



Ftrace and Perf

Go together in perfect harmony!

Injecting perf events to ftrace ring buffer

- Have a perf event trigger when a trace event is triggered
- Allows to use trace event filtering logic
- Can read perf counters between events like the scheduler

```
event_cache_misses  
event_cpu_cycles  
func-cache-misses  
func-cpu-cycles  
funcgraph-cache-misses  
funcgraph-cpu-cycles
```

Injecting perf events to ftrace ring buffer

```
# cd /sys/kernel/tracing
# echo 1 > options/event_cpu_cycles
# echo 1 > options/event_cache_misses
# echo 1 > events/syscalls/enable
# cat trace
[...]
```

```
bash-995      [007] ..... 98.255252: sys_write -> 0x2
bash-995      [007] ..... 98.255257: cpu_cycles: 1557241774 cache_misses: 449901166
bash-995      [007] ..... 98.255284: sys_dup2(oldfd: 0xa, newfd: 1)
bash-995      [007] ..... 98.255285: cpu_cycles: 1557260057 cache_misses: 449902679
bash-995      [007] ..... 98.255305: sys_dup2 -> 0x1
bash-995      [007] ..... 98.255305: cpu_cycles: 1557280203 cache_misses: 449906196
bash-995      [007] ..... 98.255343: sys_fcntl(fd: 0xa, cmd: 1, arg: 0)
bash-995      [007] ..... 98.255344: cpu_cycles: 1557322304 cache_misses: 449915522
bash-995      [007] ..... 98.255352: sys_fcntl -> 0x1
bash-995      [007] ..... 98.255353: cpu_cycles: 1557327809 cache_misses: 449916844
bash-995      [007] ..... 98.255361: sys_close(fd: 0xa)
bash-995      [007] ..... 98.255362: cpu_cycles: 1557335383 cache_misses: 449918232
bash-995      [007] ..... 98.255369: sys_close -> 0x0
```

There are many different perf events

```
~# perf list hw
```

List of pre-defined events (to be used in `-e` or `-M`):

branch-instructions OR branches	[Hardware event]
branch-misses	[Hardware event]
bus-cycles	[Hardware event]
cache-misses	[Hardware event]
cache-references	[Hardware event]
cpu-cycles OR cycles	[Hardware event]
instructions	[Hardware event]
ref-cycles	[Hardware event]

Should we only support the pre-defined ones?

```
~# perf list hw
```

List of pre-defined events (to be used in -e or -M):

branch-instructions OR branches	[Hardware event]
branch-misses	[Hardware event]
bus-cycles	[Hardware event]
cache-misses	[Hardware event]
cache-references	[Hardware event]
cpu-cycles OR cycles	[Hardware event]
instructions	[Hardware event]
ref-cycles	[Hardware event]

Should each perf event have its own option?

```
~# ls options/event_*  
options/event_branches      options/event_cache_references  
options/event_branch_misses options/event_cpu_cycles  
options/event_bus_cycles    options/event_instructions  
options/event_cache_misses  options/event_ref_cycles
```

Would a trigger be better?

```
~# echo 'perf:cache_misses' > events/sched/sched_switch/trigger
```

With multiple perf events

```
~# echo 'perf:cache_misses,branch_misses' > \
    events/sched/sched_switch/trigger
```

With filters

```
~# echo "perf:cache_misses,branch_misses if current_pid == $$" > \
    events/syscalls/sys_enter_read
~# echo "perf:cache_misses,branch_misses if current_pid == $$" > \
    events/syscalls/sys_exit_read
```

Raw perf event

```
u64 values[];  
  
// Use 56 bits for perf event values  
u64 perf_event = perf_event_read_local() & ((1LL << 56) - 1);  
  
// Use 8 bits for a perf id. Allows for 256 different events  
u64 perf_id = id << 56;  
  
if (id > 0xff)  
    return -EINVAL;
```

Raw perf event

```
u64 values[];
```

```
u64 perf_event = perf_event_read_local() & ((1LL << 56) - 1);  
u64 perf_id = id << 56;
```

```
if (id > 0xff)  
    return -EINVAL;
```

- Max of 256 (255 if we don't allow zero) different events
- Is that enough?
- What about non “pre-defined” events?

Extend the recorded perf events to all sw perf events?

- Have a tracefs file that maps the perf ids to the actual events
- Likely needs more than 256 different perf ids
- Have 52 bits for values and 12 bits for perf ids
- 2^{52} allows for 156 hours before overflow of cycles at 4 GHz

Do we need this for function / function_graph tracing?

- Would require a trigger for them
- Could we just get away with using kprobes and retprobes?
 - Do we want to trace “all functions”?

Discussion!

