Sources of Latency
And where to find them

Steven Rostedt
srostedt@vmware.com
@VMWopensource
blogs.vmware.com/opensource
What is Latency?
What is Latency?

“Latency is a time interval between the stimulation and response, or, from a more general point of view, a time delay between the cause and the effect of some physical change in the system being observed.” - Wikipedia
What is Latency?

- The time from when an event is suppose to happen to the time it actually does happen.
What is Latency?

• The time from when an event is suppose to happen to the time it actually does happen.
What is Latency?

- The time from when an event is suppose to happen to the time it actually does happen.
What is Latency?

- The time from when an event is suppose to happen to the time it actually does happen.
Where does latency come from?

HARDWARE

BIOS

KERNEL

Library

Application
Where does latency come from?

- **HARDWARE**
  - BIOS
  - Kernel
  - Library
  - Application
Latency from Hardware
Latency from Hardware

- System Management Interrupt (SMI)
Latency from Hardware

- System Management Interrupt (SMI)
- Cache miss (instruction cache, data cache)
Latency from Hardware

- System Management Interrupt (SMI)
- Cache miss (instruction cache, data cache)
- Branch prediction
Latency from Hardware

- System Management Interrupt (SMI)
- Cache miss (instruction cache, data cache)
- Branch prediction
- Hyper-threading - Simultaneous multithreading (SMT)
Latency from Hardware

- System Management Interrupt (SMI)
- Cache miss (instruction cache, data cache)
- Branch prediction
- Hyper-threading - Simultaneous multithreading (SMT)
- Page fault / Translation Lookaside Buffer (TLB)
The Hardware Latency Detector

- CONFIG_HWLAT_TRACER
The Hardware Latency Detector

- CONFIG_HWLAT_TRACER
- Available on my Fedora 31 system
  - But not on my Debian system
The Hardware Latency Detector

- CONFIG_HWLAT_TRACER
- Available on my Fedora 31 system
  - But not on my Debian system
- Runs in a tight loop with interrupts disabled
The Hardware Latency Detector

- **CONFIG_HWLAT_TRACER**
- Available on my Fedora 31 system
  - But not on my Debian system
- Runs in a tight loop with interrupts disabled
- Runs for width microseconds in window
  - `/sys/kernel/tracing/hwlat_detector/{width,window}`
    (default 500,000us in 1,000,000us or 1/2 second per second)
  - Then moves to another CPU
  - `/sys/kernel/tracing/tracing_cpumask`
The Hardware Latency Detector

- **CONFIG_HWLAT_TRACER**
- Available on my Fedora 31 system
  - But not on my Debian system
- Runs in a tight loop with interrupts disabled
- Runs for width microseconds in window
  - `/sys/kernel/tracing/hwlat_detector/{width,window}`
    (default 500,000us in 1,000,000us or 1/2 second per second)
  - Then moves to another CPU
  - `/sys/kernel/tracing/tracing_cpumask`
  - `/sys/kernel/tracing/tracing_thresh` (record if greater than this - microseconds)
The Hardware Latency Detector

```
# mount -t tracefs /sys/kernel/tracing
# cd /sys/kernel/tracing
# cat hwlat_detector/width
500000

# echo 900000 > hwlat_detector/width
# echo hwlat > current_tracer
# cat tracing_thresh
10

# sleep 100
# cat trace

# tracer: hwlat
#
# entries-in-buffer/entries-written: 3/3  #P:8
#
#                              _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID  CPU# ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
<...>-211151 [004] d... 369002.142479: #1 inner/outer(us):  14/15  ts:1591572180.015876666 count:42
<...>-211151 [006] d... 369012.222418: #2 inner/outer(us):  12/17  ts:1591572189.780477422 count:13
<...>-211151 [001] d... 369063.629994: #3 inner/outer(us):   0/11  ts:1591572241.260867298 count:1
```
The Hardware Latency Detector

# mount -t tracefs /sys/kernel/tracing
# cd /sys/kernel/tracing
# cat hwlat_detector/width

500000

# echo 900000 > hwlat_detector/width
# echo hwlat > current_tracer
# cat tracing_thresh

10

# sleep 100
# cat trace

# tracer: hwlat
#
# entries-in-buffer/entries-written: 3/3   #P:8
#
#    _-----=> irqs-off
#    /-----=> need-resched
#   | /-----=> hardirq/softirq
#   || /--=> preempt-depth
#   ||| / delay
# TASK-PID  CPU#   |   TIMESTAMP  FUNCTION
#   |   |       |   |         |
<...>-211151 [004] d... 369002.142479: #1 inner/outer(us): 14/15 ts:1591572180.015876666 count:42
<...>-211151 [006] d... 369012.222418: #2 inner/outer(us): 12/17 ts:1591572189.780477422 count:13
<...>-211151 [001] d... 369063.629994: #3 inner/outer(us): 0/11 ts:1591572241.260867298 count:1
The Hardware Latency Detector

```
# mount -t tracefs /sys/kernel/tracing
# cd /sys/kernel/tracing
# cat hwlat_detector/width
500000

# echo 900000 > hwlat_detector/width
# echo hwlat > current_tracer
# cat tracing_thres
10

# sleep 100
# cat trace

# tracer: hwlat
# entries-in-buffer/entries-written: 3/3   #P:8
#               _-----=> irqs-off
#                /-----=> need-resched
#                 | /-----=> hardirq/softirq
#                 || /-----=> preempt-depth
#                 ||| /     delay
# shortly: TASK-PID CPU# | ||| TIMESTAMP FUNCTION
#                | |       |   ||||       |         |
<...>-211151 [004] d... 369062.142479: #1 inner/outer(us): 14/15 ts:1591572180.015876666 count:42
<...>-211151 [006] d... 369012.222418: #2 inner/outer(us): 12/17 ts:1591572189.780477422 count:13
<...>-211151 [001] d... 369063.629994: #3 inner/outer(us): 0/11 ts:1591572241.260867298 count:1
```
The Hardware Latency Detector

```shell
# mount -t tracefs /sys/kernel/tracing
# cd /sys/kernel/tracing
# cat hwlat_detector/width
500000

# echo 900000 > hwlat_detector/width
# echo hwlat > current_tracer
# cat tracing_thresh
10

# sleep 100
# cat trace

# tracer: hwlat
# entries-in-buffer/entries-written: 3/3   #P:8
#
# _-----=> irqs-off
# /-----=> need-resched
# | /---=> hardirq/softirq
# || /--=> preempt-depth
# ||| /  delay
# TASK-PID CPU#   TIMESTAMP FUNCTION
# <...>-211151 [004] d... 369002.142479: #1 inner/outer(us): 14/15 ts:1591572180.015876666 count:42
# <...>-211151 [006] d... 369012.222418: #2 inner/outer(us): 12/17 ts:1591572189.780477422 count:13
# <...>-211151 [001] d... 369063.629994: #3 inner/outer(us): 0/11 ts:1591572241.260867298 count:1
```
The Hardware Latency Detector

# mount -t tracefs /sys/kernel/tracing
# cd /sys/kernel/tracing
# cat hwlat_detector/width

500000

# echo 900000 > hwlat_detector/width
# echo hwlat > current_tracer
# cat tracing_thresh

10

# sleep 100
# cat trace

# tracer: hwlat
#
# entries-in-buffer/entries-written: 3/3   #P:8
#
# ___-----=> irqs-off
# /-----=> need-resched
# |__/----=> hardirq/softirq
# ||//--=> preempt-depth
# |||/    delay
# TASK-PID CPU# |||| TIMESTAMP FUNCTION
# <...>-211151 [004] d... 369002.142479: #1 inner/outer(us): 14/15 ts:1591572180.015876666 count:42
# <...>-211151 [006] d... 369012.222418: #2 inner/outer(us): 12/17 ts:1591572189.780477422 count:13
# <...>-211151 [001] d... 369063.629994: #3 inner/outer(us): 0/11 ts:1591572241.260867298 count:1
The Hardware Latency Detector

# mount -t tracefs /sys/kernel/tracing
# cd /sys/kernel/tracing
# cat hwlat_detector/width

500000

# echo 900000 > hwlat_detector/width
# echo hwlat > current_tracer
# cat tracing_thresh

10

# sleep 100
# cat trace

# tracer: hwlat
#
# entries-in-buffer/entries-written: 3/3  #P:8
#
#                                _-----=> irqs-off
#                             / _----=> need-resched
#                            | / _---=> hardirq/softirq
#                            || / _--=> preempt-depth
#                            ||| /     delay
#           TASK-PID   CPU#  ||||    TIMESTAMP  FUNCTION
#              | |       |   ||||       |         |
<...>-211151 [004] d... 369002.142479: #1  inner/outer(us): 14/15  ts:1591572180.015876666 count:42
<...>-211151 [006] d... 369012.222418: #2  inner/outer(us): 12/17  ts:1591572189.780477422 count:13
<...>-211151 [001] d... 369063.629994: #3  inner/outer(us):  0/11  ts:1591572241.260867298 count:1
The Hardware Latency Detector

```bash
# mount -t tracefs /sys/kernel/tracing
# cd /sys/kernel/tracing
# cat hwlat_detector/width

500000

# echo 900000 > hwlat_detector/width
# echo hwlat > current_tracer
# cat tracing_thresh

10

# sleep 100
# cat trace

# tracer: hwlat

# entries-in-buffer/entries-written: 3/3   #P:8

#
#   -------=> irqs-off
#   |      ----=> need-resched
#   |          ---=> hardirq/softirq
#   |          ----=> preempt-depth
#   |          |      delay
#   TASK-PID  CPU#    |   TIMESTAMP   FUNCTION
#                   |            |            |
# <...>-211151 [004] d... 369002.142479: #1 inner/outer(us):   14/15 ts:1591572180.015876666 count:42
# <...>-211151 [006] d... 369012.222418: #2 inner/outer(us):   12/17 ts:1591572189.780477422 count:13
# <...>-211151 [001] d... 369063.629994: #3 inner/outer(us):    0/11 ts:1591572241.260867298 count:1
```
The Hardware Latency Detector

```
# mount -t tracefs /sys/kernel/tracing
# cd /sys/kernel/tracing
# cat hwlat_detector/width
500000

# echo 900000 > hwlat_detector/width
# echo hwlat > current_tracer
# cat tracing_thresh
10

# sleep 100
# cat trace

# tracer: hwlat
# # entries-in-buffer/entries-written: 3/3   #P:8
# #
# #          _-----=> irqs-off
# #           /-----=> need-resched
# #            |  ---=> hardirq/softirq
# #            |         preempt-depth
# #            ||        delay
# #
# # TASK-PID  CPU#  TIMESTAMP  FUNCTION
# #
# <...>-211151 [004] d... 369002.142479: #1 inner/outer(us): 14/15 ts:1591572180.015876666 count:42
# <...>-211151 [006] d... 369012.222418: #2 inner/outer(us): 12/17 ts:1591572189.780477422 count:13
# <...>-211151 [001] d... 369063.629994: #3 inner/outer(us): 0/11 ts:1591572241.260867298 count:1
```
last_t2 = 0

\[ t1 = \text{time\_get()} \]
\[ t2 = \text{time\_get()} \]

\[ \text{last\_t2} == 0 \]
\[ \text{diff} = t1 - \text{last\_t2} \]
\[ \text{diff} > \text{outer} \]
\[ \text{outer} = \text{diff} \]

\[ \text{diff} = t2 - t1 \]
\[ \text{last\_t2} = t2 \]

\[ \text{diff} > \text{inner} \]
\[ \text{inner} = \text{diff} \]

\[ \text{time} < \text{width} \]

exit loop
Latency from Hardware

![Diagram showing latency from Hardware with layers: BIOS, Kernel, Library, Application]
Latency from the Kernel

• Interrupt latency
  - Interrupt handlers (process must wait for interrupts)
  - Interrupts disabled (interrupt must wait for CPU)
Latency from the Kernel

- **Interrupt latency**
  - Interrupt handlers (process must wait for interrupts)
  - Interrupts disabled (interrupt must wait for CPU)

- **I/O Latency**
  - Wait for a device to do something
Latency from the Kernel

• Interrupt latency
  - Interrupt handlers (process must wait for interrupts)
  - Interrupts disabled (interrupt must wait for CPU)

• I/O Latency
  - Wait for a device to do something

• Kernel maintenance tasks
  - Lots of tasks to keep your computer running smoothly
Interrupt Latency

- What are interrupts?
Interrupt handlers
Interrupts disabled!
Latency from interrupts

Interrupt handlers

Program

Interrupt

Device triggers interrupt
Latency from interrupts

Interrupt handlers

Program

Device triggers interrupt

Interrupt

latency
Latency from interrupts

Why this delay?

Device triggers interrupt

Interrupt
Latency from interrupts

Why this delay?

Interrupt latency

Program

Device triggers interrupt

Interrupt
Latency from interrupts

Latency of expected response

Interrupt

Program

Device triggers interrupt

Interrupt latency
Ftrace and trace-cmd

- ftrace - The tracing infrastructure in the Linux kernel
- trace-cmd - Command line interface to ftrace

https://trace-cmd.org

git://git.kernel.org/pub/scm/utils/trace-cmd/trace-cmd.git
Measuring latency from interrupts

You can easily trace the latency from interrupts

• For x86:

```
# trace-cmd record -p function_graph -l do_IRQ \n -l "*_interrupt" -e irq_handler_entry
```
Tracing Latency from Interrupts

# trace-cmd report -l --cpu 3

<idle>-0 3d..1 1378.332577: funcgraph_entry: irq=26 name=em1
<idle>-0 3d.h1 1378.332584: irq_handler_entry: irq=26 name=em1
<idle>-0 3d.h1 1378.332591: funcgraph_entry: do_IRQ()
<idle>-0 3d.h1 1378.332591: funcgraph_exit: note_interrupt()
<idle>-0 3d..1 1378.332674: funcgraph_exit: do_IRQ()
<idle>-0 3d.h1 1378.332759: irq_handler_entry: irq=26 name=em1
<idle>-0 3d.h1 1378.332766: funcgraph_entry: note_interrupt()
<idle>-0 3d.h1 1378.332766: funcgraph_exit: hrtimer_interrupt()
<idle>-0 3d.h1 1378.332766: funcgraph_exit: __next_timer_interrupt()
<idle>-0 3d.s2 1378.332776: funcgraph_entry: smp_irq_work_interrupt()
<idle>-0 3d..1 1378.332776: funcgraph_exit: + 24.779 us
<idle>-0 3d.h1 1378.333014: funcgraph_entry: smp_apic_timer_interrupt()
<idle>-0 3d.h1 1378.333020: funcgraph_entry: hrtimer_interrupt()
<idle>-0 3d.h1 1378.333020: funcgraph_entry: __next_timer_interrupt()
<idle>-0 3d.s2 1378.333030: funcgraph_entry: smp_irq_work_interrupt()
<idle>-0 3d.s2 1378.333032: funcgraph_entry: __next_timer_interrupt()
<idle>-0 3d.s2 1378.333032: funcgraph_entry: smp_irq_work_interrupt()
<idle>-0 3d.s2 1378.333030: funcgraph_entry: hrtimer_interrupt()
<idle>-0 3dNs2 1378.333030: funcgraph_entry: smp_apic_timer_interrupt()
<idle>-0 3dNs2 1378.333032: funcgraph_entry: hrtimer_interrupt()
<idle>-0 3dNs2 1378.333050: funcgraph_exit: + 10.857 us
<idle>-0 3dN.s2 1378.333066: funcgraph_exit: + 52.353 us
<idle>-0 3dN.s2 1378.333066: funcgraph_exit: smp_apic_timer_interrupt()
<idle>-0 3dN.s2 1378.333066: funcgraph_exit: hrtimer_interrupt()
<idle>-0 3d.h1 1378.334025: funcgraph_entry: smp_apic_timer_interrupt()
<idle>-0 3d.h1 1378.334030: funcgraph_entry: hrtimer_interrupt()
<idle>-0 3d.h1 1378.334030: funcgraph_entry: smp_irq_work_interrupt()
<idle>-0 3d.h1 1378.334044: funcgraph_exit: smp_irq_work_interrupt()
<idle>-0 3d.h1 1378.334044: funcgraph_exit: hrtimer_interrupt()
<idle>-0 3d.h1 1378.334051: funcgraph_exit: + 13.711 us
<idle>-0 3d..1 1378.334051: funcgraph_exit: + 27.302 us
Tracing Latency from Interrupts

```
# trace-cmd report -l --cpu 3

<idle>-0  3d..1 1378.332577: funcgraph_entry:                      |
<idle>-0  3d..1 1378.332584: irq_handler_entry:                     |
<idle>-0  3d..1 1378.332591: funcgraph_entry:                        |
<idle>-0  3d..1 1378.332598: funcgraph_exit:                        |
<idle>-0  3d..1 1378.332617: funcgraph_entry:                        |
<idle>-0  3d..1 1378.332630: funcgraph_entry:                        |
<idle>-0  3d..1 1378.332650: funcgraph_exit:                        |
<idle>-0  3d..1 1378.332674: funcgraph_entry:                        |
<idle>-0  3d..1 1378.332752: funcgraph_entry:                        |
<idle>-0  3d..1 1378.332759: irq_handler_entry:                     |
<idle>-0  3d..1 1378.332766: funcgraph_entry:                        |
<idle>-0  3d..1 1378.332766: funcgraph_exit:                        |
<idle>-0  3d..1 1378.332776: funcgraph_entry:                        |
<idle>-0  3d..1 1378.332776: funcgraph_exit:                        |
<idle>-0  3d..1 1378.333014: funcgraph_entry:                        |
<idle>-0  3d..1 1378.333014: funcgraph_exit:                        |
<idle>-0  3d..1 1378.333020: funcgraph_entry:                        |
<idle>-0  3d..1 1378.333020: funcgraph_exit:                        |
<idle>-0  3d..1 1378.333030: funcgraph_entry:                        |
<idle>-0  3d..1 1378.333030: funcgraph_exit:                        |
<idle>-0  3d..1 1378.333032: funcgraph_entry:                        |
<idle>-0  3d..1 1378.333032: funcgraph_exit:                        |
<idle>-0  3d.s2  1378.333033: funcgraph_exit:                       |
<idle>-0  3d.s2  1378.333033: funcgraph_entry:                      |
<idle>-0  3d.s2  1378.333034: funcgraph_exit:                       |
<idle>-0  3d.s2  1378.333034: funcgraph_entry:                      |
<idle>-0  3d.s2  1378.333036: funcgraph_entry:                      |
<idle>-0  3d.s2  1378.333036: funcgraph_exit:                       |
<idle>-0  3dNs2 1378.333050: funcgraph_exit:                        |
<idle>-0  3dNs2 1378.333050: funcgraph_entry:                      |
<idle>-0  3dN.1 1378.333066: funcgraph_exit:                        |
<idle>-0  3dN.1 1378.333066: funcgraph_entry:                      |
<idle>-0  3d.h1 1378.334025: funcgraph_entry:                      |
<idle>-0  3d.h1 1378.334025: funcgraph_exit:                        |
<idle>-0  3d.h1 1378.334030: funcgraph_entry:                      |
<idle>-0  3d.h1 1378.334030: funcgraph_exit:                        |
<idle>-0  3d.h1 1378.334044: funcgraph_exit:                        |
<idle>-0  3d.h1 1378.334044: funcgraph_entry:                      |
<idle>-0  3d..1 1378.334051: funcgraph_exit:                        |
<idle>-0  3d..1 1378.334051: funcgraph_entry:                      |

do_IRQ() {
  note_interrupt() {
    irq=26 name=em1
  }
}

do_IRQ() {
  note_interrupt() {
    irq=26 name=em1
  }
}

smp_apic_timer_interrupt() {
  hrtimer_interrupt() {
    __next_timer_interrupt() {
      smp_irq_work_interrupt() {
      }
    }
  }
}

```

Tracing Latency from Interrupts

# trace-cmd report -l --cpu 3

<idle>-0  3d..1 1378.332577: funcgraph_entry:
<idle>-0  3d.h1 1378.332584: irq_handler_entry: irq=26 name=em1
<idle>-0  3d.h1 1378.332591: funcgraph_entry: note_interrupt()
<idle>-0  3d..1 1378.332584: funcgraph_exit: + 58.288 us
<idle>-0  3d.h1 1378.332759: irq_handler_entry:
<idle>-0  3d.h1 1378.332766: funcgraph_entry: note_interrupt()
<idle>-0  3d.h1 1378.332766: funcgraph_exit: + 0.637 us
<idle>-0  3d..1 1378.332776: funcgraph_exit: + 24.779 us
<idle>-0  3d..1 1378.333014: funcgraph_entry:
<idle>-0  3d.h1 1378.333020: funcgraph_entry:
<idle>-0  3d.h1 1378.333030: funcgraph_exit: + 13.711 us
<idle>-0  3d.s2 1378.333032: funcgraph_entry:
<idle>-0  3d.s2 1378.333033: funcgraph_exit: + 10.857 us
<idle>-0  3d.s2 1378.333039: funcgraph_entry:
<idle>-0  3dNs2 1378.333050: funcgraph_exit: + 10.857 us
<idle>-0  3dNs2 1378.333058: funcgraph_exit: + 10.857 us
<idle>-0  3d..1 1378.333066: funcgraph_exit: + 52.353 us
<idle>-0  3d.h1 1378.334025: funcgraph_entry:
<idle>-0  3d.h1 1378.334030: funcgraph_exit: + 13.711 us
<idle>-0  3d.h1 1378.334044: funcgraph_exit: + 27.302 us
Tracing Latency from Interrupts

```
# trace-cmd report -l --cpu 3

<idle>-0  3d..1 1378.332577: funcgraph_entry:                   |  do_IRQ() {
<idle>-0  3d.h1 1378.332584: irq_handler_entry:    irq=26 name=em1
<idle>-0  3d.h1 1378.332591: funcgraph_exit:         0.627 us   |    note_interrupt() {
<idle>-0  3d..1 1378.332674: funcgraph_exit:       + 98.288 us
<idle>-0  3d.h1 1378.332759: irq_handler_entry:    irq=26 name=em1
<idle>-0  3d.h1 1378.332766: funcgraph_entry:                   |    note_interrupt() {
<idle>-0  3d.s2 1378.332776: funcgraph_exit:       24.779 us
<idle>-0  3d.s2 1378.333014: funcgraph_entry:                   |    hrtimer_interrupt() {
<idle>-0  3d.s2 1378.333030: funcgraph_exit:         9.499 us   |      smp_irq_work_interrupt() {
<idle>-0  3dNs2 1378.333050: funcgraph_exit:       + 10.857 us
<idle>-0  3dN.1 1378.333066: funcgraph_exit:         + 52.353 us
<idle>-0  3d..1 1378.333020: funcgraph_entry:                   |    hrtimer_interrupt() {
<idle>-0  3d..1 1378.333030: funcgraph_exit:         9.499 us   |      smp_irq_work_interrupt() {
<idle>-0  3dNs2 1378.333050: funcgraph_exit:       + 10.857 us
<idle>-0  3dN.1 1378.333066: funcgraph_exit:         + 52.353 us
```

This is not a Real Time Kernel!
Tracing Latency from Interrupts with PREEMPT_RT (5.4.14-rt7)

# trace-cmd report -l --cpu 4

<idle>-0  4d..10  2850.449996: funcgraph_entry: get_next_timer_interrupt() {
    __next_timer_interrupt() {
        0.594 us
    }
}

<idle>-0  4d..20  2850.449997: funcgraph_entry: smp_apic_timer_interrupt() {
    hrtimer_interrupt() {
        2.260 us
    }
}

<idle>-0  4d.h10  2851.281933: funcgraph_entry: hrtimer_interrupt() {
    __next_timer_interrupt() {
        4.919 us
    }
}

<idle>-0  4d..10  2851.281951: funcgraph_exit: + 19.301 us

ksoftirq-45  4d..13  2851.281962: funcgraph_entry: __next_timer_interrupt() {
    1.639 us
}

ksoftirq-45  4d..13  2851.281972: funcgraph_entry: __next_timer_interrupt() {
    1.461 us
}

<idle>-0  4d..10  2851.282030: funcgraph_entry: get_next_timer_interrupt() {
    __next_timer_interrupt() {
        0.639 us
    }
}

<idle>-0  4d..20  2851.282031: funcgraph_entry: do_IRQ() {
    irq_handler_entry: irq=27 name=em1
    note_interrupt() {
        7.862 us
    }
}

<idle>-0  4dNh10  2851.282047: funcgraph_exit: get_next_timer_interrupt() {
    __next_timer_interrupt() {
        0.529 us
    }
}

<idle>-0  4dNh10  2851.282048: funcgraph_entry: get_next_timer_interrupt() {
    __next_timer_interrupt() {
        7.862 us
    }
}

<idle>-0  4dNh10  2851.282049: funcgraph_exit: get_next_timer_interrupt() {
    __next_timer_interrupt() {
        0.491 us
    }
}

<idle>-0  4d..10  2851.282070: funcgraph_entry: hrtimer_interrupt() {
    smp_apic_timer_interrupt() {
        1.526 us
    }
}

<idle>-0  4d..10  2851.282071: funcgraph_exit: smp_apic_timer_interrupt() {
    hrtimer_interrupt() {
        4.174 us
    }
}

<idle>-0  4dNh10  2851.282091: funcgraph_entry: smp_apic_timer_interrupt() {
    hrtimer_interrupt() {
        6.447 us
    }
}

<idle>-0  4dNh10  2851.282092: funcgraph_exit:
<table>
<thead>
<tr>
<th>Function</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_next_timer_interrupt()</td>
<td>0.594 us</td>
</tr>
<tr>
<td>__next_timer_interrupt()</td>
<td>2.260 us</td>
</tr>
<tr>
<td>smp_apic_timer_interrupt()</td>
<td>4.919 us</td>
</tr>
<tr>
<td>ksoftirq-45 get_next_timer_interrupt()</td>
<td>1.639 us</td>
</tr>
<tr>
<td>ksoftirq-45 hrtimer_interrupt()</td>
<td>1.461 us</td>
</tr>
<tr>
<td>ksoftirq-45 do_IRQ()</td>
<td>0.639 us</td>
</tr>
<tr>
<td>ksoftirq-45 note_interrupt()</td>
<td>7.862 us</td>
</tr>
<tr>
<td>ksoftirq-45 note_interrupt()</td>
<td>6.447 us</td>
</tr>
<tr>
<td>smp_apic_timer_interrupt()</td>
<td>4.174 us</td>
</tr>
<tr>
<td>smp_apic_timer_interrupt()</td>
<td>4.174 us</td>
</tr>
</tbody>
</table>

```c
get_next_timer_interrupt() {
    __next_timer_interrupt() {
    ...
}
}smp_apic_timer_interrupt() {
    hrtimer_interrupt() {
    ...
}
}
Latency from interrupts

What about this delay?
Interrupts are disabled

- Keeps interrupts from happening
Preemption disabled

- Not preempting the current task for other tasks
Preemption disabled

- Not preempting the current task for other tasks
  - Interrupts disabled (can’t notify to stop the task)
Preemption disabled

- Not preempting the current task for other tasks
  - Interrupts disabled (can’t notify to stop the task)
  - Spinning locks (Can’t be held by non running tasks)
Preemption disabled

• Not preempting the current task for other tasks
  - Interrupts disabled (can’t notify to stop the task)
  - Spinning locks (Can’t be held by non running tasks)
  - Accessing per CPU data (Locked on CPU keeps the data safe)
Preemption and interrupt disabled latency tracers

- irqsoff
- preemptoff
- preemptirqsoff
Preemption and interrupt disabled latency tracers

- irqsoff
- preemptoff
- preemptirqsoff

Note, these are usually not configured on production kernels
  - They cause noticeable overhead even when turned off
Preemption and interrupt disabled latency tracers

- irqsoff
- preemptoff
- preemptirqsoff

- Note, these are usually not configured on production kernels
  - They cause noticeable overhead even when turned off

- There’s also preempt and irq enable/disabling events
  - More on this later
Preemption and interrupt disabled latency tracers

- make menuconfig (Kernel Hacking -> Tracers menu)
Tracing Latency from Interrupts with PREEMPT_RT (5.4.14-rt7)

```plaintext
# trace-cmd start -p preemptirqsoff -O sym-offset -l '*_interrupt' -l do_IRQ -l '*spin_*'
# trace-cmd show

# tracer: preemptirqsoff
# preemptirqsoff latency trace v1.1.5 on 5.6.14-test-rt7+
# latency: 60 us, #25/25, CPU#5 | (M:preempt_rt VP:0, KP:0, SP:0 HP:0 #P:8)
# | task: rcuc/5-51 (uid:0 nice:0 policy:1 rt_prio:1)
# => started at: cpuidle_enter_state+0x89/0x4a0
# => ended at:     schedule+0x54/0x100
#  
#          _------=> CPU#
#          /   ------=> irqs-off
#          |         ------=> need-resched
#          ||          ------=> need-resched_lazy
#          |||          ------=> hardirq/softirq
#          ||||          ------=> preempt-depth
#          ||||||          ------=> migrate-disable
#          |||||||          delay
#          ||||||||          caller
#          \     /      |      |
#        <idle>-0 5d...1.. 1us : cpuidle_enter_state+0x89/0x4a0
#        <idle>-0 5d...1.. 3us : smp_apic_timer_interrupt+0x0/0x220 <-apic_timer_interrupt+0xf/0x20
#        <idle>-0 5d...1.. 5us : _raw_spin_lock+0x0/0x30 <-tick_do_update_jiffies64.part.0+0x15/0x1d0
#        <idle>-0 5d...1.. 6us : _raw_spin_lock_irqsave+0x0/0x60 <-timekeeping_advance+0x25/0x5f0
#        <idle>-0 5d...2.. 8us : raw_spin_unlock_irqrestore+0x0/0x80 <-timekeeping_advance+0x3b9/0x5f0
#        <idle>-0 5d..h1.. 9us : hrtimer_interrupt+0x0/0x240 <-smp_apic_timer_interrupt+0xa1/0x220
```
Tracing Latency from Interrupts with PREEMPT_RT (5.4.14-rt7)

```bash
# trace-cmd start -p preemptirqsoff -O sym-offset -l '*_interrupt' -l do_IRQ -l '*spin_*'
# trace-cmd show

# tracer: preemptirqsoff
# preemptirqsoff latency trace v1.1.5 on 5.6.14-test-rt7+
# latency: 60 us, #25/25, CPU#5 | (M:preempt_rt VP:0, KP:0, SP:0 HP:0 #P:8)
# | task: rcuc/5-51 (uid:0 nice:0 policy:1 rt_prio:1)
# => started at: cpuidle_enter_state+0x89/0x4a0
# => ended at: schedule+0x54/0x100

<idle>-0 5d...1.. 1us : cpuidle_enter_state+0x89/0x4a0
<idle>-0 5d...1.. 3us : smp_apic_timer_interrupt+0x0/0x220 <-apic_timer_interrupt+0xf/0x20
<idle>-0 5d...1.. 5us : _raw_spin_lock+0x0/0x30 <-tick_do_update_jiffies64.part.0+0x15/0x1d0
<idle>-0 5d...1.. 6us : _raw_spin_lock_irqsave+0x0/0x60 <-timekeeping_advance+0x25/0x5f0
<idle>-0 5d...2.. 8us : raw_spin_unlock_irqrestore+0x0/0x80 <-timekeeping_advance+0x3b9/0x5f0
<idle>-0 5d..h1.. 9us : hrtimer_interrupt+0x0/0x240 <-smp_apic_timer_interrupt+0xa1/0x220
```
Tracing Latency from Interrupts with PREEMPT_RT (5.4.14-rt7)

```bash
# trace-cmd start -p preemptirqsoff -O sym-offset -l '*_interrupt' -l do_IRQ -l '*spin_*'
# trace-cmd show
```

```
# tracer: preemptirqsoff
# preemptirqsoff latency trace v1.1.5 on 5.6.14-test-rt7+
# latency: 60 us, #25/25, CPU#5 | (M:preempt_rt VP:0, KP:0, SP:0 HP:0 #P:8)
#    -----------------
#    | task: rcuc/5-51 (uid:0 nice:0 policy:1 rt_prio:1)
#    -----------------
# => started at: cpuidle Enter State+0x89/0x4a0
# => ended at:   schedule+0x54/0x100
#    
#    _------=> CPU#
#    / _-----=> irqs-off
#    | / _----=> need-resched
#    || / _----=> need-resched Lazy
#    ||| / _---=> hardirq/softirq
#    |||| / _--=> preempt-depth
#    |||||| / _-=> migrate-disable
#    ||||||| /     delay
#    cmd  pid  | time  |   caller
#    \   /      | |      |
# <idle>-0       5d...1..    1us : cpuidle Enter State+0x89/0x4a0
# <idle>-0       5d...1..    3us : smp_apic Timer Interrupt+0x0/0x220 <-apic_timer_interrupt+0xf/0x20
# <idle>-0       5d...1..    5us : _raw_spin_lock+0x0/0x30 <-tick do update jiffies64.part_0+0x15/0x1d0
# <idle>-0       5d...1..    6us : _raw_spin_lock_irqsave+0x0/0x60 <-timekeeping Advance+0x25/0x5f0
# <idle>-0       5d...2..    8us : raw spin unlock IRQ restore+0x0/0x80 <-timekeeping Advance+0x3b9/0x5f0
# <idle>-0       5d...h1..    9us : hrtimer Interrupt+0x0/0x240 <-smp_apic Timer Interrupt+0xa1/0x220
```
Tracing Latency from Interrupts with PREEMPT_RT (5.4.14-rt7)

```bash
# trace-cmd start -p preemptirqsoff -O sym-offset -l '*_interrupt' -l do_IRQ -l '*spin_*' # trace-cmd show
```

```bash
# tracer: preemptirqsoff
# preemptirqsoff latency trace v1.1.5 on 5.6.14-test-rt7+
# -----------------------------
# latency: 60 us, #25/25, CPU#5 | (M:preempt_rt VP:0, KP:0, SP:0 HP:0 #P:8)
# -----------------------------
# | task: rcuc/5-51 (uid:0 nice:0 policy:1 rt_prio:1)
# -----------------------------
# => started at: cpuidle_enter_state+0x89/0x4a0
# => ended at: schedule+0x54/0x100
```

```
<table>
<thead>
<tr>
<th>cmd</th>
<th>pid</th>
<th>time</th>
<th>caller</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;idle&gt;-0</td>
<td>5d...1.. 1us : cpuidle_enter_state+0x89/0x4a0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;idle&gt;-0</td>
<td>5d...1.. 3us : smp_apic_timer_interrupt+0x0/0x220 &lt;-apic_timer_interrupt+0xf/0x20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;idle&gt;-0</td>
<td>5d...1.. 5us : _raw_spin_lock+0x0/0x30 &lt;-tick_do_update_jiffies64.part.0+0x15/0x1d0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;idle&gt;-0</td>
<td>5d...1.. 6us : _raw_spin_lock_irqsave+0x0/0x60 &lt;-timekeeping_advance+0x25/0x5f0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;idle&gt;-0</td>
<td>5d...2.. 8us : raw_spin_unlock_irqrestore+0x0/0x80 &lt;-timekeeping_advance+0x3b9/0x5f0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;idle&gt;-0</td>
<td>5d..h1.. 9us : hrtimer_interrupt+0x0/0x240 &lt;-smp_apic_timer_interrupt+0xa1/0x220</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
Tracing Latency from Interrupts with PREEMPT_RT (5.4.14-rt7)

```
<idle>-0  5d..h1..  9us : _raw_spin_lock_irqsave+0x0/0x60 <-hrtimer_interrupt+0x6c/0x240
<idle>-0  5d..h2.. 10us : _raw_spin_unlock irqrestore+0x0/0x80 <-__hrtimer_run_queues+0x116/0x3a0
<idle>-0  5d..h2.. 13us+: _raw_spin_lock_irqsave+0x0/0x60 <-try_to_wake_up+0x34/0x7f0
<idle>-0  5d..h3..  23us : _raw_spin_lock+0x0/0x30 <-try_to_wake_up+0x1d7/0x7f0
<idle>-0  5d.h4..  26us : _raw_spin_lock+0x0/0x30 <-enqueue_task rt+0x189/0x360
<idle>-0  5dN.h3.. 29us : _raw_spin_unlock_irqrestore+0x0/0x80 <-try_to_wake_up+0x24c/0x7f0
<idle>-0  5dN.h1..  30us : _raw_spin_lock+0x0/0x30 <-scheduler_tick+0x39/0x130
<idle>-0  5dN.h1..  33us : _raw_spin_lock_irq+0x0/0x40 <-__hrtimer_run_queues+0x140/0x3a0
<idle>-0  5dN.h2..  34us : _raw_spin_unlock_irqrestore+0x0/0x80 <-hrtimer_interrupt+0x13f/0x240
<idle>-0  5dN..2..  35us : _raw_spin_lock_irqsave+0x0/0x60 <-try_to_wake_up+0x34/0x7f0
<idle>-0  5dN..3..  35us : _raw_spin_lock+0x0/0x30 <-enqueue_task rt+0x189/0x360
<idle>-0  5dN..3..  40us : _raw_spin_unlock_irqrestore+0x0/0x80 <-try_to_wake_up+0x24c/0x7f0
<idle>-0  5dN..1..  43us : _raw_spin_lock_irqsave+0x0/0x60 <-lock_hrtimer_base+0x25/0x50
<idle>-0  5dN..2..  45us : _raw_spin_unlock_irqrestore+0x0/0x80 <-hrtimer_start_range ns+0x218/0x3b0
<idle>-0  5dN..1..  46us+: _raw_spin_lock+0x0/0x30 <-__schedule+0x95/0x890
rcuc/5-51  5d..2..  59us : _raw_spin_unlock irq+0x0/0x60 <-finish_task_switch+0xa0/0x2f0
rcuc/5-51  5..1..  60us : schedule+0x54/0x100 <-schedule+0x54/0x100
rcuc/5-51  5..1..  61us : tracer_preempt_on+0xee/0x100 <-schedule+0x54/0x100
rcuc/5-51  5..1..  70us : <stack trace>

=> smplboot_thread_fn+0xf2/0x2c0
=> kthread+0xf9/0x130
=> ret_from_fork+0x3a/0x50
=> 0
=> 0x8316998000000000
=> 0x8316a571ffffffff
=> 0x1d28ffffffff
=> 0x1250001
=> 0
=> rcu_preempt_need_deferred_qs+0x0/0x40
=> rcu_preempt_deferred_qs+0x23/0x80
```
The Scheduling Latency Tracer

- Does not have the overhead when not enabled
The Scheduling Latency Tracer

- Does not have the overhead when not enabled
  - OK to keep configured in production systems
The Scheduling Latency Tracer

- Does not have the overhead when not enabled
  - OK to keep configured in production systems

- The types of scheduling tracers
  - wakup - trace the highest priority task (any task)
  - wakeup_rt - trace the highest priority RT task
  - wakeup_dl - trace the highest priority deadline task
wakeup_rt with PREEMPT_RT (5.4.14-rt7)

```
# trace-cmd start -p wakeup_rt -O sym-offset -l '*_interrupt' -l do_IRQ -l '*spin*' -e sched_switch -e sched_waking
# trace-cmd show

# tracer: wakeup_rt
#
# wakeup_rt latency trace v1.1.5 on 5.6.14-test-rt7+
# --------------------------------------------------------------------
# latency: 43 us, #15/15, CPU#3 | (M:preempt_rt VP:0, KP:0, SP:0 HP:0 #P:8)
#                 ----------------
#                 | task: irq/27-em1-1479 (uid:0 nice:0 policy:1 rt_prio:50)
#                 ----------------
#                 __________=> CPU#
#                 / __________=> irqs-off
#                 | / __________=> need-resched
#                 || / __________=> need-resched_lazy
#                 ||| / __________=> hardirq/softirq
#                 |||| / __________=> preempt-depth
#                 |||||| / __________=> migrate-disable
#                 ||||||| / delay
cmd   pid     time    caller
\   /      \        /      
<idle>-0       3dN.h5..    0us :      0:120:R   + [003] 1479: 49:R irq/27-em1
<idle>-0       3dN.h5..    9us : <stack trace>
=> __ftrace_trace_stack+0x190/0x1d0
=> probe_wakeup+0x280/0x320
=> ttwu_do_wakeup+0x141/0x1a0
=> try_to_wake_up+0x201/0x7f0
=> __handle_irq_event_percpu+0x9a/0x240
=> handle_irq_event_percpu+0x45/0x80
=> handle_irq_event+0x52/0x90
```
wakeup_rt with PREEMPT_RT (5.4.14-rt7)

# trace-cmd start -p wakeup_rt -O sym-offset -l '*_interrupt' -l do_IRQ \ 
-1 '*spin_*' -e sched_switch -e sched_waking
# trace-cmd show

# tracer: wakeup_rt
#
# wakeup_rt latency trace v1.1.5 on 5.6.14-test-rt7+
# --------------------------------------------------------------------
# latency: 43 us, #15/15, CPU#3 | (M:preempt_rt VP:0, KP:0, SP:0 HP:0 #P:8)
#   -----------------   | task: irq/27-em1-1479 (uid:0 nice:0 policy:1 rt_prio:50)
#   -----------------   
#   
# _-------=> CPU#
# / _-------=> irqs-off
# | / _------=> need-resched
# || / _------=> need-resched_lazy
# ||| / _------=> hardirq/softirq
# |||| / _----=> preempt-depth
# |||||| / _---=> migrate-disable
# ||||||| / delay
cmd   pid   time  |   caller
\   /      \   |   /
<idle>-0    3dN.h5..  0us : 0:120:R   + [003] 1479: 49:R irq/27-em1
<idle>-0    3dN.h5..  9us : <stack trace>

=> __ftrace_trace_stack+0x190/0x1d0
=> probe_wakeup+0x28b/0x320
=> ttwu_do_wakeup+0x141/0x1a0
=> try_to_wake_up+0x201/0x7f0
=> __handle_irq_event_percpu+0x9a/0x240
=> handle_irq_event_percpu+0x45/0x80
=> handle_irq_event+0x52/0x90
wakeup_rt with PREEMPT_RT (5.4.14-rt7)

```shell
# trace-cmd start -p wakeup_rt -O sym-offset -l '*_interrupt' -l do_IRQ \
   -l '*spin_*' -e sched_switch -e sched_waking
# trace-cmd show

# tracer: wakeup_rt
#
# wakeup_rt latency trace v1.1.5 on 5.6.14-test-rt7+
# *--------------------------------------------------------------------
# latency: 43 us, #15/15, CPU#3 | (M:preempt_rt VP:0, KP:0, SP:0 HP:0 #P:8)
# *--------------------------------------------------------------------
# | task: irq/27-em1-1479 (uid:0 nice:0 policy:1 rt_prio:50)
# *--------------------------------------------------------------------
#
# __________=> CPU#
#  / __________=> irqs-off
#  | / __________=> need-resched
#  || / __________=> need-resched_lazy
#  ||| / __________=> hardirq/softirq
#  |||| / __________=> preempt-depth
#  |||||| / __________=> migrate-disable
#  ||||||| / delay
cmd  pid  time  caller
\  /  
<idle>-0 3dN.h5.. 0us : 0:120:R + [003] 1479: 49:R irq/27-em1
<idle>-0 3dN.h5.. 9us : <stack trace>
```

```c
=> __ftrace_trace_stack+0x190/0x1d0
=> probe_wakeup+0x28b/0x320
=> ttwu_do_wakeup+0x141/0x1a0
=> try_to_wake_up+0x201/0x7f0
=> __handle_irq_event_percpu+0x9a/0x240
=> handle_irq_event_percpu+0x45/0x80
=> handle_irq_event+0x52/0x90
```
wakeup_rt with PREEMPT_RT (5.4.14-rt7)

<idle>-0  3dN.h5..  10us+: 0
<idle>-0  3dN.h3..  21us : __raw_spin_unlock_irqrestore+0x0/0x80 <-try_to_wake_up+0x24c/0x7f0
<idle>-0  3dN.h1..  21us : note_interrupt+0x0/0x206 <-handle_irq_event_percpu+0x66a/0x80
<idle>-0  3dN.h1..  21us : __raw_spin_lock+0x0/0x30 <-handle_irq_event+0x5d/0x90
<idle>-0  3dN..1..  23us+: __raw_spin_lock_irqsave+0x0/0x60 <-lock_hrtimer_base+0x25/0x50
<idle>-0  3dN..2..  35us : __raw_spin_unlock_irqrestore+0x0/0x80 <-hrtimer_try_to_cancel+0x5f/0x140
<idle>-0  3dN..1..  35us : __raw_spin_lock_irqsave+0x0/0x60 <-lock_hrtimer_base+0x25/0x50
<idle>-0  3dN..2..  37us : __raw_spin_unlock_irqrestore+0x0/0x80 <-hrtimer_start_range_ns+0x218/0x3b0
<idle>-0  3dN..1..  38us : __raw_spin_lock+0x0/0x30 <-__schedule+0x95/0x890
<idle>-0  3d...2..  40us : sched_switch: prev_comm=swapper/3 prev_pid=0 prev_prio=120 prev_state=R ==> next_comm=irq/27-em1 next_pid=1479 next_prio=49
<idle>-0  3d...3..  41us : __schedule+0x69a/0x890
<idle>-0  3d...3..  41us : 0:120:R ==> [003] 1479: 49:R irq/27-em1
<idle>-0  3d...3..  44us : <stack trace>

=> __ftrace_trace_stack+0x190/0x1d0
=> probe_wakeup_sched_switch+0x2a0/0x2e1
=> __schedule+0x69a/0x890
=> schedule_idle+0x28/0x40
=> do_idle+0x1aa/0x310
=> cpu_startup_entry+0x19/0x20
=> start_secondary+0x150/0x190
=> secondary_startup_64+0xb6/0xc0
=> 0x10300000025
=> 0x2400000025
=> 0x62007800000003
=> 0xcc000017e60
=> 0x2503010004
=> 0xf000000103
=> __ftrace_trace_stack+0x190/0x1d0
=> probe_wakeup_sched_switch+0x2a0/0x2e1
Issues with the Latency Tracers
Issues with the Latency Tracers

• Rigid (not very flexible)
Issues with the Latency Tracers

• Rigid (not very flexible)
• Always the highest priority task (or all tasks)
  – Can’t look at a specific task
Issues with the Latency Tracers

• Rigid (not very flexible)
• Always the highest priority task (or all tasks)
  - Can’t look at a specific task
• Always the max latency
Issues with the Latency Tracers

- Rigid (not very flexible)
- Always the highest priority task (or all tasks)
  - Can’t look at a specific task
- Always the max latency
- Specific to irqs or preemption disabled or wake up latency
Histogram Triggers and Synthetic Events!
Histogram Triggers and Synthetic Events!

• Choose your own events
  – This is where those preempt and irqs enabling and disabling events come in handy
Histogram Triggers and Synthetic Events!

• Choose your own events
  - This is where those preempt and irqs enabling and disabling events come in handy

• Add filters (specific for a task or other event field)
Histogram Triggers and Synthetic Events!

- Choose your own events
  - This is where those preempt and irqs enabling and disabling events come in handy
- Add filters (specific for a task or other event field)
- Create a nice histogram of latency timings
Creating a synthetic event

# mount -t tracefs nodev /sys/kernel/tracing
# echo 'irq_lat pid_t pid u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd list -e synthetic

synthetic:irq_lat
# Creating a synthetic event

```
# mount -t tracefs nodev /sys/kernel/tracing
# echo 'irq_lat pid_t pid u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd list -e synthetic

synthetic:irq_lat
```
Creating a synthetic event

```
# mount -t tracefs nodev /sys/kernel/tracing
# echo 'irq_lat pid_t pid u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd list -e synthetic

synthetic:irq_lat
```
Creating a synthetic event

```
# mount -t tracefs nodev /sys/kernel/tracing
# echo 'irq_lat pid_t pid u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd list -e synthetic

synthetic:irq_lat
```
Making the irq disabled histogram

```
# trace-cmd start \
   -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \
   -e irq_enable -R 'hist:keys=cpu:'
   'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:'
   'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)' \   
   -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```
# trace-cmd start \
  -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \
  -e irq_enable -R 'hist:keys=cpu:'
  'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:'
  'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)' \
  -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```bash
# trace-cmd start \
  -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \n  -e irq_enable -R 'hist:keys=cpu:' \n  'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:' \n  'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)' \n  -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```bash
# trace-cmd start \
  -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \
  -e irq_enable -R 'hist:keys=cpu:'
  'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:'
  'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)'
  -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```
# trace-cmd start \
  -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \
  -e irq_enable -R 'hist:keys=cpu:'
  'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:'
  'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)' \
  -e irq_lat -R 'hist:keys=lat:sort=lat'
```
# trace-cmd start \
  -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \
  -e irq_enable -R 'hist:keys=cpu:'
  'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:ovich' \
  'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)' \
  -e irq_lat -R 'hist:keys=lat:sort=lat'
Making the irq disabled histogram

```
# trace-cmd start \
   -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \
   -e irq_enable -R 'hist:keys=cpu:'
'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:'
'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)' \
   -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```bash
# trace-cmd start \
  -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \
  -e irq_enable -R 'hist:keys=cpu:'
  'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:'
  'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)' \
  -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```bash
# trace-cmd start \
  -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \n  -e irq_enable -R 'hist:keys=cpu:'
  'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:,'
  'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)' \n  -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```
# cat /sys/kernel/tracing/events/synthetic_events/irq_lat/hist

# event histogram
#
# trigger info: hist:keys=lat:vals=hitcount:sort=lat:size=2048 [active]
#
{ lat:          0 } hitcount:       8150
{ lat:          1 } hitcount:      55870
{ lat:          2 } hitcount:       5378
{ lat:          3 } hitcount:       2219
{ lat:          4 } hitcount:        781
{ lat:          5 } hitcount:       6519
{ lat:          6 } hitcount:       1967
{ lat:          7 } hitcount:       263
{ lat:          8 } hitcount:        170
{ lat:          9 } hitcount:        136
{ lat:         10 } hitcount:          91
{ lat:         11 } hitcount:          47
{ lat:         12 } hitcount:          26
{ lat:         13 } hitcount:           9
{ lat:         14 } hitcount:           14
{ lat:         15 } hitcount:           9
{ lat:         16 } hitcount:            8
{ lat:         17 } hitcount:            3
{ lat:         18 } hitcount:            4
{ lat:         19 } hitcount:            3
{ lat:         21 } hitcount:            1

Totals:
Hits: 81673
Entries: 21
Dropped: 0
```
Creating a wake up latency synthetic event

# mount -t tracefs nodev /sys/kernel/tracing
# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd list -e synthetic

synthetic:wakeup_lat
Creating a wake up latency synthetic event

```
# mount -t tracefs nodev /sys/kernel/tracing
# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd list -e synthetic

synthetic:wakeup_lat
```
Creating a wake up latency synthetic event

```
# mount -t tracefs nodev /sys/kernel/tracing
# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd list -e synthetic

synthetic:wakeup_lat
```
Making the irq disabled histogram

```bash
# trace-cmd start \
  -e sched_waking -R 'hist:keys=pid:ts1=common_timestamp.usecs if prio < 100' \
  -e sched_switch -R 'hist:keys=next_pid:' \n  'pid=common_pid, lat=common_timestamp.usecs-$ts1:' \n  'onmatch(sched.sched_waking).trace(wakeup_lat,$pid,next_prio,$lat)' \
  -e wakeup_lat -R 'hist:keys=prio, lat:sort=prio, lat'
```
Making the wakeup latency histogram

```
# cat /sys/kernel/tracing/events/synthetic_events/wakeup_lat/hist

# event histogram
#
# trigger info: hist:keys=prio, lat:vals=hitcount:sort=prio, lat:size=2048 [active]
#
{
  prios: [0, 1],
  lat: [7]
} hitcount: 2
{
  prios: [49, 50],
  lat: [15]
} hitcount: 9
{
  prios: [49, 50],
  lat: [16]
} hitcount: 4
{
  prios: [49, 50],
  lat: [17]
} hitcount: 4
{
  prios: [49, 50],
  lat: [18]
} hitcount: 1
{
  prios: [98, 99],
  lat: [3]
} hitcount: 5
{
  prios: [98, 99],
  lat: [4]
} hitcount: 2
{
  prios: [98, 99],
  lat: [5]
} hitcount: 2
{
  prios: [98, 99],
  lat: [6]
} hitcount: 5
{
  prios: [98, 99],
  lat: [12]
} hitcount: 1
{
  prios: [98, 99],
  lat: [14]
} hitcount: 1
{
  prios: [98, 99],
  lat: [15]
} hitcount: 3
{
  prios: [98, 99],
  lat: [16]
} hitcount: 2
{
  prios: [98, 99],
  lat: [17]
} hitcount: 1
{
  prios: [98, 99],
  lat: [18]
} hitcount: 2
{
  prios: [98, 99],
  lat: [19]
} hitcount: 36
{
  prios: [98, 99],
  lat: [20]
} hitcount: 23
{
  prios: [98, 99],
  lat: [21]
} hitcount: 7
{
  prios: [98, 99],
  lat: [22]
} hitcount: 4
{
  prios: [98, 99],
  lat: [23]
} hitcount: 6
{
  prios: [98, 99],
  lat: [24]
} hitcount: 1
{
  prios: [98, 99],
  lat: [25]
} hitcount: 2
{
  prios: [98, 99],
  lat: [26]
} hitcount: 1
{
  prios: [98, 99],
  lat: [27]
} hitcount: 2

Totals:
  Hits: 135
  Entries: 24
  Dropped: 0
```
Histogram Triggers and Synthetic Events

• Powerful! Can be enabled on production systems!
Histogram Triggers and Synthetic Events

- Powerful! Can be enabled on production systems!
- `<sarcasm>Easy to use</sarcasm>`
Histogram Triggers and Synthetic Events

- Powerful! Can be enabled on production systems!
- `<sarcasm>`Easy to use`/sarcasm`
  - You understood all I talked about, right?
Histogram Triggers and Synthetic Events

- Powerful! Can be enabled on production systems!
- `<sarcasm>Easy to use</sarcasm>`
  - You understood all I talked about, right?
- It has a rather strange format
  - Takes a while to get use to
Histogram Triggers and Synthetic Events

- Powerful! Can be enabled on production systems!
- `<sarcasm>`Easy to use`</sarcasm>`
  - You understood all I talked about, right?
- It has a rather strange format
  - Takes a while to get use to
- Not many users
  - I know, because I found unreported bugs while using it
Histogram Triggers and Synthetic Events

• Powerful! Can be enabled on production systems!

• <sarcasm>Easy to use</sarcasm>
  – You understood all I talked about, right?

• It has a rather strange format
  – Takes a while to get use to

• Not many users
  – I know, because I found unreported bugs while using it
  – If it is hard to use, people wont use it
Histogram Triggers and Synthetic Events

• Need a language that is well known
  – Something people don’t need to “re-learn”
Welcome
to
the
Vaporware!
Well really
Almostware!
Histogram Triggers and Synthetic Events

• Need a language that is well known
  – Something people don’t need to “re-learn”
Histogram Triggers and Synthetic Events

- Need a language that is well known
  - Something people don’t need to “re-learn”

SQL!
Histogram Triggers and Synthetic Events

• Need a language that is well known
  – Something people don’t need to “re-learn”

• Think about it..
Histogram Triggers and Synthetic Events

• Need a language that is well known
  - Something people don’t need to “re-learn”

• Think about it..
  - Events are like tables
Histogram Triggers and Synthetic Events

• Need a language that is well known
  – Something people don’t need to “re-learn”

• Think about it..
  – Events are like tables
  – Each field of an event is a column
Histogram Triggers and Synthetic Events

• Need a language that is well known
  - Something people don’t need to “re-learn”

• Think about it..
  - Events are like tables
  - Each field of an event is a column
  - Each instance of the event is a row
Histogram Triggers and Synthetic Events

• Need a language that is well known
  - Something people don’t need to “re-learn”

• Think about it..
  - Events are like tables
  - Each field of an event is a column
  - Each instance of the event is a row

• We can join tables
Histogram Triggers and Synthetic Events

• Need a language that is well known
  - Something people don’t need to “re-learn”

• Think about it..
  - Events are like tables
  - Each field of an event is a column
  - Each instance of the event is a row

• We can join tables
  - Why not join events?
Making the irq disabled histogram

```bash
# echo 'irq_lat pid_t pid u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \
  -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \n  -e irq_enable -R 'hist:keys=cpu:' \n  'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:'\n  'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)' \n    -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```bash
# trace-cmd start \\
--sql 'select start.common_pid as pid,
    (end.common_timestamp.usecs - start.common_timestamp.usecs) as irq_lat
    from irq_disable as start
    join irq_enable as end
    on start.common_pid = end.common_pid
    where pid > 0) as irq_lat' \\
-e irq_disable -e irq_enable \\
-e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```bash
# trace-cmd start \
   --sql '{select start.common_pid as pid, 
             (end.common_timestamp.usecs - start.common_timestamp.usecs) as irq_lat 
             from irq_disable as start 
             join irq_enable as end 
             on start.common_pid = end.common_pid 
             where pid > 0) as irq_lat' \
   -e irq_disable -e irq_enable \
   -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```
# trace-cmd start \
   --sql '(select start.common_pid as pid, 
           (end.common_timestamp.usecs - start.common_timestamp.usecs) as irq_lat 
           from irq_disable as start 
           join irq_enable as end 
           on start.common_pid = end.common_pid 
           where pid > 0) as irq_lat' \ 
   -e irq_disable -e irq_enable \ 
   -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```bash
# trace-cmd start
   --sql '(
      select start.common_pid as pid,
      (end.common_timestamp.usecs - start.common_timestamp.usecs) as irq_lat
      from irq_disable as start
      join irq_enable as end
      on start.common_pid = end.common_pid
      where pid > 0) as irq_lat'
   -e irq_disable -e irq_enable
   -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```bash
# trace-cmd start \
   --sql '
   (select start.common_pid as pid,
    (end.common_timestamp.usecs - start.common_timestamp.usecs) as irq_lat
    from irq_disable as start
    join irq_enable as end
    on start.common_pid = end.common_pid
    where pid > 0) as irq_lat'
   -e irq_disable -e irq_enable \
   -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```
# trace-cmd start \
  --sql '(
    select start.common_pid as pid,
    (end.common_timestamp.usecs - start.common_timestamp.usecs) as irq_lat
    from irq_disable as start
    join irq_enable as end
    on start.common_pid = end.common_pid
    where pid > 0) as irq_lat' \
  -e irq_disable -e irq_enable \
  -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```bash
# trace-cmd start \
  --sql '(
    select start.common_pid as pid,
    (end.common_timestamp.usecs - start.common_timestamp.usecs) as irq_lat
    from irq_disable as start
    join irq_enable as end
    on start.common_pid = end.common_pid
    where pid > 0) as irq_lat' \n-e irq_disable -e irq_enable \n-e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```bash
# trace-cmd start \
  --sql '(
select start.common_pid as pid,
  (end.common_timestamp.usecs - start.common_timestamp.usecs) as irq_lat
from irq_disable as start
join irq_enable as end
on start.common_pid = end.common_pid
where pid > 0) as irq_lat' \
-e irq_disable -e irq_enable \
-e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the irq disabled histogram

```
# echo 'irq_lat pid_t pid u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \n    -e irq_disable -R 'hist:keys=cpu:ts0=common_timestamp.usecs if common_pid > 0' \n    -e irq_enable -R 'hist:keys=cpu:'\n    'pid=common_pid,irq_lat=common_timestamp.usecs-$ts0:'\n    'onmatch(preemptirq.irq_disable).trace(irq_lat,$pid,$irq_lat)'
    -e irq_lat -R 'hist:keys=lat:sort=lat'

# trace-cmd start \n    --sql '(select start.common_pid as pid, 
          (end.common_timestamp.usecs - start.common_timestamp.usecs) as irq_lat
          from irq_disable as start 
          join irq_enable as end 
          on start.common_pid = end.common_pid 
          where pid > 0) as irq_lat' \n    -e irq_disable -e irq_enable \n    -e irq_lat -R 'hist:keys=lat:sort=lat'
```
Making the wake up latency histogram

```bash
# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \
  -e sched_waking -R 'hist:keys=pid:ts1=common_timestamp.usecs if prio < 100' \
  -e sched_switch-R 'hist:keys=next_prio:'\n  'pid=common_pid,lat=common_timestamp.usecs-$ts1:'\n  'onmatch(sched.sched_waking).trace(wakeup_lat,$pid,next_prio,$lat)' \
  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'
```
Making the wake up latency histogram

```
# trace-cmd start \
   --sql '
   (select start.common_pid as pid, end.next_prio as prio,
   (end.common_timestamp.usecs - start.common_timestamp.usecs) as lat
   from irq_disable as start
   join irq_enable as end
   on start.common_pid = end.common_pid
   where end.prio < 100) as wakeup_lat'
   -e sched_waking -e sched_switch \
   -e wakeup_lat -R 'hist:keys=prio, lat:sort=prio, lat'
```
Making the wake up latency histogram

# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \
   -e sched_waking -R 'hist:keys=pid:ts1=common_timestamp.usecs if prio < 100' \n   -e sched_switch -R 'hist:keys=next_pid:lat=common_timestamp.usecs-$ts1:' \n   'onmatch(sched.sched_waking).trace(wakeup_lat, next_pid, next_prio, $irq_lat)' \n   -e wakeup_lat -R 'hist:keys=prio, lat:sort=prio, lat'

# trace-cmd start \
   --sql 'select end.next_pid as pid, end.next_prio as prio 
      (end.common_timestamp.usecs - start.common_timestamp.usecs) as lat 
      from sched_waking as start 
      join sched_switch as end 
      on start.pid = end.next_pid 
      where start.prio < 100) as wakeup_lat' \
   -e sched_waking -e sched_switch \
   -e wakeup_lat -R 'hist:keys=prio, lat:sort=prio, lat'
Making the wake up latency histogram

```
# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \
  -e sched_waking -R 'hist:keys=pid:ts1=common_timestamp.usecs if prio < 100' \
  -e sched_switch -R 'hist:keys=next_pid:lat=common_timestamp.usecs-$ts1:' \n  ‘onmatch(sched.sched_waking).trace(wakeup_lat,next_pid,next_prio,$irq_lat)' \n  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'

# trace-cmd start \
  --sql 'select end.next_pid as pid, end.next_prio as prio,    
    (end.common_timestamp.usecs - start.common_timestamp.usecs) as lat    
  from sched_waking as start    
  join schedSwitch as end    
  on start.pid = end.next_pid    
  where start.prio < 100) as wakeup_lat' \n  -e sched_waking -e sched_switch \n  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'
```
Making the wake up latency histogram

```bash
# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \n  -e sched_waking -R 'hist:keys=pid:ts1=common_timestamp.usecs if prio < 100' \n  -e sched_switch -R 'hist:keys=next_pid:lat=common_timestamp.usecs-$ts1:' \n  'onmatch(sched.sched_waking).trace(wakeup_lat, next_pid, next_prio, $irq_lat)' \n  -e wakeup_lat -R 'hist:keys=prio, lat:sort=prio, lat'

# trace-cmd start \n  --sql '(select end.next_pid as pid, end.next_prio as prio, 
       (end.common_timestamp.usecs - start.common_timestamp.usecs) as lat
       from sched_waking as start 
       join sched_switch as end 
       on start.pid = end.next_pid 
       where start.prio < 100) as wakeup_lat'
  -e sched_waking -e sched_switch \n  -e wakeup_lat -R 'hist:keys=prio, lat:sort=prio, lat'
```
Making the wake up latency histogram

```
# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \
  -e sched_waking -R 'hist:keys=pid:ts1=common_timestamp.usecs if prio < 100' \
  -e sched_switch -R 'hist:keys=next_pid:lat=common_timestamp.usecs-$ts1:' \n  'onmatch(sched.sched_waking).trace(wakeup_lat,next_pid,next_prio,$irq_lat)' \n  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'

# trace-cmd start \
  --sql '(select end.next_pid as pid, end.next_prio as prio, 
          (end.common_timestamp.usecs - start.common_timestamp.usecs) as lat
          from sched_waking as start
          join sched_switch as end
          on start.pid = end.next_pid 
          where start.prio < 100) as wakeup_lat' \
  -e sched_waking -e sched_switch \
  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'
```
Making the wake up latency histogram

```bash
# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \
  -e sched_waking -R 'hist:keys=pid:ts1=common_timestamp.usecs if prio < 100' \n  -e sched_switch -R 'hist:keys=next_pid:lat=common_timestamp.usecs-$ts1:' \n  'onmatch(sched.sched_waking).trace(wakeup_lat,next_pid,next_prio,$irq_lat)' \n  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'

# trace-cmd start \
  --sql '(select end.next_pid as pid, end.next_prio as prio, 
          (end.common_timestamp.usecs - start.common_timestamp.usecs) as lat 
          from sched_waking as start 
          join sched_switch as end 
          on start.pid = end.next_pid 
          where start.prio < 100) as wakeup_lat' \n  -e sched_waking -e sched_switch \n  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'
```
Making the wake up latency histogram

# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \
  -e sched_waking -R 'hist:keys=pid:ts1=common_timestamp.usecs if prio < 100' \
  -e sched_switch -R 'hist:keys=next_pid:lat=common_timestamp.usecs-$ts1:' \
  'onmatch(sched.sched_waking).trace(wakeup_lat,next_pid,next_prio,$irq_lat)' \
  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'

# trace-cmd start \
  --sql '(select end.next_pid as pid, end.next_prio as prio, (end.common_timestamp.usecs - start.common_timestamp.usecs) as lat from sched_waking as start join sched_switch as end on start.pid = end.next_pid where start.prio < 100) as wakeup_lat' \
  -e sched_waking -e sched_switch \
  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'
Making the wake up latency histogram

```bash
# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \
  -e sched_waking -R 'hist:keys=pid:ts1=common_timestamp.usecs if prio < 100' \n  -e sched_switch -R 'hist:keys=next_pid:lat=common_timestamp.usecs-$ts1:' \n  "onmatch(sched.sched_waking).trace(wakeup_lat,next_pid,next_prio,$irq_lat)' \n  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'

# trace-cmd start \
  --sql "(select end.next_pid as pid, end.next_prio as prio, \n   (end.common_timestamp.usecs - start.common_timestamp.usecs) as lat \n   from sched_waking as start \n   join sched_switch as end \n   on start.pid = end.next_pid \n   where start.prio < 100) as wakeup_lat' \
  -e sched_waking -e sched_switch  \
  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'
```
Making the wake up latency histogram

```bash
# echo 'wakeup_lat pid_t pid int prio u64 lat' > /sys/kernel/tracing/synthetic_events
# trace-cmd start \
  -e sched_waking -R 'hist:keys=pid:ts1=common_timestamp.usecs if prio < 100' \ 
  -e sched_switch -R 'hist:keys=next_pid:lat=common_timestamp.usecs-ts1:' \ 
  'onmatch(sched.sched_waking).trace(wakeup_lat,next_pid,next_prio,$irq_lat)' \ 
  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'

# trace-cmd start \
  --sql '(select end.next_pid as pid, end.next_prio as prio, \ 
         (end.common_timestamp.usecs - start.common_timestamp.usecs) as lat \ 
         from sched_waking as start \ 
         join sched_switch as end \ 
         on start.pid = end.next_pid \ 
         where start.prio < 100) as wakeup_lat' \ 
  -e sched_waking -e sched_switch \ 
  -e wakeup_lat -R 'hist:keys=prio,lat:sort=prio,lat'
```
trace-cmd --sql : Coming soon!

• Have it mostly working
  – The “WHERE” clause is not working yet
trace-cmd --sql : Coming soon!

- Have it mostly working
  - The “WHERE” clause is not working yet

https://github.com/rostedt/sqlhist

```bash
# echo '(
select end.next_pid as pid, end.next_prio as prio,
    (end.common_timestamp.usecs - start.common_timestamp.usecs) as lat
from sched_waking as start
join sched_switch as end
on start.pid = end.next_pid) as wakeup_lat' | ./sqlhist

echo 'wakeup_lat pid_t pid u64 lat' > synthetic_events
echo 'wakeup_lat pid_t pid int prio u64 lat' > synthetic_events

echo 'hist:keys=pid:__arg0__=common_timestamp.usecs' > events/sched/sched_waking/trigger
echo 'hist:keys=next_pid:lat=common_timestamp.usecs-$__arg0__:onmatch(sched.sched_waking).trace(wakeup_lat,next_pid,next_prio,$lat)' > events/sched/sched_switch/trigger
```
Thank You

srostedt@vmware.com

@VMWopensource

blogs.vmware.com/opensource