# Fun with Dynamic Kernel Tracing Events

## The things you just shouldn't be able to do!

Steven Rostedt
10/23/2018

vmware

# I assume you are familiar with:

- ftrace
  - /sys/kernel/{debug/}tracing
  - current_tracer
  - trace file (output)
  - tracing_on (enabling tracing
  - Function tracer
  - Function graph tracer
  - Trace events
    - sched_switch, sched_waking, hrtimer, etc

# I assume you are familiar with:

- ftrace
  - /sys/kernel/{debug/}tracing
  - current_tracer
  - trace file (output)
  - tracing_on (enabling tracing
  - Function tracer
  - Function graph tracer
  - Trace events
    - sched_switch, sched_waking, hrtimer, etc
- If not, pretend you are

# Static events are boring

```
# cd /sys/kernel/tracing
# echo 1 > events/sched/enable
# cat trace
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
            bash-12649 [007] d..3 100238.467121: sched_waking: comm=kworker/u16:0 pid=12674 prio=120 target_cpu=005
            bash-12649 [007] d..4 100238.467130: sched_wake_idle_without_ipi: cpu=5
            bash-12649 [007] d..4 100238.467131: sched_wakeup: comm=kworker/u16:0 pid=12674 prio=120 target_cpu=005
          <idle>-0     [005] d..2 100238.467139: sched_switch: prev_comm=swapper/5 prev_pid=0 prev_prio=120 prev_state=S ==> next_comm
    kworker/u16:0-12674 [005] d..2 100238.467145: sched_waking: comm=sshd pid=12648 prio=120 target_cpu=002
    kworker/u16:0-12674 [005] d..3 100238.467152: sched_wake_idle_without_ipi: cpu=2
    kworker/u16:0-12674 [005] d..3 100238.467153: sched_wakeup: comm=sshd pid=12648 prio=120 target_cpu=002
    kworker/u16:0-12674 [005] d..2 100238.467155: sched_stat_runtime: comm=kworker/u16:0 pid=12674 runtime=22534 [ns] vruntime=24569732
    kworker/u16:0-12674 [005] d..2 100238.467158: sched_switch: prev_comm=kworker/u16:0 prev_pid=12674 prev_prio=120 prev_state=R+ ==> n
          <idle>-0     [002] d..2 100238.467160: sched_switch: prev_comm=swapper/2 prev_pid=0 prev_prio=120 prev_state=S ==> next_comm=
            sshd-12648 [002] d..2 100238.467251: sched_stat_runtime: comm=sshd pid=12648 runtime=96885 [ns] vruntime=251307031 [ns]
            sshd-12648 [002] d..2 100238.467257: sched_switch: prev_comm=sshd prev_pid=12648 prev_prio=120 prev_state=D ==> next_comm=s
            bash-12649 [007] d.h4 100238.481840: sched_waking: comm=kworker/7:2 pid=12613 prio=120 target_cpu=007
            bash-12649 [007] dNh5 100238.481845: sched_wakeup: comm=kworker/7:2 pid=12613 prio=120 target_cpu=007
            bash-12649 [007] dNh4 100238.481935: sched_waking: comm=systemd-journal pid=614 prio=120 target_cpu=004
            bash-12649 [007] dNh5 100238.481936: sched_wake_idle_without_ipi: cpu=4
            bash-12649 [007] dNh5 100238.481936: sched_wakeup: comm=systemd-journal pid=614 prio=120 target_cpu=004
            bash-12649 [007] dNh2 100238.481937: sched_stat_runtime: comm=bash pid=12649 runtime=15794405 [ns] vruntime=124251877 [ns]
            bash-12649 [007] dNh2 100238.481940: sched_stat_runtime: comm=bash pid=12649 runtime=3061 [ns] vruntime=124254938 [ns]
            bash-12649 [007] dNs3 100238.481942: sched_waking: comm=rcu_preempt pid=10 prio=120 target_cpu=006
            bash-12649 [007] dNs4 100238.481942: sched_wake_idle_without_ipi: cpu=6
```

# Static events are boring

- They are already defined for you
- You only see what the developer wants you to see
- They can’t be changed
- They’re just *static*



# What about Function Tracing??

```
# cd /sys/kernel/tracing
# echo '*spin_*' > set_ftrace_filter
# echo function > current_tracer
# cat trace
# tracer: function
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
          <idle>-0     [000] d..1 101232.558539: _raw_spin_lock <-get_next_timer_interrupt
          <idle>-0     [004] d..1 101232.558539: _raw_spin_lock <-get_next_timer_interrupt
          <idle>-0     [000] d..2 101232.558541: _raw_spin_unlock <-get_next_timer_interrupt
          <idle>-0     [004] d..2 101232.558541: _raw_spin_unlock <-get_next_timer_interrupt
          <idle>-0     [004] d..1 101232.558542: _raw_spin_lock_irqsave <-hrtimer_get_next_event
          <idle>-0     [000] d..1 101232.558542: _raw_spin_lock_irqsave <-hrtimer_get_next_event
          <idle>-0     [000] d..2 101232.558542: _raw_spin_unlock_irqrestore <-hrtimer_get_next_event
          <idle>-0     [004] d..2 101232.558542: _raw_spin_unlock_irqrestore <-hrtimer_get_next_event
          <idle>-0     [004] d..1 101232.558543: _raw_spin_lock_irqsave <-hrtimer_next_event_without
          <idle>-0     [000] d..1 101232.558543: _raw_spin_lock_irqsave <-hrtimer_next_event_without
          <idle>-0     [000] d..2 101232.558543: _raw_spin_unlock_irqrestore <-hrtimer_next_event_without
          <idle>-0     [004] d..2 101232.558544: _raw_spin_unlock_irqrestore <-hrtimer_next_event_without
            bash-12649 [007] .... 101232.558545: _raw_spin_lock <-ksys_dup3
            bash-12649 [007] ...1 101232.558546: _raw_spin_unlock <-do_dup2
            bash-12649 [007] .... 101232.558548: _raw_spin_lock_irq <-task_work_run
            bash-12649 [007] d..1 101232.558548: _raw_spin_unlock_irq <-task_work_run
            bash-12649 [007] .... 101232.558550: _raw_spin_lock_irq <-task_work_run
            bash-12649 [007] d..1 101232.558550: _raw_spin_unlock_irq <-task_work_run
            bash-12649 [007] .... 101232.558554: _raw_spin_lock <-__close_fd
```

# What about Function Tracing??

- You can pick which functions to trace
- All sorts of filtering of these functions
- You can pick functions just in a particular module:
  - `echo ':mod:ext3' > set_ftrace_filter`
- Is it still boring?



# What about Function Tracing??

- You can pick which functions to trace
- All sorts of filtering of these functions
- You can pick functions just in a particular module:
  - `echo ':mod:ext3' > set_ftrace_filter`
- Is it still boring?

  YES!

# What about Function Tracing??

- Only shows you the function (and parent function)
- No parameters
- No variables
- No structures
- Boring!

# What do you want?

vmware

# What do you want?

KPROBES!

# Kprobes

- Been around since 2004 (before git history)
  - Just the basic infrastructure
  - Needed more elaborate tools on top



# Kprobes

- Been around since 2004 (before git history)
  - Just the basic infrastructure
  - Needed more elaborate tools on top
- kprobe events ( for tracing )
  - Introduced in 2009 (by Masami Hiramatsu)
  - Allows to create dynamic events
  - Can access parameters
  - Can access variables
  - They then act just like any other trace event



# Kprobes are great!

- They been around forever, why isn’t anyone using them?

# Kprobes are great!

- They been around forever, why isn’t anyone using them?

  They’re complicated



# Using kprobes

- From Linux kernel source: Documentation/trace/kprobetrace.rst

```
Synopsis of kprobe_events
-------------------------
::

  p[:[GRP/]EVENT] [MOD:]SYM[+offs]|MEMADDR [FETCHARGS]  : Set a probe
  r[MAXACTIVE][:[GRP/]EVENT] [MOD:]SYM[+0] [FETCHARGS]  : Set a return probe
  -:[GRP/]EVENT                                         : Clear a probe

 GRP            : Group name. If omitted, use "kprobes" for it.
 EVENT          : Event name. If omitted, the event name is generated
                  based on SYM+offs or MEMADDR.
 MOD            : Module name which has given SYM.
 SYM[+offs]     : Symbol+offset where the probe is inserted.
 MEMADDR        : Address where the probe is inserted.
 MAXACTIVE      : Maximum number of instances of the specified function that
                  can be probed simultaneously, or 0 for the default value
                  as defined in Documentation/kprobes.txt section 1.3.1.

 FETCHARGS      : Arguments. Each probe can have up to 128 args.
  %REG          : Fetch register REG
  @ADDR         : Fetch memory at ADDR (ADDR should be in kernel)
  @SYM[+|-offs] : Fetch memory at SYM +|- offs (SYM should be a data symbol)
  $stackN       : Fetch Nth entry of stack (N >= 0)
  $stack        : Fetch stack address.
  $retval       : Fetch return value.(*)
  $comm         : Fetch current task comm.
  +|-offs(FETCHARG) : Fetch memory at FETCHARG +|- offs address.(**)
  NAME=FETCHARG : Set NAME as the argument name of FETCHARG.
  FETCHARG:TYPE : Set TYPE as the type of FETCHARG. Currently, basic types
                  (u8/u16/u32/u64/s8/s16/s32/s64), hexadecimal types
                  (x8/x16/x32/x64), "string" and bitfield are supported.

  (*) only for return probe.
  (**) this is useful for fetching a field of data structures.
```

# Using kprobes

- Let’s say you want to see what files are being opened
- Look in the Linux source tree for sys_open()

```
SYSCALL_DEFINE3(open, const char __user *, filename, int, flags, umode_t, mode)
{
        if (force_o_largefile())
                flags |= O_LARGEFILE;

        return do_sys_open(AT_FDCWD, filename, flags, mode);
}
```

# Using kprobes

- Let’s say you want to see what files are being opened
- Look in the Linux source tree for sys_open()

```
SYSCALL_DEFINE3(open, const char __user *, filename, int, flags, umode_t, mode)
{
        if (force_o_largefile())
                flags |= O_LARGEFILE;

        return do_sys_open(AT_FDCWD, filename, flags, mode);
}
```

# From arch/x86/entry/calling.h

```
x86 function call convention, 64-bit:
 --------------------------------------
  arguments           |  callee-saved      | extra caller-saved | return
 [callee-clobbered]   |                    | [callee-clobbered] |
 ---------------------------------------------------------------------------
 rdi rsi rdx rcx r8-9 | rbx rbp [*] r12-15 | r10-11             | rax, rdx [**]
```

# From arch/x86/entry/calling.h

```
x86 function call convention, 64-bit:
 -------------------------------------
  arguments           |  callee-saved      | extra caller-saved | return
 [callee-clobbered]   |                    | [callee-clobbered] |
 ---------------------------------------------------------------------------
 rdi rsi rdx rcx r8-9 | rbx rbp [*] r12-15 | r10-11             | rax, rdx [**]
```

# Using kprobes

- You know what register to get (for the second argument)
- But how do you get it?

# Using kprobes

- You know what register to get (for the second argument)
- But how do you get it?

  **arch/x86/include/asm/ptrace.h**

# Using kprobes

- You know what register to get (for the second argument)
- But how do you get it?

**arch/x86/include/asm/ptrace.h**

```
struct pt_regs {
        unsigned long bx;
        unsigned long cx;
        unsigned long dx;
        unsigned long si;
        unsigned long di;
        unsigned long bp;
        unsigned long ax;
        unsigned short ds;
        unsigned short __dsh;
        unsigned short es;
        unsigned short __esh;
        unsigned short fs;
        unsigned short __fsh;
        unsigned short gs;
        unsigned short __gsh;
        unsigned long orig_ax;
        unsigned long ip;
        unsigned short cs;
        unsigned short __csh;
        unsigned long flags;
        unsigned long sp;
        unsigned short ss;
        unsigned short __ssh;
};
```

# Using kprobes

- You know what register to get (for the second argument)
- But how do you get it?

**arch/x86/include/asm/ptrace.h**

```
struct pt_regs {
        unsigned long bx;
        unsigned long cx;
        unsigned long dx;
        unsigned long si;
        unsigned long di;
        unsigned long bp;
        unsigned long ax;
        unsigned short ds;
        unsigned short __dsh;
        unsigned short es;
        unsigned short __esh;
        unsigned short fs;
        unsigned short __fsh;
        unsigned short gs;
        unsigned short __gsh;
        unsigned long orig_ax;
        unsigned long ip;
        unsigned short cs;
        unsigned short __csh;
        unsigned long flags;
        unsigned long sp;
        unsigned short ss;
        unsigned short __ssh;
};
```

# Using kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:open do_sys_open file=%si' > kprobe_events
# echo 1 > events/kprobes/open/enable
# ls > /dev/null
# cat trace
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
              ls-13261 [005] ...1 104673.170507: open: (do_sys_open+0x0/0x250) file=0x562bbf8fe790
              ls-13261 [005] ...1 104673.171421: open: (do_sys_open+0x0/0x250) file=0x7f7d2b7c42ae
              ls-13261 [005] ...1 104673.171465: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c7640
              ls-13261 [005] ...1 104673.171602: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c7b10
              ls-13261 [005] ...1 104673.171690: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c7fe0
              ls-13261 [005] ...1 104673.171803: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c84b0
              ls-13261 [005] ...1 104673.171889: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c8980
              ls-13261 [005] ...1 104673.171974: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c8ef8
              ls-13261 [005] ...1 104673.172728: open: (do_sys_open+0x0/0x250) file=0x7f7d2b599b07
              ls-13261 [005] ...1 104673.172850: open: (do_sys_open+0x0/0x250) file=0x7f7d2b142670
              ls-13261 [005] ...1 104673.172943: open: (do_sys_open+0x0/0x250) file=0x55dfbcbcbc80
            <...>-13262 [001] ...1 104674.626324: open: (do_sys_open+0x0/0x250) file=0x7efeb7fcf2ae
            <...>-13262 [001] ...1 104674.626365: open: (do_sys_open+0x0/0x250) file=0x7efeb81d2640
            <...>-13262 [001] ...1 104674.626798: open: (do_sys_open+0x0/0x250) file=0x7efeb7d79670
            <...>-13262 [001] ...1 104674.626869: open: (do_sys_open+0x0/0x250) file=0x7fff02d3765e
```

# Using kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:open do_sys_open file=%si' > kprobe_events
# echo 1 > events/kprobes/open/enable
# ls > /dev/null
# cat trace
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  |||  TIMESTAMP  FUNCTION
#              | |       |   ||||     |         |
              ls-13261 [005] ...1 104673.170507: open: (do_sys_open+0x0/0x250) file=0x562bbf8fe790
              ls-13261 [005] ...1 104673.171421: open: (do_sys_open+0x0/0x250) file=0x7f7d2b7c42ae
              ls-13261 [005] ...1 104673.171465: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c7640
              ls-13261 [005] ...1 104673.171602: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c7b10
              ls-13261 [005] ...1 104673.171690: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c7fe0
              ls-13261 [005] ...1 104673.171803: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c84b0
              ls-13261 [005] ...1 104673.171889: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c8980
              ls-13261 [005] ...1 104673.171974: open: (do_sys_open+0x0/0x250) file=0x7f7d2b9c8ef8
              ls-13261 [005] ...1 104673.172728: open: (do_sys_open+0x0/0x250) file=0x7f7d2b599b07
              ls-13261 [005] ...1 104673.172850: open: (do_sys_open+0x0/0x250) file=0x7f7d2b142670
              ls-13261 [005] ...1 104673.172943: open: (do_sys_open+0x0/0x250) file=0x55dfbcbcbc80
            <...>-13262 [001] ...1 104674.626324: open: (do_sys_open+0x0/0x250) file=0x7efeb7fcf2ae
            <...>-13262 [001] ...1 104674.626365: open: (do_sys_open+0x0/0x250) file=0x7efeb81d2640
            <...>-13262 [001] ...1 104674.626798: open: (do_sys_open+0x0/0x250) file=0x7efeb7d79670
            <...>-13262 [001] ...1 104674.626869: open: (do_sys_open+0x0/0x250) file=0x7fff02d3765e
```

BORING!!!!

# Using kprobes

```
file=0x562bbf8fe790
file=0x7f7d2b7c42ae
file=0x7f7d2b9c7640
file=0x7f7d2b9c7b10
file=0x7f7d2b9c7fe0
file=0x7f7d2b9c84b0
file=0x7f7d2b9c8980
file=0x7f7d2b9c8ef8
file=0x7f7d2b599b07
file=0x7f7d2b142670
file=0x55dfbcbcbc80
file=0x7efeb7fcf2ae
file=0x7efeb81d2640
file=0x7efeb7d79670
file=0x7fff02d3765e
```

# Using kprobes

- From Linux kernel source: Documentation/trace/kprobetrace.rst

```
Synopsis of kprobe_events
-------------------------
::

  p[:[GRP/]EVENT] [MOD:]SYM[+offs]|MEMADDR [FETCHARGS]  : Set a probe
  r[MAXACTIVE][:[GRP/]EVENT] [MOD:]SYM[+0] [FETCHARGS]  : Set a return probe
  -:[GRP/]EVENT                                         : Clear a probe

 GRP            : Group name. If omitted, use "kprobes" for it.
 EVENT          : Event name. If omitted, the event name is generated
                  based on SYM+offs or MEMADDR.
 MOD            : Module name which has given SYM.
 SYM[+offs]     : Symbol+offset where the probe is inserted.
 MEMADDR        : Address where the probe is inserted.
 MAXACTIVE      : Maximum number of instances of the specified function that
                  can be probed simultaneously, or 0 for the default value
                  as defined in Documentation/kprobes.txt section 1.3.1.

 FETCHARGS      : Arguments. Each probe can have up to 128 args.
  %REG          : Fetch register REG
  @ADDR         : Fetch memory at ADDR (ADDR should be in kernel)
  @SYM[+|-offs] : Fetch memory at SYM +|- offs (SYM should be a data symbol)
  $stackN       : Fetch Nth entry of stack (N >= 0)
  $stack        : Fetch stack address.
  $retval       : Fetch return value.(*)
  $comm         : Fetch current task comm.
  +|-offs(FETCHARG) : Fetch memory at FETCHARG +|- offs address.(**)
  NAME=FETCHARG : Set NAME as the argument name of FETCHARG.
  FETCHARG:TYPE : Set TYPE as the type of FETCHARG. Currently, basic types
                  (u8/u16/u32/u64/s8/s16/s32/s64), hexadecimal types
                  (x8/x16/x32/x64), "string" and bitfield are supported.

  (*) only for return probe.
  (**) this is useful for fetching a field of data structures.
```

# Using kprobes

```
echo ‘p:open do_sys_open file=%si:string’ > kprobe_events
```



# Using kprobes

???

```
echo ‘p:open do_sys_open file=%si:string’ > kprobe_events
```

bash: echo: write error: Invalid argument

# Using kprobes

???

```
echo ‘p:open do_sys_open file=%si:string’ > kprobe_events
```

bash: echo: write error: Invalid argument

trace_probe: string only accepts memory or address.
trace_kprobe: Parse error at argument[0]. (-22)

# Using kprobes

+0(%reg):string  - Makes %reg into an address that string can use

```
echo ‘p:open do_sys_open file=+0(%si):string’ > kprobe_events
```



# Using kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:open do_sys_open file=+0(%si):string' > kprobe_events
# echo 1 > events/kprobes/open/enable
# ls > /dev/null
# cat trace
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
              ls-13379 [006] ...1 105634.300773: open: (do_sys_open+0x0/0x250) file="/dev/null"
              ls-13379 [006] ...1 105634.301671: open: (do_sys_open+0x0/0x250) file="/etc/ld.so.cache"
              ls-13379 [006] ...1 105634.301714: open: (do_sys_open+0x0/0x250) file="/lib64/libselinux.so.1"
              ls-13379 [006] ...1 105634.301835: open: (do_sys_open+0x0/0x250) file="/lib64/libcap.so.2"
              ls-13379 [006] ...1 105634.301921: open: (do_sys_open+0x0/0x250) file="/lib64/libc.so.6"
              ls-13379 [006] ...1 105634.302033: open: (do_sys_open+0x0/0x250) file="/lib64/libpcre.so.1"
              ls-13379 [006] ...1 105634.302118: open: (do_sys_open+0x0/0x250) file="/lib64/libdl.so.2"
              ls-13379 [006] ...1 105634.302203: open: (do_sys_open+0x0/0x250) file="/lib64/libpthread.so.0"
              ls-13379 [006] ...1 105634.302951: open: (do_sys_open+0x0/0x250) file="/proc/filesystems"
              ls-13379 [006] ...1 105634.303072: open: (do_sys_open+0x0/0x250) file="/usr/lib/locale/locale-archive"
              ls-13379 [006] ...1 105634.303162: open: (do_sys_open+0x0/0x250) file="."
            <...>-13380 [006] ...1 105636.017950: open: (do_sys_open+0x0/0x250) file="/etc/ld.so.cache"
            <...>-13380 [006] ...1 105636.017991: open: (do_sys_open+0x0/0x250) file="/lib64/libc.so.6"
            <...>-13380 [006] ...1 105636.018391: open: (do_sys_open+0x0/0x250) file="/usr/lib/locale/locale-archive"
            <...>-13380 [006] ...1 105636.018470: open: (do_sys_open+0x0/0x250) file="trace"
```

# Using kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:open do_sys_open file=+0(%si):string' > kprobe_events
# echo 1 > events/kprobes/open/enable
# ls > /dev/null
# cat trace
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||   TIMESTAMP  FUNCTION
#              | |       |   ||||      |         |
              ls-13379 [006] ...1 105634.300773: open: (do_sys_open+0x0/0x250) file="/dev/null"
              ls-13379 [006] ...1 105634.301671: open: (do_sys_open+0x0/0x250) file="/etc/ld.so.cache"
              ls-13379 [006] ...1 105634.301714: open: (do_sys_open+0x0/0x250) file="/lib64/libselinux.so.1"
              ls-13379 [006] ...1 105634.301835: open: (do_sys_open+0x0/0x250) file="/lib64/libcap.so.2"
              ls-13379 [006] ...1 105634.301921: open: (do_sys_open+0x0/0x250) file="/lib64/libc.so.6"
              ls-13379 [006] ...1 105634.302033: open: (do_sys_open+0x0/0x250) file="/lib64/libpcre.so.1"
              ls-13379 [006] ...1 105634.302118: open: (do_sys_open+0x0/0x250) file="/lib64/libdl.so.2"
              ls-13379 [006] ...1 105634.302203: open: (do_sys_open+0x0/0x250) file="/lib64/libpthread.so.0"
              ls-13379 [006] ...1 105634.302951: open: (do_sys_open+0x0/0x250) file="/proc/filesystems"
              ls-13379 [006] ...1 105634.303072: open: (do_sys_open+0x0/0x250) file="/usr/lib/locale/locale-archive"
              ls-13379 [006] ...1 105634.303162: open: (do_sys_open+0x0/0x250) file="."
            <...>-13380 [006] ...1 105636.017950: open: (do_sys_open+0x0/0x250) file="/etc/ld.so.cache"
            <...>-13380 [006] ...1 105636.017991: open: (do_sys_open+0x0/0x250) file="/lib64/libc.so.6"
            <...>-13380 [006] ...1 105636.018391: open: (do_sys_open+0x0/0x250) file="/usr/lib/locale/locale-archive"
            <...>-13380 [006] ...1 105636.018470: open: (do_sys_open+0x0/0x250) file="trace"
```

EXCITING!!!!

# Using kprobes

- But it is still complex
- What to do about it?

```
echo ‘p:open do_sys_open file=+0(%si):string’ > kprobe_events
```

# function based events!



# function based events!

- Created in January 2018

```
echo ‘do_sys_open(NULL, string file)’ > function_events
```

# function based events!

```
# cd /sys/kernel/tracing/events
# echo 'do_sys_open(NULL, string file)' > function_events
# echo 1 > events/functions/do_sys_open/enable
# ls > /dev/null
# cat trace
# tracer: nop
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
              ls-822   [002] ...2   534.996438: do_syscall_64->do_sys_open(file=/dev/null)
              ls-822   [002] ...2   535.002703: do_syscall_64->do_sys_open(file=/etc/ld.so.cache)
              ls-822   [002] ...2   535.003098: do_syscall_64->do_sys_open(file=/lib64/libselinux.so.1)
              ls-822   [002] ...2   535.004079: do_syscall_64->do_sys_open(file=/lib64/libcap.so.2)
              ls-822   [002] ...2   535.004823: do_syscall_64->do_sys_open(file=/lib64/libc.so.6)
              ls-822   [002] ...2   535.006689: do_syscall_64->do_sys_open(file=/lib64/libpcre.so.1)
              ls-822   [002] ...2   535.007348: do_syscall_64->do_sys_open(file=/lib64/libdl.so.2)
              ls-822   [002] ...2   535.007882: do_syscall_64->do_sys_open(file=/lib64/libpthread.so.0)
              ls-822   [002] ...2   535.012683: do_syscall_64->do_sys_open(file=/usr/lib/locale/locale-archive)
              ls-822   [002] ...2   535.012847: do_syscall_64->do_sys_open(file=/usr/share/locale/locale.alias)
              ls-822   [002] ...2   535.013179: do_syscall_64->do_sys_open(file=/usr/lib/locale/en_US.utf8/LC_IDENTIFICA
              ls-822   [002] ...2   535.013384: do_syscall_64->do_sys_open(file=/usr/lib64/gconv/gconv-modules.cache)
              ls-822   [002] ...2   535.013637: do_syscall_64->do_sys_open(file=/usr/lib/locale/en_US.utf8/LC_MEASUREMEN
              ls-822   [002] ...2   535.013834: do_syscall_64->do_sys_open(file=/usr/lib/locale/en_US.utf8/LC_TELEPHONE)
```

# function based events!

```
# cd /sys/kernel/tracing/events
# echo 'do_sys_open(NULL, string file)' > function_events
# echo 1 > events/functions/do_sys_open/enable
# ls > /dev/null
# cat trace
# tracer: nop
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||   TIMESTAMP  FUNCTION
#              | |       |   ||||      |         |
              ls-822   [002] ...2   534.996438: do_syscall_64->do_sys_open(file=/dev/null)
              ls-822   [002] ...2   535.002703: do_syscall_64->do_sys_open(file=/etc/ld.so.cache)
              ls-822   [002] ...2   535.003098: do_syscall_64->do_sys_open(file=/lib64/libselinux.so.1)
              ls-822   [002] ...2   535.004079: do_syscall_64->do_sys_open(file=/lib64/libcap.so.2)
              ls-822   [002] ...2   535.004823: do_syscall_64->do_sys_open(file=/lib64/libc.so.6)
              ls-822   [002] ...2   535.006689: do_syscall_64->do_sys_open(file=/lib64/libpcre.so.1)
              ls-822   [002] ...2   535.007348: do_syscall_64->do_sys_open(file=/lib64/libdl.so.2)
              ls-822   [002] ...2   535.007882: do_syscall_64->do_sys_open(file=/lib64/libpthread.so.0)
              ls-822   [002] ...2   535.012683: do_syscall_64->do_sys_open(file=/usr/lib/locale/locale-archive)
              ls-822   [002] ...2   535.012847: do_syscall_64->do_sys_open(file=/usr/share/locale/locale.alias)
              ls-822   [002] ...2   535.013179: do_syscall_64->do_sys_open(file=/usr/lib/locale/en_US.utf8/LC_IDENTIFICA
              ls-822   [002] ...2   535.013384: do_syscall_64->do_sys_open(file=/usr/lib64/gconv/gconv-modules.cache)
              ls-822   [002] ...2   535.013637: do_syscall_64->do_sys_open(file=/usr/lib/locale/en_US.utf8/LC_MEASUREMEN
              ls-822   [002] ...2   535.013834: do_syscall_64->do_sys_open(file=/usr/lib/locale/en_US.utf8/LC_TELEPHONE)
```

COOL!!!!

# function based events!

- Created in January 2018
- Where are they?



# function based events!

- Created in January 2018
- Where are they?
  - Well, it basically just added a new interface for kprobes
  - Nothing more :-(
- People asked to update kprobes instead

# function based kprobes!



# function based kprobes!

- kprobes have hooked into function tracing for a long time
  - Since 2012
  - If a kprobe is attached to a ftrace nop (start of function on x86)
- ftrace can pass registers (like a breakpoint)
  - Created for kprobes
  - Used by like kernel patching
- Registers give access to parameters



# Updated Kprobes

- Parsing arguments

```
echo ‘p:open do_sys_open file=+0(%si):string’ > kprobe_events
```

# Updated Kprobes

- Parsing arguments

```
echo ‘p:open do_sys_open file=+0($arg2):string’ > kprobe_events
```

# Updated Kprobes

- Parsing arguments

```
echo ‘p:open do_sys_open file=+0($arg2):string’ > kprobe_events

echo ‘do_sys_open(NULL, string file)’ > function_events
```

# Updated Kprobes

- Parsing arguments

```
echo ‘p:open do_sys_open file=+0($arg2):string’ > kprobe_events

echo ‘do_sys_open(NULL, string file)’ > function_events
```

- Doesn’t automatically get the parent either

# Let’s get CRAZY!

vmware

# Let’s get CRAZY!

- Getting a parameter is nice, but we want more!

# Let’s get CRAZY!

- Getting a parameter is nice, but we want more!

```
gdb vmlinux
Reading symbols from vmlinux...done.
(gdb) li ip_rcv
407
408    /*
409     *     Main IP Receive routine.
410     */
411    int ip_rcv(struct sk_buff *skb, struct net_device *dev, struct packet_type *pt, struct net_device *orig_dev)
412    {
413           const struct iphdr *iph;
414           struct net *net;
415           u32 len;
416
(gdb)
```

# Let’s get CRAZY!

- Getting a parameter is nice, but we want more!

```
gdb vmlinux
Reading symbols from vmlinux...done.
(gdb) li ip_rcv
407
408    /*
409     *     Main IP Receive routine.
410     */
411    int ip_rcv(struct sk_buff *skb, struct net_device *dev, struct packet_type *pt, struct net_device *orig_dev)
412    {
413           const struct iphdr *iph;
414           struct net *net;
415           u32 len;
416
(gdb)
```

# Let’s get CRAZY!

- Getting a parameter is nice, but we want more!

```
(gdb) ptype struct net_device
type = struct net_device {
    char name[16];
    struct hlist_node name_hlist;
    struct dev_ifalias *ifalias;
    unsigned long mem_end;
    unsigned long mem_start;
    unsigned long base_addr;
    int irq;
    unsigned long state;
    struct list_head dev_list;
    struct list_head napi_list;
    struct list_head unreg_list;
    struct list_head close_list;
    struct list_head ptype_all;
    struct list_head ptype_specific;
```

# More with kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:net ip_rcv dev=$arg2 name=+0($arg2):string' > kprobe_events
# echo 1 > events/kprobes/enable
# cat trace
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
          <idle>-0     [000] ..s2 13651.243916: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.244376: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.666827: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.668604: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.756301: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.758255: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.817958: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.819973: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.874055: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.876018: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.137970: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.139764: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.178533: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.180256: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.265574: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.267472: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
```

# More with kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:net ip_rcv dev=$arg2 name=+0($arg2):string' > kprobe_events
# echo 1 > events/kprobes/enable
# cat trace
# tracer: nop
#
#                                _-----=> irqs-off
#                               / _----=> need-resched
#                              | / _---=> hardirq/softirq
#                              || / _--=> preempt-depth
#                              ||| / _-=> delay
#           TASK-PID     CPU#  ||||   TIMESTAMP  FUNCTION
#              | |         |   ||||      |         |
          <idle>-0     [000] ..s2 13651.[illegible]: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.244376: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.666827: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.668604: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.756301: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.758255: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.817958: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.819973: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.874055: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13651.876018: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.137970: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.139764: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.178533: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.180256: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.265574: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
          <idle>-0     [000] ..s2 13652.267472: net: (ip_rcv+0x0/0x150) dev=0xffff9727f1dcc2c0 name="ens9"
```

It's OK

# Let’s get CRAZY!

- Getting a parameter is nice, but we want more!

```
(gdb) ptype struct net_device
type = struct net_device {
[..]
    unsigned int mtu;
    unsigned int min_mtu;
    unsigned int max_mtu;
    unsigned short type;
    unsigned short hard_header_len;
    unsigned char min_header_len;
    unsigned short needed_headroom;
    unsigned short needed_tailroom;
    unsigned char perm_addr[32];
    unsigned char addr_assign_type;
    unsigned char addr_len;
    unsigned short neigh_priv_len;
    unsigned short dev_id;
    unsigned short dev_port;
    spinlock_t addr_list_lock;
    unsigned char name_assign_type;
    bool uc_promisc;
```

# Let’s get CRAZY!

- Getting a parameter is nice, but we want more!

```
(gdb) print (int)&((struct net_device *)0)->perm_addr
$1 = 574
```

# More with kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:net ip_rcv name=+0($arg2):string a1=+574($arg2):x8 a2=+575($arg2):x8
a3=+576($arg2):x8 a4=+577($arg2):x8 a5=+578($arg2):x8 a6=+579($arg2):x8' > kprobe_events
# echo 1 > events/kprobes/enable
# cat trace
# tracer: nop
#
#                               _-----=> irqs-off
#                              / _----=> need-resched
#                             | / _---=> hardirq/softirq
#                             || / _--=> preempt-depth
#                             ||| /     delay
#           TASK-PID   CPU#   ||||    TIMESTAMP  FUNCTION
#              | |       |    ||||       |         |
            sshd-737   [000] ..s2 14814.780017: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14814.781802: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
          <idle>-0     [000] ..s2 14815.030452: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.032377: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
          <idle>-0     [000] ..s2 14815.126808: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.128838: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
          <idle>-0     [000] ..s2 14815.291484: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.293853: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.324377: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.325724: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.327043: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.329848: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.334772: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
          <idle>-0     [000] ..s2 14815.770628: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.771532: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
          <idle>-0     [000] ..s2 14815.919881: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.923384: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
```

# More with kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:net ip_rcv name=+0($arg2):string a1=+574($arg2):x8 a2=+575($arg2):x8
a3=+576($arg2):x8 a4=+577($arg2):x8 a5=+578($arg2):x8 a6=+579($arg2):x8' > kprobe_events
# echo 1 > events/kprobes/enable
# cat trace
# tracer: nop
#
#                               _-----=> irqs-off
#                              / _----=> need-resched
#                             | / _---=> hardirq/softirq
#                             || / _--=> preempt-depth
#                             ||| /     delay
#           TASK-PID   CPU#   ||||    TIMESTAMP  FUNCTION
#              | |       |    ||||       |         |
            sshd-737   [000] ..s2 14814.780017: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14814.781802: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
          <idle>-0     [000] ..s2 14815.030452: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.032377: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
          <idle>-0     [000] ..s2 14815.126808: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.128838: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
          <idle>-0     [000] ..s2 14815.291484: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.293853: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.324377: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.325724: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.327043: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.329848: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.334772: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
          <idle>-0     [000] ..s2 14815.770628: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.771532: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
          <idle>-0     [000] ..s2 14815.919881: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
            sshd-737   [000] ..s2 14815.923384: net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
```

# More with kprobes

```
net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
```

```
# ifconfig ens9
ens9: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.122.63  netmask 255.255.255.0  broadcast 192.168.122.255
        inet6 fe80::2f86:705e:78e3:c516  prefixlen 64  scopeid 0x20<link>
        ether 52:54:00:c0:76:ec  txqueuelen 1000  (Ethernet)
        RX packets 2655  bytes 193170 (188.6 KiB)
        RX errors 0  dropped 64  overruns 0  frame 0
        TX packets 1500  bytes 214413 (209.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 7418
```

# More with kprobes

```
net: (ip_rcv+0x0/0x150) name="ens9" a1=0x52 a2=0x54 a3=0x0 a4=0xc0 a5=0x76 a6=0xec
```

```
# ifconfig ens9
ens9: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.122.63  netmask 255.255.255.0  broadcast 192.168.122.255
        inet6 fe80::2[illegible]e:7[illegible]e[illegible].c5[illegible]6  prefixlen 64  scopeid 0x20<link>
        ether 52:54:0[illegible]:c[illegible]7:[illegible]  txqueuelen 1000  (Ethernet)
        RX packets 265[illegible]  bytes 193[illegible] (18[illegible] KiB)
        RX errors 0  dropped 64  overruns 0  frame 0
        TX packets 1500  bytes 214413 (209.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 7418
```

GROOVY!!!

vmware

# Let’s go NUTS!

vmware

# Let’s go NUTS!

- How far can we go?

```
(gdb) li __vfs_read
409           return ret;
410    }
411
412    ssize_t __vfs_read(struct file *file, char __user *buf, size_t count,
413                       loff_t *pos)
414    {
415           if (file->f_op->read)
416                  return file->f_op->read(file, buf, count, pos);
417           else if (file->f_op->read_iter)
418                  return new_sync_read(file, buf, count, pos);
(gdb)
```

# Let’s go NUTS!

- How far can we go?

```
(gdb) ptype struct file
type = struct file {
    union {
        struct llist_node fu_llist;
        struct callback_head fu_rcuhead;
    } f_u;
    struct path f_path;
    struct inode *f_inode;
    const struct file_operations *f_op;
    spinlock_t f_lock;
    enum rw_hint f_write_hint;
    atomic_long_t f_count;
    unsigned int f_flags;
    fmode_t f_mode;
    struct mutex f_pos_lock;
    loff_t f_pos;
    struct fown_struct f_owner;
    const struct cred *f_cred;
    struct file_ra_state f_ra;
    u64 f_version;
    void *f_security;
    void *private_data;
    struct list_head f_ep_links;
    struct list_head f_tfile_llink;
    struct address_space *f_mapping;
    errseq_t f_wb_err;
}
```

# Let’s go NUTS!

- How far can we go?

```
(gdb) ptype struct inode
type = struct inode {
    umode_t i_mode;
    unsigned short i_opflags;
    kuid_t i_uid;
    kgid_t i_gid;
    unsigned int i_flags;
    struct posix_acl *i_acl;
    struct posix_acl *i_default_acl;
    const struct inode_operations *i_op;
     struct super_block *i_sb;
    struct address_space *i_mapping;
    void *i_security;
    unsigned long i_ino;
    union {
        const unsigned int i_nlink;
        unsigned int __i_nlink;
    };
    dev_t i_rdev;
    loff_t i_size;
    struct timespec64 i_atime;
    struct timespec64 i_mtime;
    struct timespec64 i_ctime;
```

# Let’s go NUTS!

- How far can we go?

```
(gdb) ptype struct super_block
type = struct super_block {
    struct list_head s_list;
    dev_t s_dev;
    unsigned char s_blocksize_bits;
    unsigned long s_blocksize;
    loff_t s_maxbytes;
    struct file_system_type *s_type;
    const struct super_operations *s_op;
    const struct dquot_operations *dq_op;
    const struct quotactl_ops *s_qcop;
    const struct export_operations *s_export_op;
    unsigned long s_flags;
    unsigned long s_iflags;
    unsigned long s_magic;
    struct dentry *s_root;
    struct rw_semaphore s_umount;
    int s_count;
    atomic_t s_active;
    void *s_security;
    const struct xattr_handler **s_xattr;
    const struct fscrypt_operations *s_cop;
```

# Let’s go NUTS!

- How far can we go?

```
(gdb) ptype struct file_system_type
type = struct file_system_type {
    const char *name;
    int fs_flags;
    struct dentry *(*mount)(struct file_system_type *, int, const char *, void *);
    void (*kill_sb)(struct super_block *);
    struct module *owner;
    struct file_system_type *next;
    struct hlist_head fs_supers;
    struct lock_class_key s_lock_key;
    struct lock_class_key s_umount_key;
    struct lock_class_key s_vfs_rename_key;
    struct lock_class_key s_writers_key[3];
    struct lock_class_key i_lock_key;
    struct lock_class_key i_mutex_key;
    struct lock_class_key i_mutex_dir_key;
}
(gdb)
```

# Let’s go NUTS!

- How far can we go?

```
(gdb) print (int)&((struct file *)0)->f_inode
$2 = 32
(gdb) print (int)&((struct inode *)0)->i_sb
$3 = 40
(gdb) print (int)&((struct super_block *)0)->s_type
$4 = 40
(gdb) print (int)&((struct file_system_type *)0)->name
$5 = 0
```

# More with kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:crazy name=+0(+0(+40(+40(+32($arg1))))):string' > kprobe_events
# echo 1 > events/kprobes/enable
# cat trace
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
            sshd-737   [000] ...1 17160.284246: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17160.451834: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            bash-739   [003] ...1 17160.452430: crazy: (__vfs_read+0x0/0x180) name="devpts"
            sshd-737   [000] ...1 17160.452467: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17160.571763: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            bash-739   [003] ...1 17160.572341: crazy: (__vfs_read+0x0/0x180) name="devpts"
            sshd-737   [000] ...1 17160.572386: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17160.614499: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            bash-739   [003] ...1 17160.615195: crazy: (__vfs_read+0x0/0x180) name="devpts"
            sshd-737   [000] ...1 17160.615233: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17161.133741: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            sshd-737   [000] ...1 17161.134373: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
             cat-877   [001] ...1 17161.140369: crazy: (__vfs_read+0x0/0x180) name="pipefs"
             cat-877   [001] ...1 17161.141364: crazy: (__vfs_read+0x0/0x180) name="ext4"
             cat-877   [001] ...1 17161.141738: crazy: (__vfs_read+0x0/0x180) name="ext4"
             cat-877   [001] ...1 17161.141778: crazy: (__vfs_read+0x0/0x180) name="ext4"
```

# More with kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:crazy name=+0(+0(+40(+40(+32($arg1))))):string' > kprobe_events
# echo 1 > events/kprobes/enable
# cat trace
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
            sshd-737   [000] ...1 17160.284246: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17160.451834: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            bash-739   [003] ...1 17160.452430: crazy: (__vfs_read+0x0/0x180) name="devpts"
            sshd-737   [000] ...1 17160.452467: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17160.571763: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            bash-739   [003] ...1 17160.572341: crazy: (__vfs_read+0x0/0x180) name="devpts"
            sshd-737   [000] ...1 17160.572386: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17160.614499: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            bash-739   [003] ...1 17160.615195: crazy: (__vfs_read+0x0/0x180) name="devpts"
            sshd-737   [000] ...1 17160.615233: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17161.133741: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            sshd-737   [000] ...1 17161.134373: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
             cat-877   [001] ...1 17161.140369: crazy: (__vfs_read+0x0/0x180) name="pipefs"
             cat-877   [001] ...1 17161.141364: crazy: (__vfs_read+0x0/0x180) name="ext4"
             cat-877   [001] ...1 17161.141738: crazy: (__vfs_read+0x0/0x180) name="ext4"
             cat-877   [001] ...1 17161.141778: crazy: (__vfs_read+0x0/0x180) name="ext4"
```

# More with kprobes

```
# cd /sys/kernel/tracing/events
# echo 'p:crazy name=+0(+0(+40(+40(+32($arg1))))):string' > kprobe_events
# echo 1 > events/kprobes/enable
# cat trace
# tracer: nop
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||   TIMESTAMP  FUNCTION
#              | |       |   ||||      |         |
            sshd-737   [000] ...1 17160.451834: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            bash-739   [003] ...1 17160.452430: crazy: (__vfs_read+0x0/0x180) name="devpts"
            sshd-737   [000] ...1 17160.452467: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17160.571763: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            bash-739   [003] ...1 17160.572341: crazy: (__vfs_read+0x0/0x180) name="devpts"
            sshd-737   [000] ...1 17160.572386: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17160.614499: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            bash-739   [003] ...1 17160.615195: crazy: (__vfs_read+0x0/0x180) name="devpts"
            sshd-737   [000] ...1 17160.615233: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
            sshd-737   [000] ...1 17161.133741: crazy: (__vfs_read+0x0/0x180) name="sockfs"
            sshd-737   [000] ...1 17161.134373: crazy: (__vfs_read+0x0/0x180) name="devtmpfs"
             cat-877   [001] ...1 17161.140369: crazy: (__vfs_read+0x0/0x180) name="pipefs"
             cat-877   [001] ...1 17161.141364: crazy: (__vfs_read+0x0/0x180) name="ext4"
             cat-877   [001] ...1 17161.141738: crazy: (__vfs_read+0x0/0x180) name="ext4"
             cat-877   [001] ...1 17161.141778: crazy: (__vfs_read+0x0/0x180) name="ext4"
```

AWESOME!!!!

# In Conclusion

- Dynamic tracing with kprobes is:



# In Conclusion

- Dynamic tracing with kprobes is:

**EXCITING!!!!**

**COOL!!!!**

**GROOVY!!!**

**AWESOME!!!!**



# Questions?