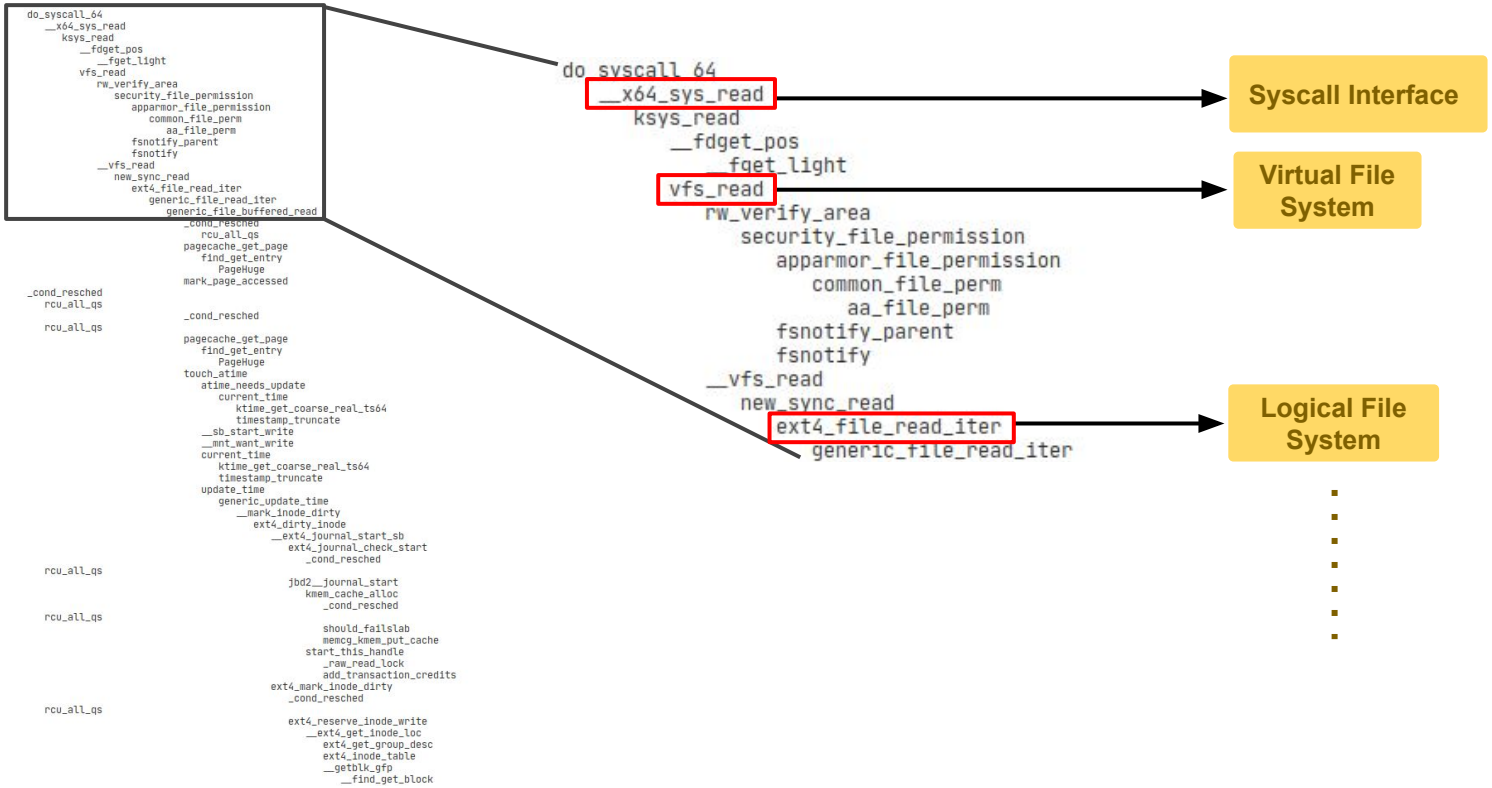
- 1 read syscall
- 5 other syscalls
- 5550 function calls

Tracing a `read()` with `ftrace`



Tracing a `read()` with `ftrace`

Tracing a `read()` with `ftrace`



Tracing a `read()` with `ftrace`

```
example-2535  [007] 1580764.896732: function:             do_syscall_64
example-2535  [007] 1580764.896732: function:                __x64_sys_read
example-2535  [007] 1580764.896732: function:                   ksys_read
example-2535  [007] 1580764.896733: function:                      __fdget_pos
example-2535  [007] 1580764.896733: function:                         __fget_light
example-2535  [007] 1580764.896734: function:                      vfs_read
                                 .
                                 .
                                 .
example-2535  [007] 1580764.896770: function:                   __wake_up_common_lock
example-2535  [007] 1580764.896770: function:                      _raw_spin_lock_irqsave
                                 .
                                 .
                                 .
example-2535  [007] 1580764.896794: function:             irq_exit
                                 .
                                 .
                                 .
                                 .
                                 .
                                 .
                                 .
                                 .
                                 .
example-2535  [007] 1580764.896808: function:             __mnt_drop_write
example-2535  [007] 1580764.896808: function:             __sb_end_write
example-2535  [007] 1580764.896809: function:             fsnotify_parent
example-2535  [007] 1580764.896810: function:             fsnotify
example-2535  [007] 1580764.896810: function:             fpregs_assert_state_consistent
```

Tracing a `read()` with `ftrace`

```
example-2535  [007] 1580764.896732:  function:             do_syscall_64
example-2535  [007] 1580764.896732:  function:                __x64_sys_read
example-2535  [007] 1580764.896732:  function:                   ksys_read
example-2535  [007] 1580764.896733:  function:                      __fdget_pos
example-2535  [007] 1580764.896733:  function:                         __fget_light
example-2535  [007] 1580764.896734:  function:                      vfs_read
                                     .
                                     .
                                     .
example-2535  [007] 1580764.896770:  function:                   __wake_up_common_lock
example-2535  [007] 1580764.896770:  function:                      _raw_spin_lock_irqsave
                                     .
                                     .
                                     .
example-2535  [007] 1580764.896794:  function:             irq_exit
                                     .
                                     .
                                     .
                                     .
                                     .
                                     .
                                     .
                                     .
                                     .
example-2535  [007] 1580764.896808:  function:             __mnt_drop_write
example-2535  [007] 1580764.896808:  function:             __sb_end_write
example-2535  [007] 1580764.896809:  function:             fsnotify_parent
example-2535  [007] 1580764.896810:  function:             fsnotify
example-2535  [007] 1580764.896810:  function:             fpregs_assert_state_consistent
```

Tracing a `read()` with `ftrace`

```
example-2535  [007] 1580764.896732:  function:             do_syscall_64
example-2535  [007] 1580764.896732:  function:                __x64_sys_read
example-2535  [007] 1580764.896732:  function:                   ksys_read
example-2535  [007] 1580764.896733:  function:                      __fdget_pos
example-2535  [007] 1580764.896733:  function:                         __fget_light
example-2535  [007] 1580764.896734:  function:                      vfs_read
                                     .
                                     .
                                     .
example-2535  [007] 1580764.896770:  function:                      __wake_up_common_lock
example-2535  [007] 1580764.896770:  function:                         _raw_spin_lock_irqsave
                                     .
                                     .
                                     .
example-2535  [007] 1580764.896794:  function:             irq_exit
                                     .
                                     .
                                     .
                                     .
                                     .
                                     .
                                     .
                                     .
                                     .
example-2535  [007] 1580764.896808:  function:             __mnt_drop_write
example-2535  [007] 1580764.896808:  function:             __sb_end_write
example-2535  [007] 1580764.896809:  function:             fsnotify_parent
example-2535  [007] 1580764.896810:  function:             fsnotify
example-2535  [007] 1580764.896810:  function:             fpregs_assert_state_consistent
```

Syscall Enter

IRQ Entry

IRQ Exit

Syscall Exit

FS and Performance

What can /proc tell us?

```
# cat /proc/diskstats

 259       0 nvme0n1 1823879 1141588 58777929 328845 3512978 2244953 151315050
7397322 0 2759684 7859940 88819 0 543261592 26313 273795 107459
 259       1 nvme0n1p1 1443 0 27072 2273 2 0 2 34 0 392 2310 4 0 1034184 1 0 0
 259       2 nvme0n1p2 573 398 59794 77 367 115 151352 415 0 776 525 91 0
645136 33 0 0
 259       3 nvme0n1p3 1820929 1141190 58667503 326372 3238845 2244838
151163696 7287293 0 2747016 7639943 88724 0 541582272 26277 0 0
```

[1]https://www.kernel.org/doc/Documentation/iostats.txt

What can /proc tell us?

```
# cat /proc/diskstats

 259       0 nvme0n1 1823879 1141588 58777929 328845 3512978 2244953 151315050
7397322 0 2759684 7859940 88819 0 543261592 26311 273795 107459
 259       1 nvme0n1p1 1443 0 27072 2273 2 0 2 34 0 392 2310 4 0 1034184 1 0 0
 259       2 nvme0n1p2 573 398 59794 77 367 115 151352 415 0 776 525 91 0
645136 33 0 0
 259       3 nvme0n1p3 1820929 1141190 58667503 326372 3238845 2244838
151163696 7287293 0 2747016 7639943 88724 0 541582272 26277 0 0
```

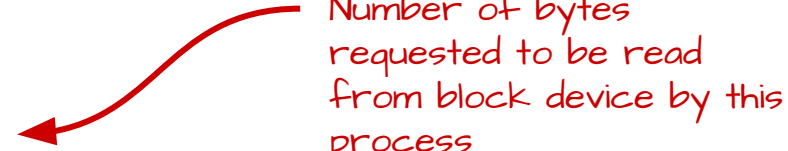
Milliseconds spent in all IOs

Milliseconds spent in all disk reads

[1]https://www.kernel.org/doc/Documentation/iostats.txt

What can /proc tell us?

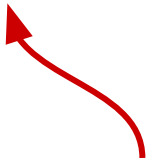
```
# cat /proc/1475/io

rchar: 723810543
wchar: 1609216447277
syscr: 27156292
syscw: 142054696
read_bytes: 30699520
write_bytes: 6516736
cancelled_write_bytes: 5160960
```



[1]https://www.kernel.org/doc/Documentation/filesystems/proc.txt

Building tools over /proc and /sys

```
# iostat -d nvme0n1

Linux 5.7.1-050701-generic (isengard)     09/11/2020     _x86_64_    (8 CPU)

Device            tps    kB_read/s    kB_wrtn/s     kB_read     kB_wrtn
nvme0n1          7.61        42.62       109.59    30521808    78476133
```



Total kilobytes written per second

```
# iotop

(Exercise left for the reader)
```

More Tracing Techniques

Targeted Analysis

```
example-2535  [007] 1580764.896732: function:            do_syscall_64
example-2535  [007] 1580764.896732: function:               __x64_sys_read
example-2535  [007] 1580764.896732: function:                  ksys_read
example-2535  [007] 1580764.896733: function:                     __fdget_pos
example-2535  [007] 1580764.896733: function:                        __fget_light
example-2535  [007] 1580764.896734: function:                     vfs_read
```

Syscall Enter

```
example-2535  [007] 1580764.896770: function:               __wake_up_common_lock
example-2535  [007] 1580764.896770: function:                  _raw_spin_lock_irqsave
```

IRQ Entry

```
example-2535  [007] 1580764.896794: function:            irq_exit
```

IRQ Exit

```
example-2535  [007] 1580764.896808: function:            __mnt_drop_write
example-2535  [007] 1580764.896808: function:            __sb_end_write
example-2535  [007] 1580764.896809: function:            fsnotify_parent
example-2535  [007] 1580764.896810: function:            fsnotify
example-2535  [007] 1580764.896810: function:            fpregs_assert_state_consistent
```

Syscall Exit

[1]https://github.com/iovisor/bpftrace/tree/master/tools

More Tracing Techniques

Targeted Analysis


```
example-2535  [007] 1580764.896732: function:            do_syscall_64
example-2535  [007] 1580764.896732: function:               __x64_sys_read
example-2535  [007] 1580764.896732: function:                  ksys_read
example-2535  [007] 1580764.896733: function:                     __fdget_pos
example-2535  [007] 1580764.896733: function:                        __fget_light
example-2535  [007] 1580764.896734: function:                     vfs_read
```

Syscall Enter

```
example-2535  [007] 1580764.896770: function:                     __wake_up_common_lock
example-2535  [007] 1580764.896770: function:                        _raw_spin_lock_irqsave
```

IRQ Entry

```
example-2535  [007] 1580764.896794: function:            irq_exit
```

IRQ Exit

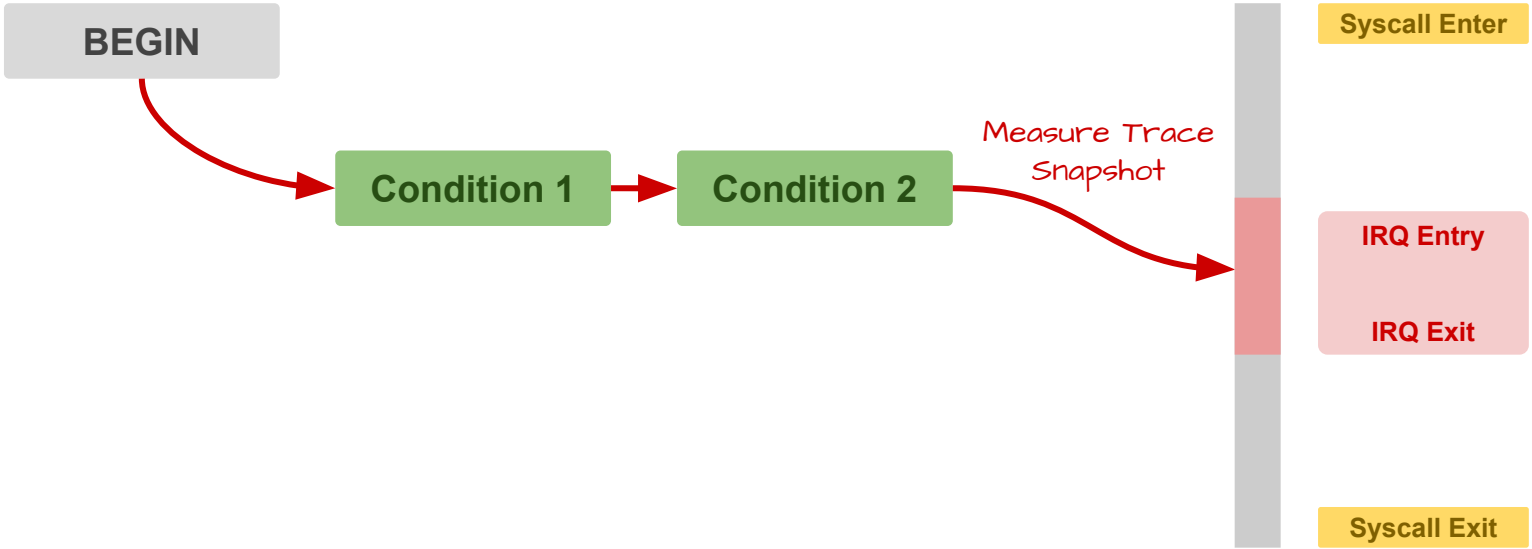
```
example-2535  [007] 1580764.896808: function:            __mnt_drop_write
example-2535  [007] 1580764.896808: function:            __sb_end_write
example-2535  [007] 1580764.896809: function:            fsnotify_parent
example-2535  [007] 1580764.896810: function:            fsnotify
example-2535  [007] 1580764.896810: function:            fpregs_assert_state_consistent
```

Syscall Exit

[1]https://github.com/iovisor/bpftrace/tree/master/tools

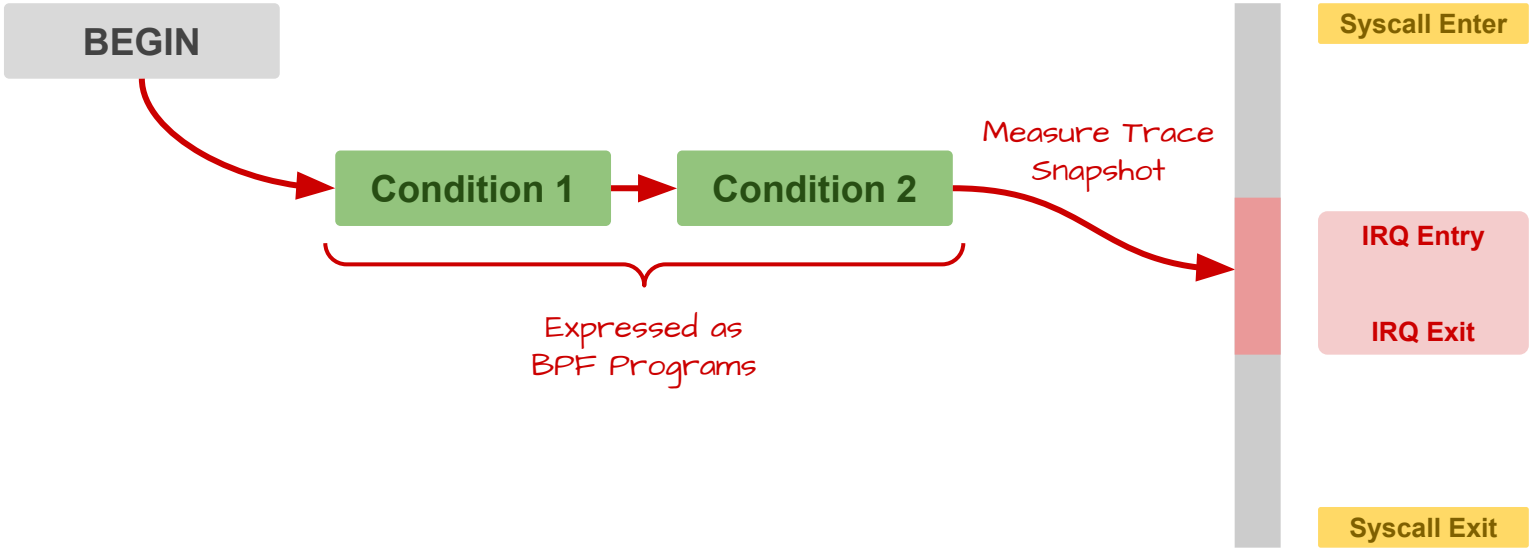
More Tracing Techniques

Live and Programmatic Analysis



More Tracing Techniques

Live and Programmatic Analysis



Tracing with BPF

bpftrace Tools

```
# ./biolatency.bt
Attaching 3 probes...
Tracing block device I/O... Hit Ctrl-C to end.
^C

@usecs:
[256, 512)             2   |                                                    |
[512, 1K)              10  |@                                                   |
[1K, 2K)               426 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@|
[2K, 4K)               230 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@                  |
[4K, 8K)               9   |@                                                   |
[8K, 16K)              128 |@@@@@@@@@@@@@@@@                                    |
[16K, 32K)             68  |@@@@@@@@                                            |
[32K, 64K)             0   |                                                    |
[64K, 128K)            0   |                                                    |
[128K, 256K)           10  |@                                                   |
```

Most block operations took between 1024 to 2048 micro-seconds

[1]https://github.com/iovisor/bpftrace/tree/master/tools

Tracing with BPF

bpftrace Tools

```
# ./biosnoop.bt
Attaching 4 probes...
TIME(ms)   COMM             PID   LAT(ms)
611        bash             4179       10
611        cksum            4179       0
627        cksum            4179       15
641        cksum            4179       13
644        cksum            4179       3
658        cksum            4179       13
673        cksum            4179       14
686        cksum            4179       13
```

[1]https://github.com/iovisor/bpftrace/tree/master/tools

Tracing with BPF

bpftrace Tools

biostack.bt
bitesize.bt
dcsnoop.bt
mdflush.bt

BCC Tools

biostat.py
bitesize.py
bitehist.py
biotop.py

[1]**https://github.com/iovisor/bpftrace/tree/master/tools**

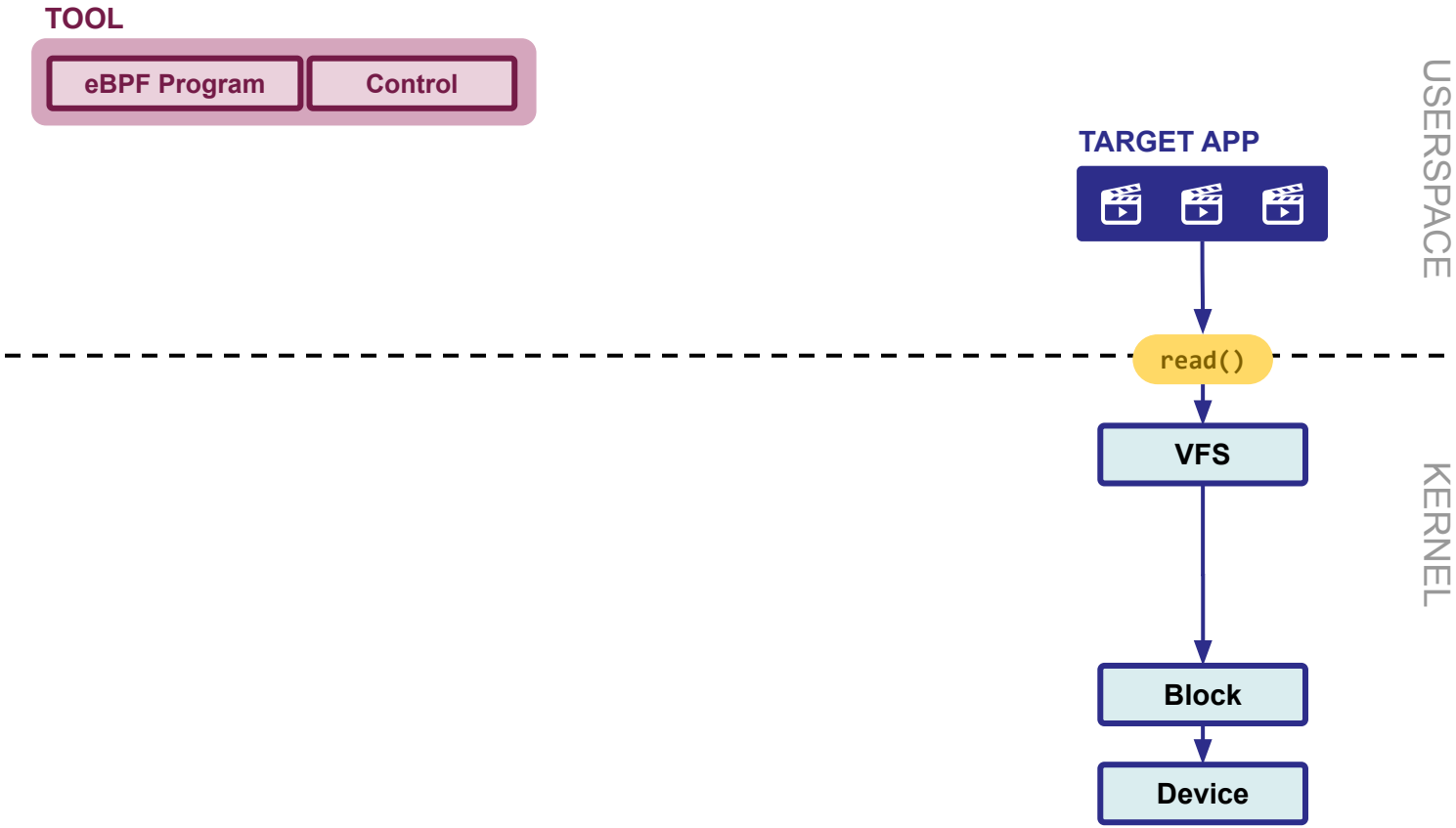
Building tools with

eBPF for Observability

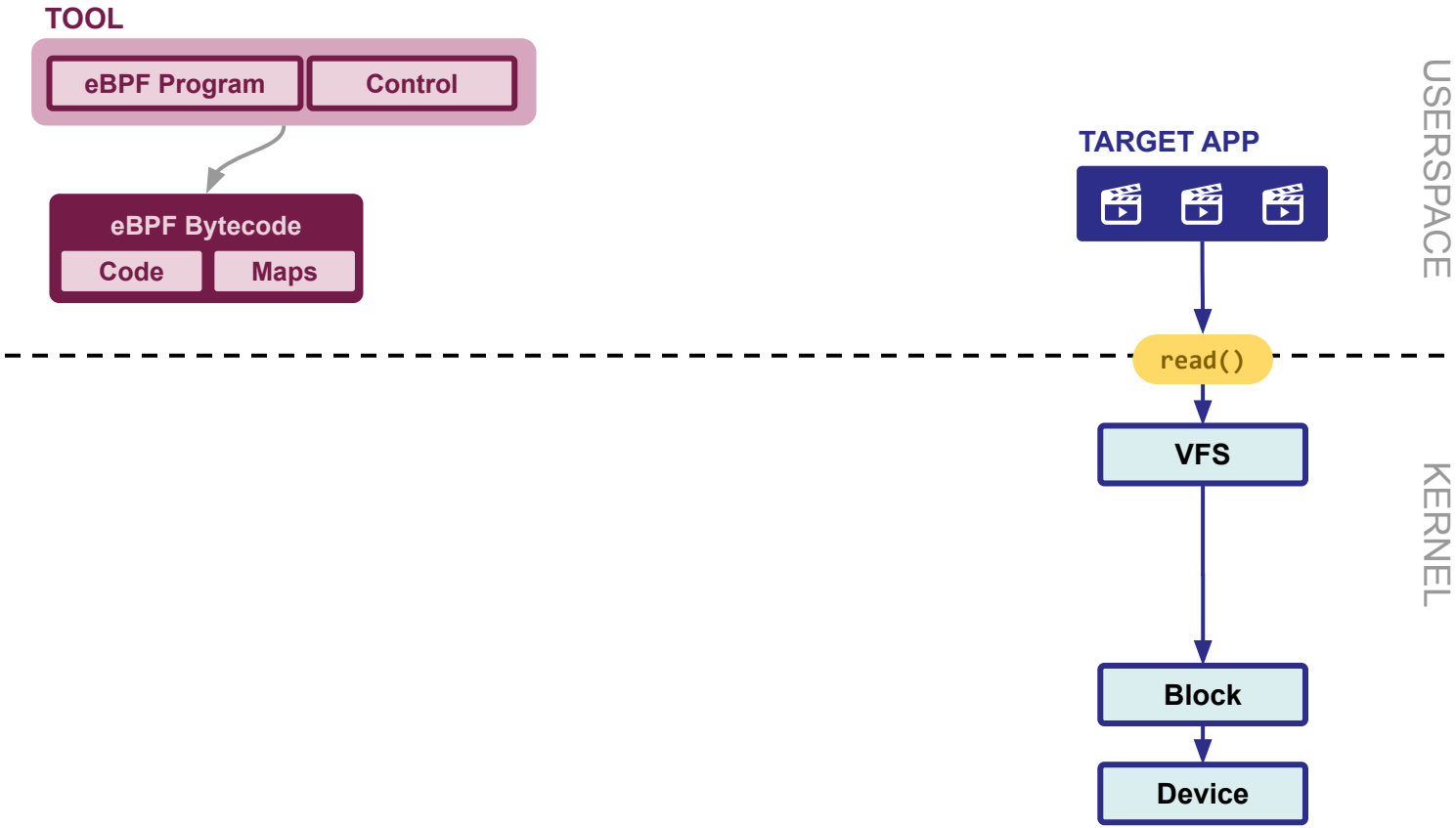
- **Probe the OS, Apps, Network**
  - Single programmatic way of tracing and monitoring each and every aspect of the software infrastructure
  - Hook to any userspace or kernel function* or predefined tracepoints, then extract and visualize data
  - Enable and disable probes, live filtering, low runtime overhead
- **Workflow**
  - Create your observation tool in userspace
  - Compile it and send it to kernel and attach to hooks
  - It runs and returns data back via buffer or maps

eBPF for Observability

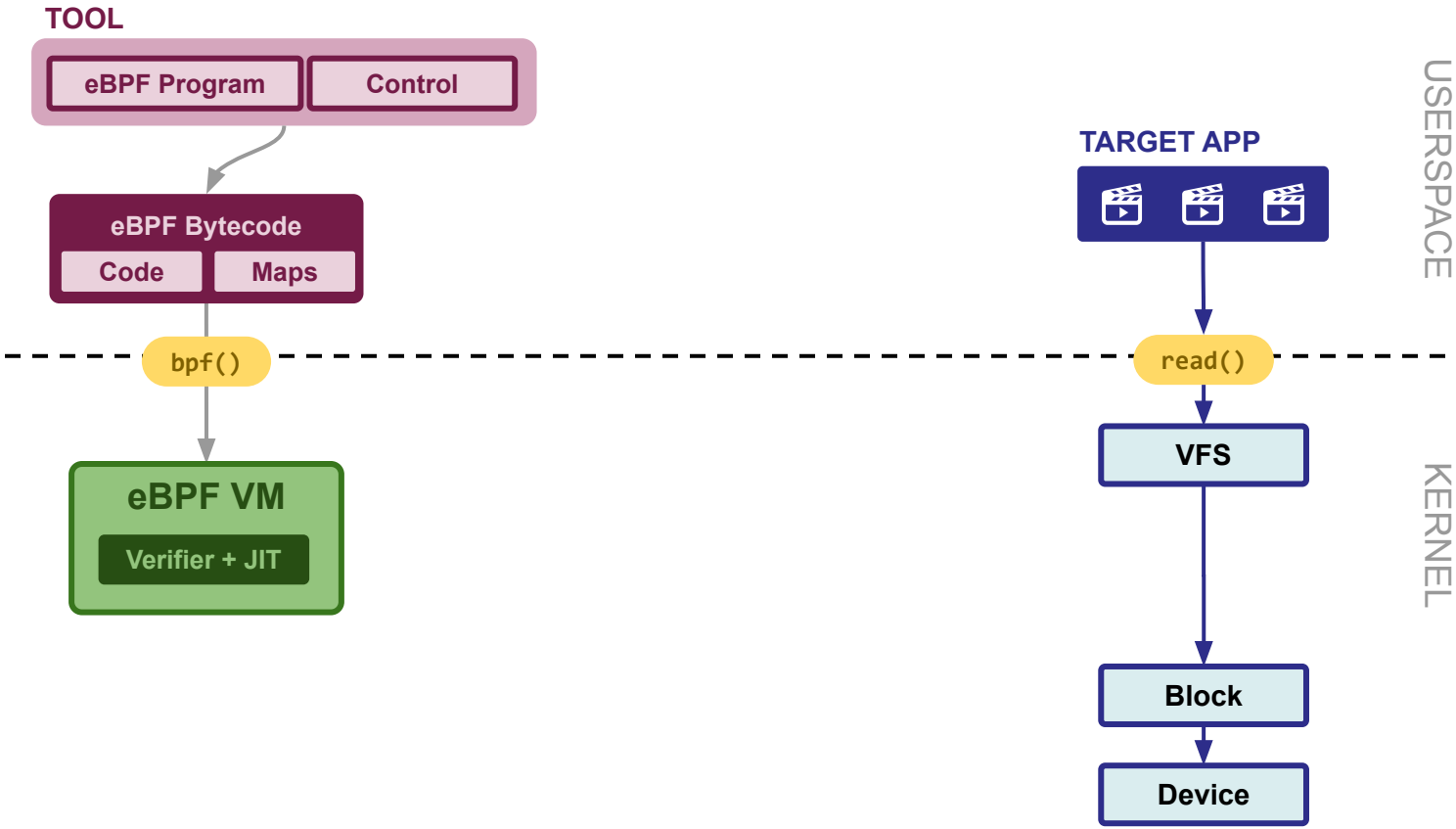
eBPF for Observability



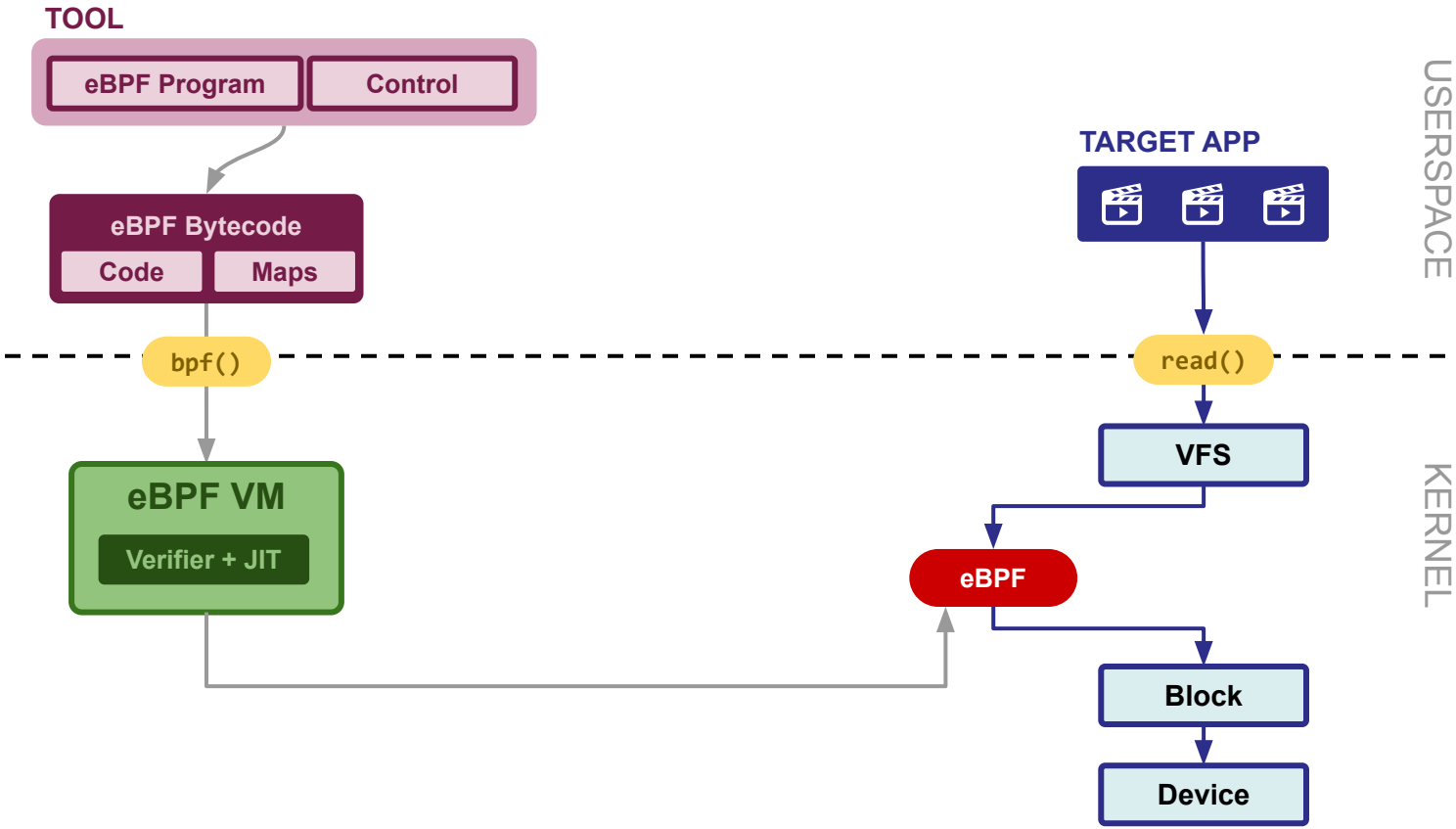
eBPF for Observability



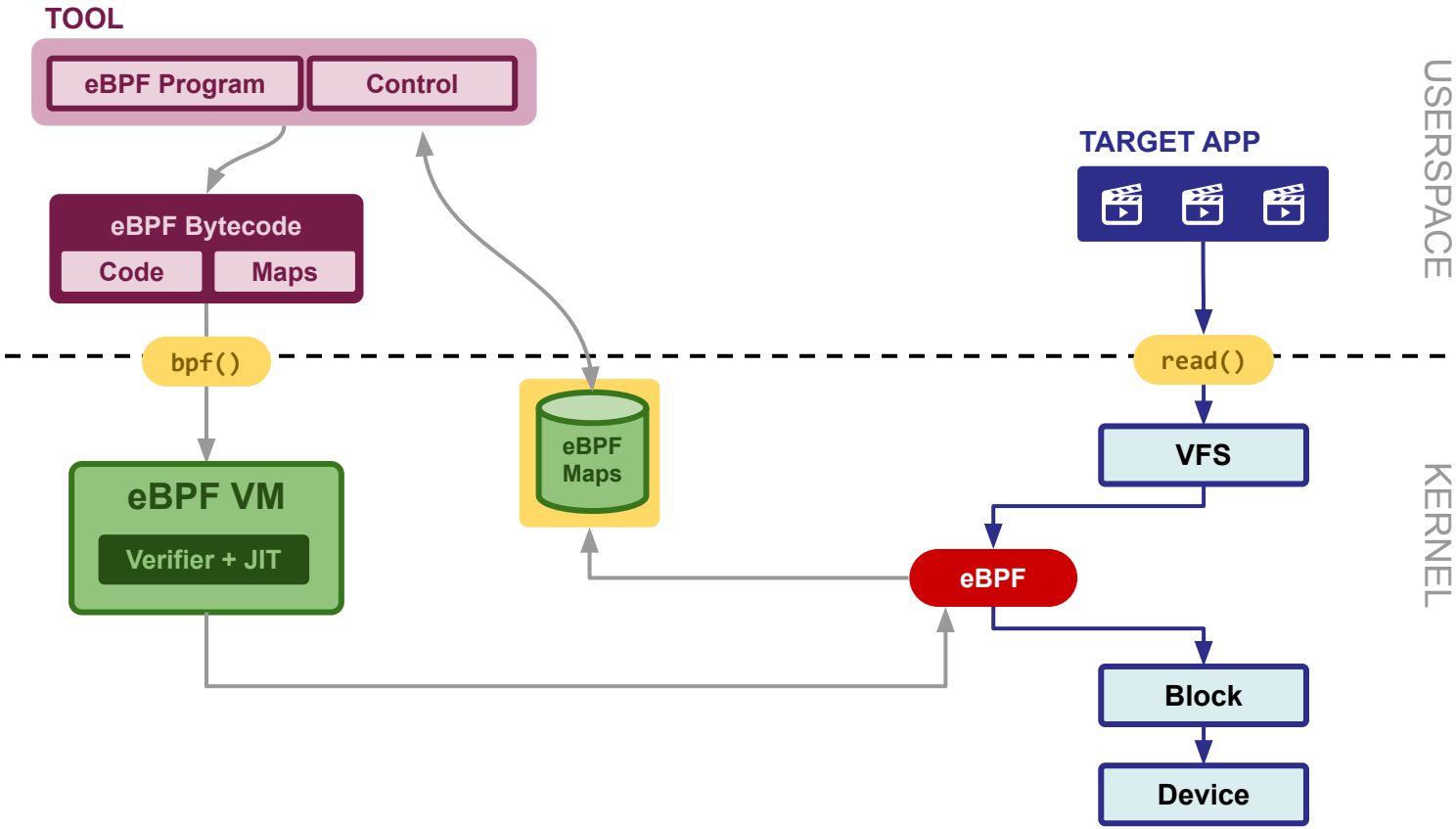
eBPF for Observability



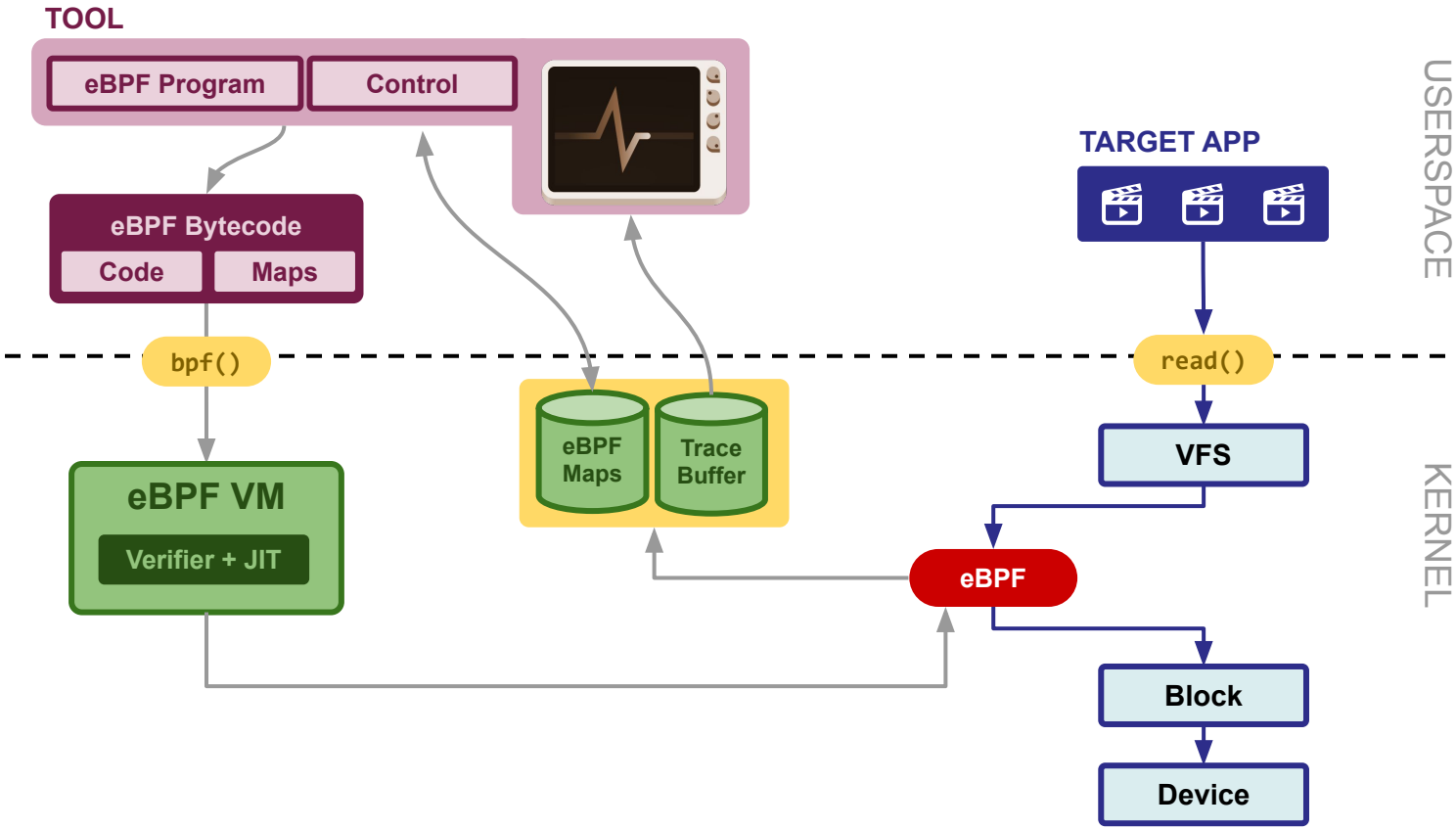
eBPF for Observability



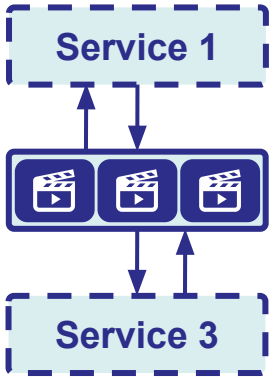
eBPF for Observability



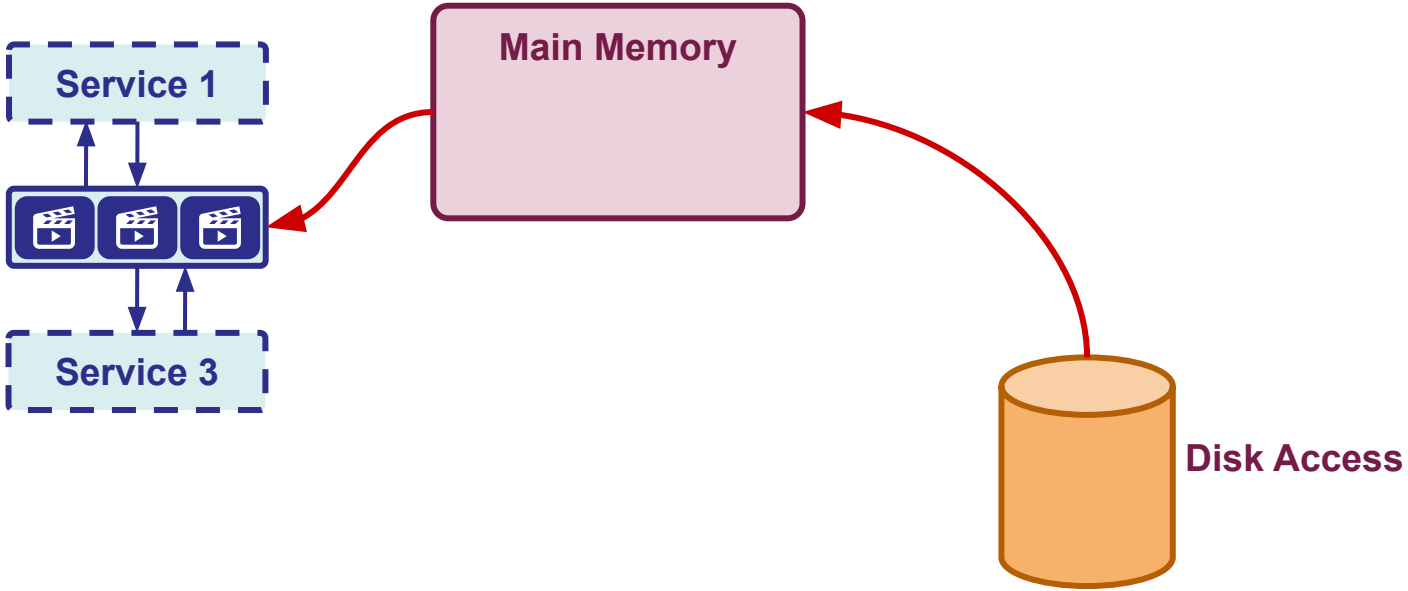
eBPF for Observability



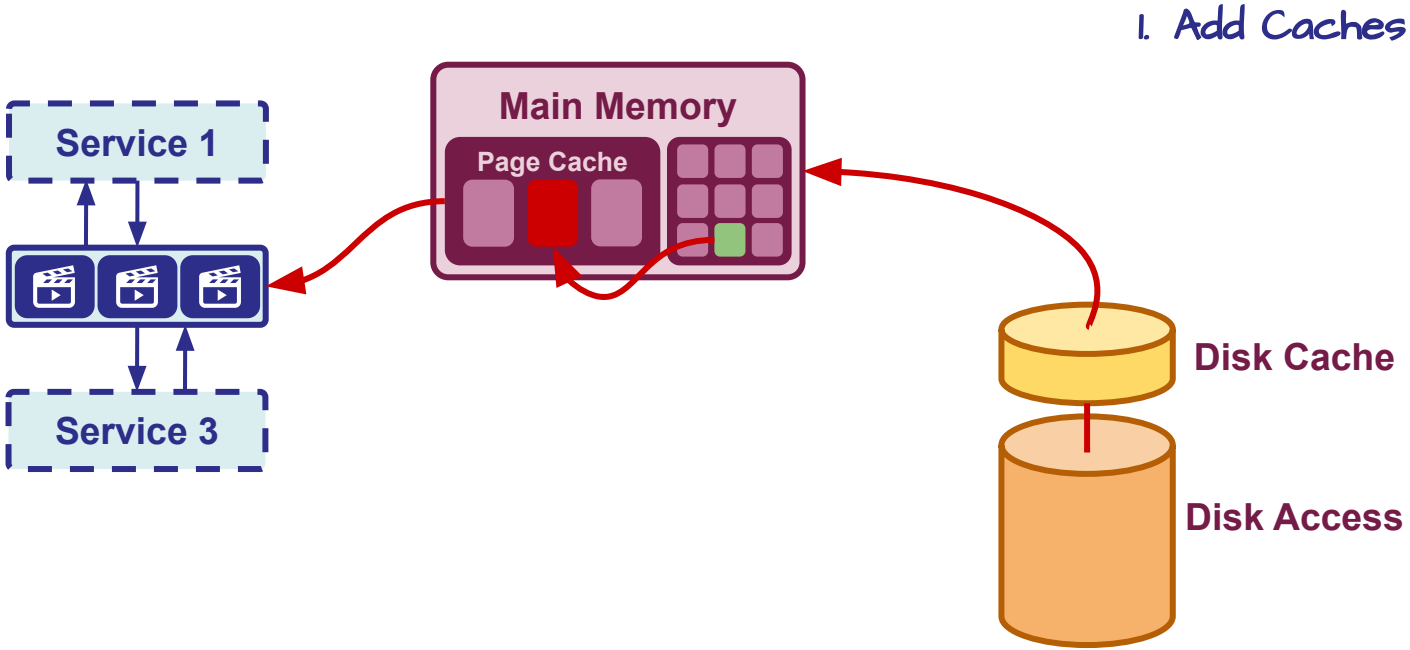
Case Study: Read-ahead



Case Study: Read-ahead

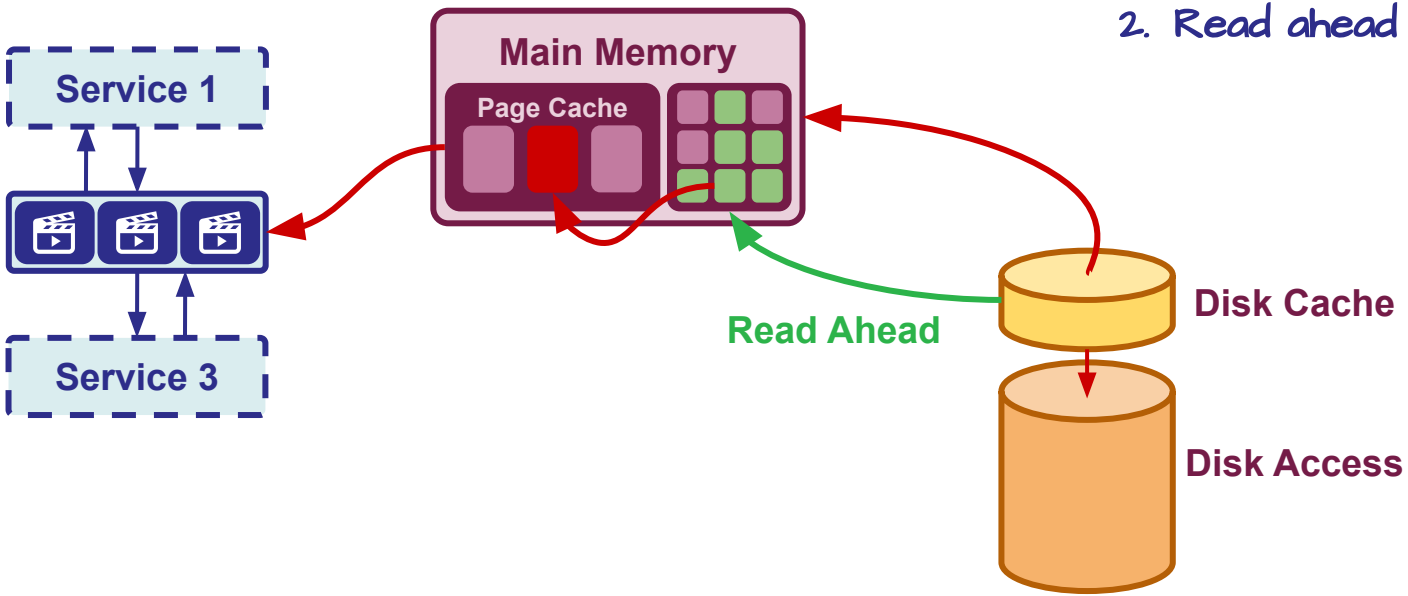


Case Study: Read-ahead

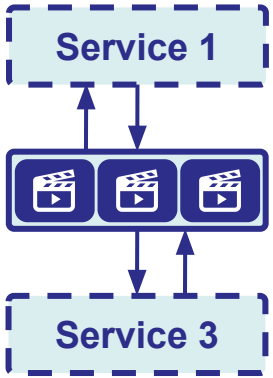


Case Study: Read-ahead

1. Add Caches
2. Read ahead



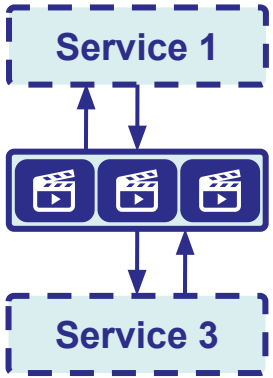
Case Study: Read-ahead



How?

- Apps can “advise” the OS to use efficient read-ahead mechanism for sequential reads (like the one here)
  - `madvise (addr, len, MADV_SEQUENTIAL)`
- OS enables an aggressive read-ahead of pages in memory since it now know faults will be less due to seq read.

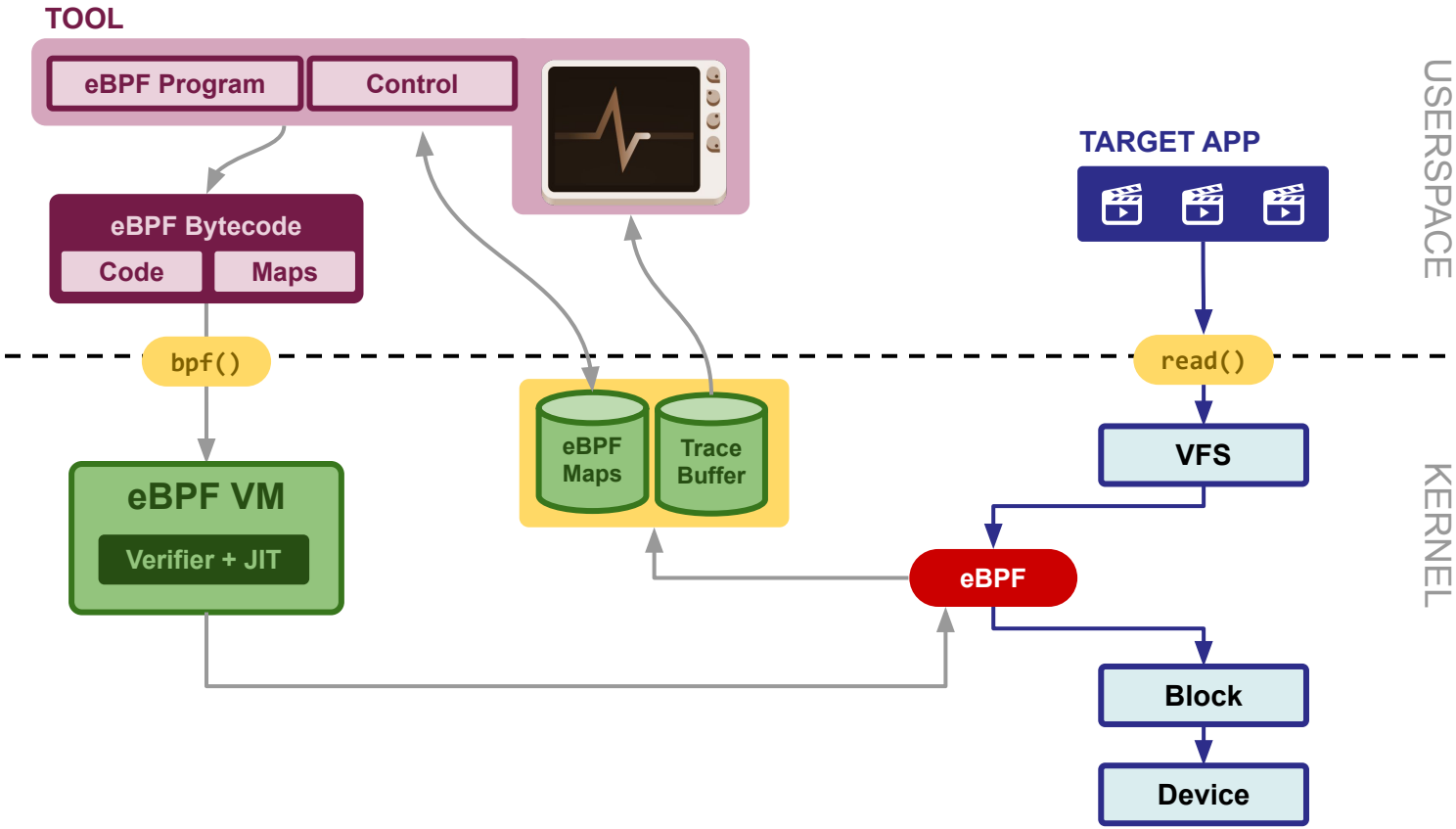
Case Study: Read-ahead



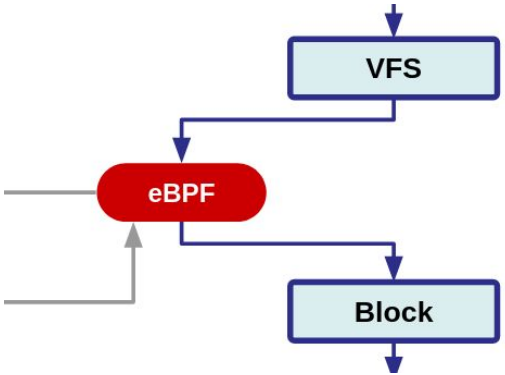
Problems

- Do apps really know how to use it?
- Modern apps are unaware of underlying infrastructure
- Lack of insight on how the different hardwares/OS optimize this really
- Has been a lack of **dynamic and deep-observability** tools to “see” the problems in real time

eBPF for Observability

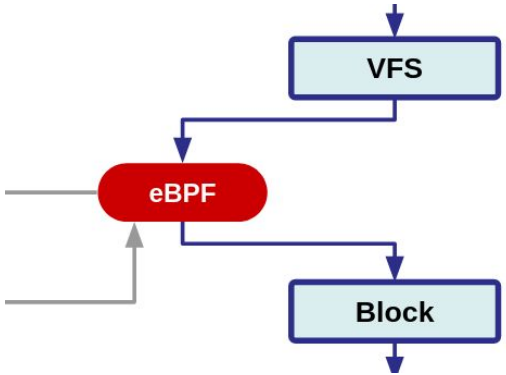


Building readaheadstat



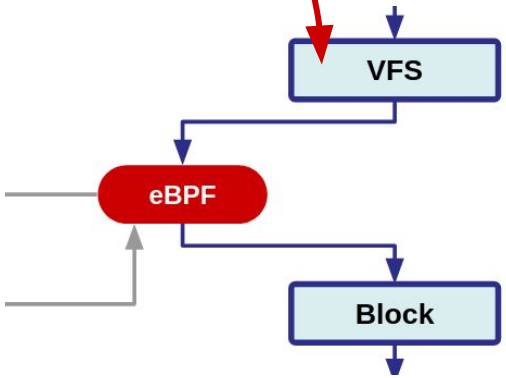
Building `readaheadstat`

- **Where and what are the probes?**
  - *Dynamic:*
    - **Kprobes/Kretprobes (Kernel)**
    - USDT/Uprobes (Userspace)
  - *Static:* Tracepoints



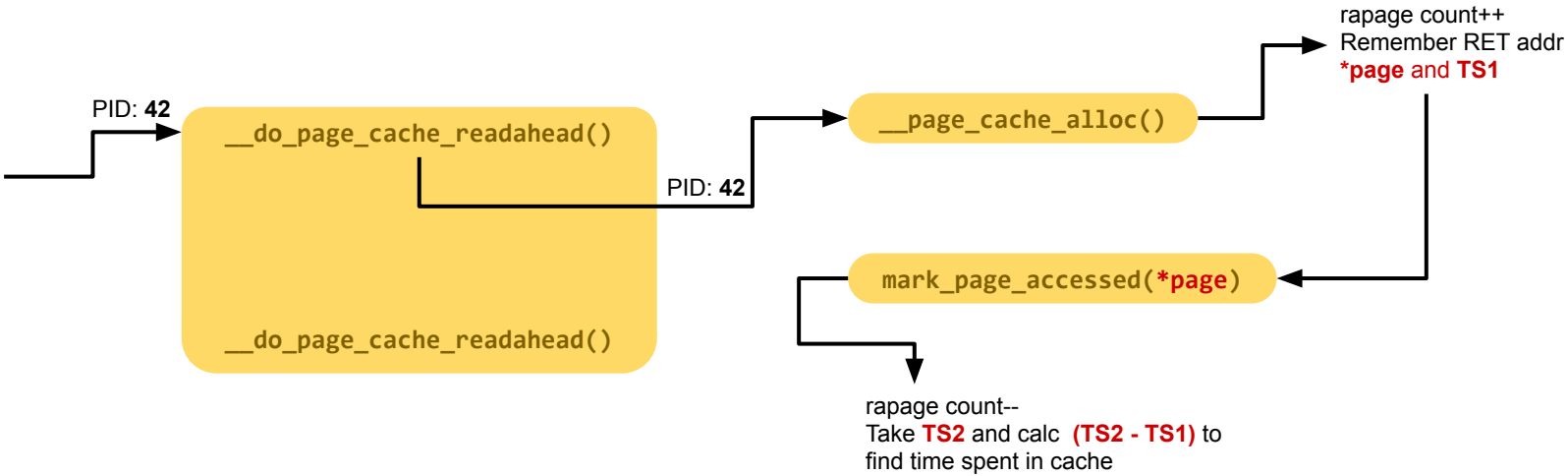
Building readaheadstat

- **Where and what are the probes?**
  - *Dynamic:*
    - **Kprobes/Kretprobes (Kernel)**
    - USDT/Uprobes (Userspace)
  - *Static:* Tracepoints



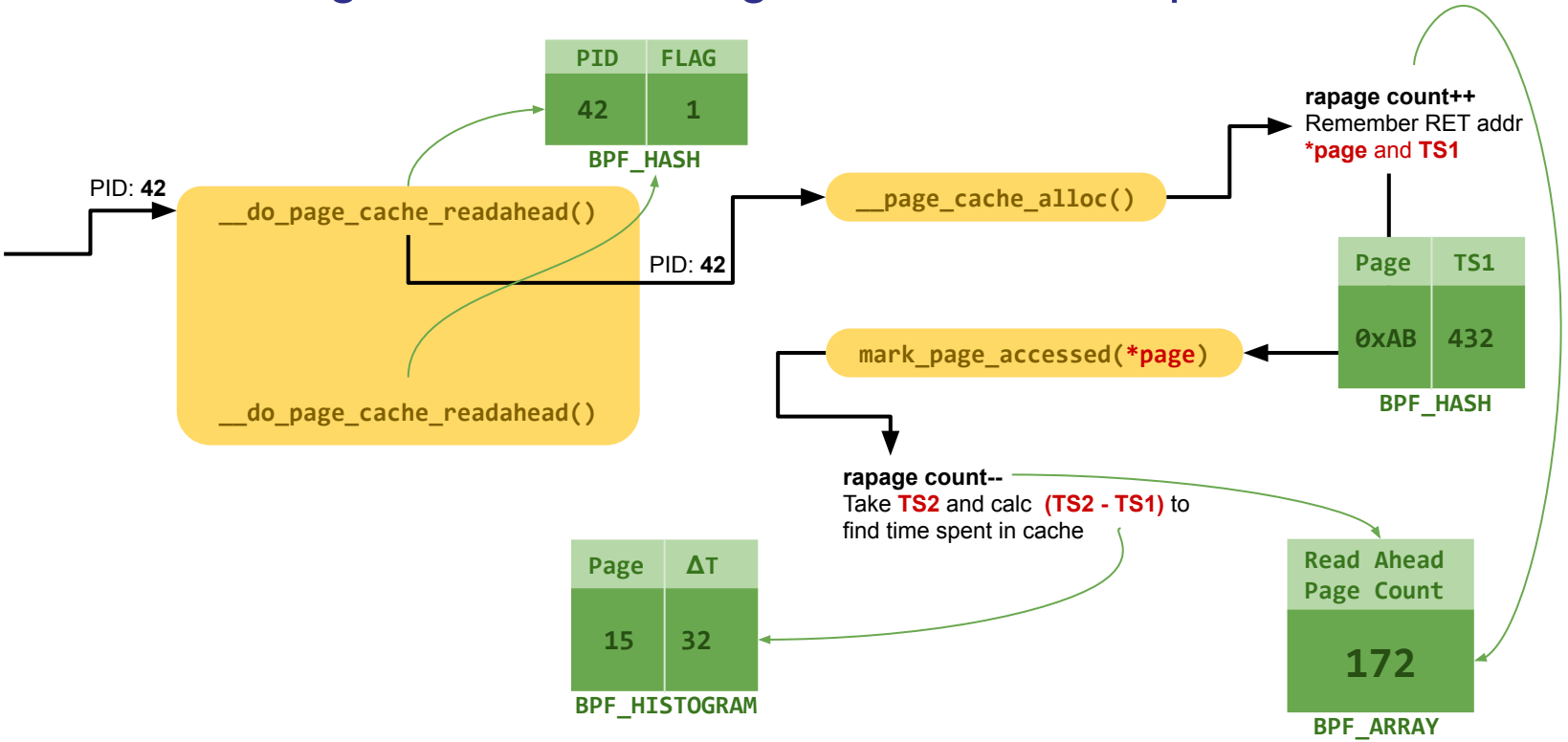
Building readaheadstat

- Selecting Kprobes/Kretprobes



Building readaheadstat

- Maintaining state and storing data in hash-maps

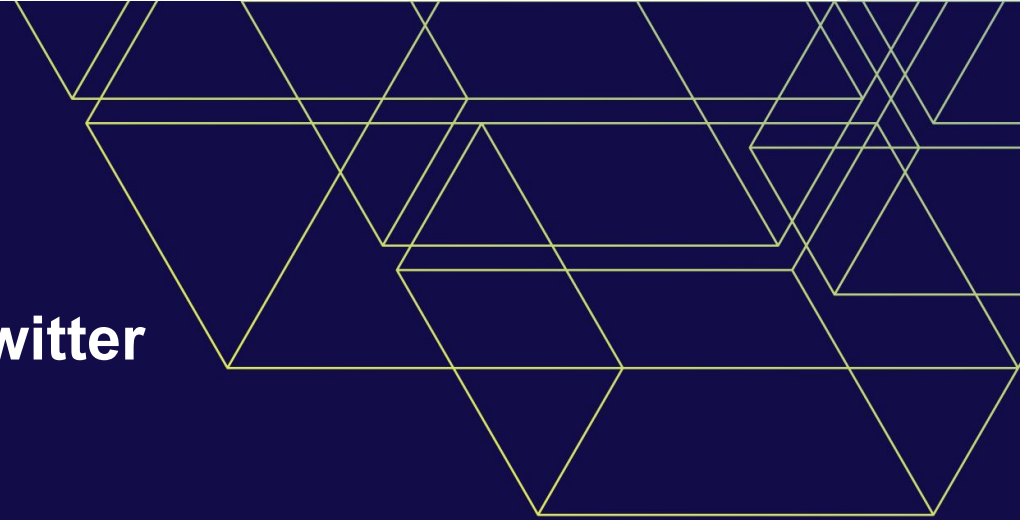


`readaheadstat` **Demo**

Download and Try

https://github.com/tuxology/readaheadstat

Building readaheadstat



You can reach out to us on Twitter

@tuxology

@hani_nemati

Or e-mail us

suchakra@shiftleft.io

hanemati@microsoft.com