perf kvm-events

Xiao Guangrong
<xiaoguangrong@linux.vnet.ibm.com>

Index

- Motivation
- perf kvm-events
- Future work

Motivation

- Quickly finding / resolving KVM issue is very important
  - KVM is very successful
  - But has become complex
- The presentation addresses introduction of perf kvm-events to help you quickly locate issues of KVM

perf kvm-events

- Background
- Advantage
- Usage
- Implementation

perf kvm-events

- Background
  - We lack a way to track the very detail events of KVM
    - e.g: most of current tools can count the event of MMIO access, but we hardly know the statistics of the accessed MMIO address
  - We lack a tool which can analyze the result more smartly
    - e.g: we can know the statistics of MMIO, but we hardly know the time of handling MMIO emulation and which MMIO address emulation has the most overload

perf kvm-events

- Advantage
  - perf kvm-events is a tool based on perf which can smartly analyze kvm events
  - Currently, it supports the events of vmexit, mmio and ioport
  - It can work for both old-kernels and new-kernels
  - It supports off-box analysis -- collect data on target and analyze it on another
  - **Note: the initial idea is from Xen**

perf kvm-events

- Usage
  - record events:
    - # ./perf kvm-events record
    - If you are interested in other events, using '-e' append the events, e.g,
      - # ./perf kvm-events record -e timer:*
  - Analyze kvm events:
    - # ./perf kvm-events report

```
[root@localhost perf]# ./perf kvm-events report help

 usage: perf kvm-events report [<options>]

        --event <report event>
                              event for reporting: vmexit, mmio, ioport
        --vcpu <n>            vcpu id to report
    -k, --key <sort-key>      key for sorting: sample(sort by samples number) time (
sort by avg time)
```

perf kvm-events

- Trace vm-exit
  - Show the event-analyze for all VCPUs, event sorted by samples number

```
[root@localhost perf]# ./perf kvm-events report --event=vmexit --key=sample

Analyze events for all VCPUs:

             VM-EXIT    Samples  Samples%     Time%         Avg time

          APIC_ACCESS       1466     49.05%     7.71%      9.25us ( +-   1.90% )
    EXTERNAL_INTERRUPT       1182     39.54%     7.18%     10.69us ( +-   2.02% )
     PENDING_INTERRUPT        244      8.16%     0.40%      2.89us ( +-   1.40% )
        EXCEPTION_NMI         53      1.77%     0.12%      3.85us ( +-   3.19% )
                  HLT         42      1.41%    84.57%   3542.66us ( +-   4.16% )
        EPT_VIOLATION          2      0.07%     0.02%     16.43us ( +-   0.85% )

Total Samples:2989, Total events handled time:175930.73us.
```

The reason cause VM-EXIT

Total event sample number

The weight of event sample number

The weight of event handle time

The avg event handle time

Standard Deviation

perf kvm-events

- Trace vm-exit
  - Show the event-analyze for all VCPUs, event sorted by avg-time

```
[root@localhost perf]# ./perf kvm-events report --event=vmexit --key=time

Analyze events for all VCPUs:

             VM-EXIT    Samples  Samples%     Time%         Avg time

                 HLT         42     1.41%    84.57%    3542.66us ( +-   4.16% )
        EPT_VIOLATION          2     0.07%     0.02%      16.43us ( +-   0.85% )
    EXTERNAL_INTERRUPT       1182    39.54%     7.18%      10.69us ( +-   2.02% )
          APIC_ACCESS        1466    49.05%     7.71%       9.25us ( +-   1.90% )
        EXCEPTION_NMI          53     1.77%     0.12%       3.85us ( +-   3.19% )
    PENDING_INTERRUPT         244     8.16%     0.40%       2.89us ( +-   1.40% )

Total Samples:2989, Total events handled time:175930.73us.
```

perf kvm-events

- Trace vm-exit
  - Show the event-analyze for VCPU 0, event sorted by avg-time

```
[root@localhost perf]# ./perf kvm-events report --event=vmexit --key=time --vcpu
=0

Analyze events for VCPU 0:

             VM-EXIT    Samples  Samples%     Time%         Avg time

  EXTERNAL_INTERRUPT        806    40.71%    46.23%     10.00us ( +-   2.53% )
         APIC_ACCESS       1044    52.73%    51.85%      8.66us ( +-   2.17% )
       EXCEPTION_NMI         34     1.72%     0.70%      3.61us ( +-   4.87% )
    PENDING_INTERRUPT         96     4.85%     1.22%      2.21us ( +-   2.09% )

Total Samples:1980, Total events handled time:17434.88us.
```

perf kvm-events

- Trace mmio events
  - Show the event-analyze for VCPU 0, event sorted by avg-time

```
[root@localhost perf]# ./perf kvm-events report --event=mmio --key=time --vcpu=0

Analyze events for VCPU 0:

         MMIO Access    Samples  Samples%     Time%         Avg time

       0xfee00300:W         43     7.34%    19.35%      7.48us ( +-   2.83% )
       0xfee00380:W        457    77.99%    73.22%      2.66us ( +-   1.07% )
       0xfee00300:R         43     7.34%     4.29%      1.66us ( +-   2.71% )
       0xfee00310:W         43     7.34%     3.13%      1.21us ( +-   2.49% )

Total Samples:586, Total events handled time:1662.84us.
```

W: MMIO write emulation
R: MMIO Read emulation

The MMIO address

perf kvm-events

- Trace ioport events
  - Show the event-analyze for VCPU 0, event sorted by avg-time

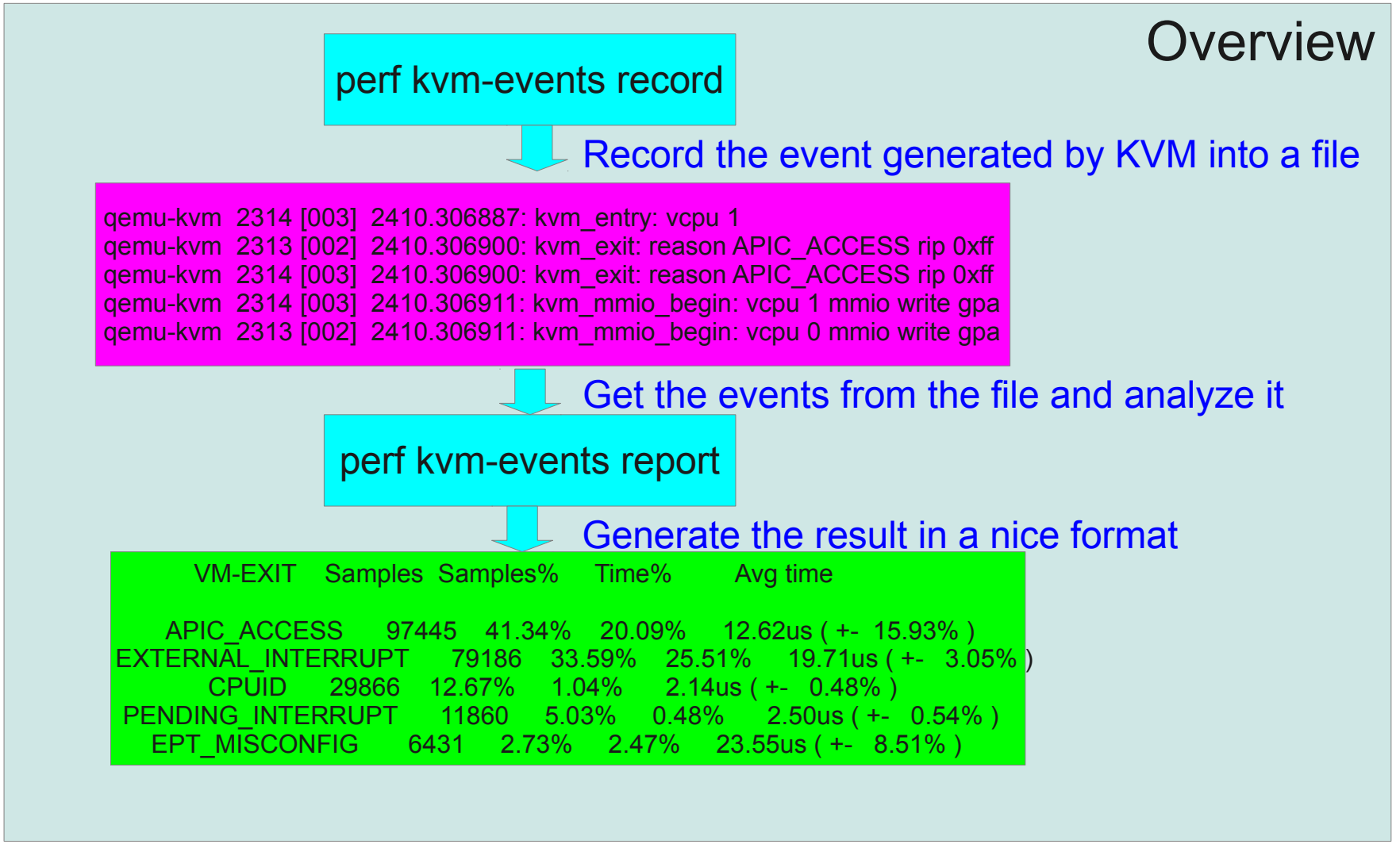
```
[root@localhost perf]# ./perf kvm-events report --event=ioport --key=time --vcpu
=0

Analyze events for VCPU 0:

      IO Port Access    Samples  Samples%      Time%         Avg time

         0x170:PIN          8      0.25%      0.79%      24.21us ( +-  34.61% )
         0x376:PIN         22      0.68%      1.65%      18.38us ( +-  53.44% )
        0xc000:POUT        31      0.95%      1.94%      15.40us ( +-  15.23% )
          0x60:PIN        424     13.05%     25.62%      14.85us ( +-   2.76% )
         0x1f7:PIN         43      1.32%      2.26%      12.89us ( +-   9.93% )
        0xc002:PIN         14      0.43%      0.66%      11.67us ( +-  16.08% )
          0x64:PIN        424     13.05%     17.72%      10.27us ( +-   5.08% )
         0x174:PIN          8      0.25%      0.32%       9.70us ( +-  48.86% )
         0x177:PIN         22      0.68%      0.80%       8.91us ( +-   6.95% )
        0x3c6:POUT          2      0.06%      0.07%       8.50us ( +-  28.86% )
        0xc000:PIN         31      0.95%      0.84%       6.69us ( +-  14.98% )
        0xc00a:PIN         14      0.43%      0.38%       6.61us ( +-  15.70% )
        0x1f6:POUT         24      0.74%      0.64%       6.57us ( +-  13.00% )
        0x3c9:POUT         96      2.95%      2.44%       6.25us ( +-   5.66% )
        0x1f7:POUT         12      0.37%      0.30%       6.15us ( +-  10.89% )
        0x175:POUT         16      0.49%      0.39%       5.95us ( +-  23.85% )
        0x3c8:POUT         32      0.98%      0.76%       5.84us ( +-   0.84% )
         0x3f6:PIN         38      1.17%      0.89%       5.73us ( +-   3.97% )
        0xc004:POUT        12      0.37%      0.28%       5.72us ( +-   6.58% )
         0x171:PIN          8      0.25%      0.18%       5.61us ( +-   8.53% )
        0x3d4:POUT       1180     36.31%     25.88%       5.39us ( +-   2.08% )
         0x3d5:PIN        336     10.34%      7.16%       5.24us ( +-   1.46% )
         0x175:PIN          8      0.25%      0.17%       5.23us ( +-   9.67% )
         0x173:PIN          8      0.25%      0.17%       5.20us ( +-   9.32% )
         0x172:PIN          8      0.25%      0.17%       5.20us ( +-   8.92% )
         0x176:PIN          8      0.25%      0.17%       5.16us ( +-   9.00% )
        0xc00a:POUT        14      0.43%      0.28%       4.93us ( +-   5.66% )
        0x1f1:POUT         16      0.49%      0.31%       4.78us ( +-   6.91% )
        0xc002:POUT         7      0.22%      0.14%       4.76us ( +-   9.19% )
        0x1f2:POUT         16      0.49%      0.30%       4.53us ( +-   5.99% )
        0x1f4:POUT         16      0.49%      0.29%       4.50us ( +-   5.95% )
        0x1f5:POUT         16      0.49%      0.29%       4.42us ( +-   6.06% )
        0x3d5:POUT        336     10.34%      5.76%       4.21us ( +-   1.59% )

Total Samples:3250, Total events handled time:24575.19us.
```

PIN: port read emulation
POUT: port write emulation

The Port Number

perf kvm-events

- Implementation



perf kvm-events

- Implementation
  - Abstract for the event

```
struct event_key {
        #define INVALID_KEY     (~0ULL)
        u64 key;
        int info;
};
```

```
qemu-kvm  2313 [002]  2411.587516: kvm_mmio: mmio write len 4 gpa 0xfee00380 val 0xe676
```

event_key->key = 0xfee00380 event_key->info = 1 (1 = write, 0 = read)

perf kvm-events

- Implementation
  - Abstract for the event operation

```
struct kvm_events_ops {
        bool (*is_begin_event)(struct event *event, void *data,
                               struct event_key *key);
        bool (*is_end_event)(struct event *event, void *data,
                             struct event_key *key);
        void (*decode_key)(struct event_key *key, char decode[20]);
        const char *name;
};
```

is_begin_event: recognize the begin event and encode the event to the event_key
is_end_event: recognize the end event and encode the event_key
decode_key: decode the event_key to the readable format

event-key is decoded to: “0xfee00380:W”

event_key = {.key = 0xfee00380, .info = 1}

```
qemu-kvm  2313 [002]  2411.587516: kvm_mmio: mmio write len 4 gpa 0xfee00380 val 0xe676
qemu-kvm  2313 [002]  2411.587520: kvm_mmio_done: vcpu 0
```

perf kvm-events

- Implementation

```
static struct kvm_events_ops exit_events = {
        .is_begin_event = exit_event_begin,
        .is_end_event = exit_event_end,
        .decode_key = exit_event_decode_key,
        .name = "VM-EXIT"
};
```

--event = vmexit

```
static struct kvm_events_ops mmio_events = {
        .is_begin_event = mmio_event_begin,
        .is_end_event = mmio_event_end,
        .decode_key = mmio_event_decode_key,
        .name = "MMIO Access"
};
```

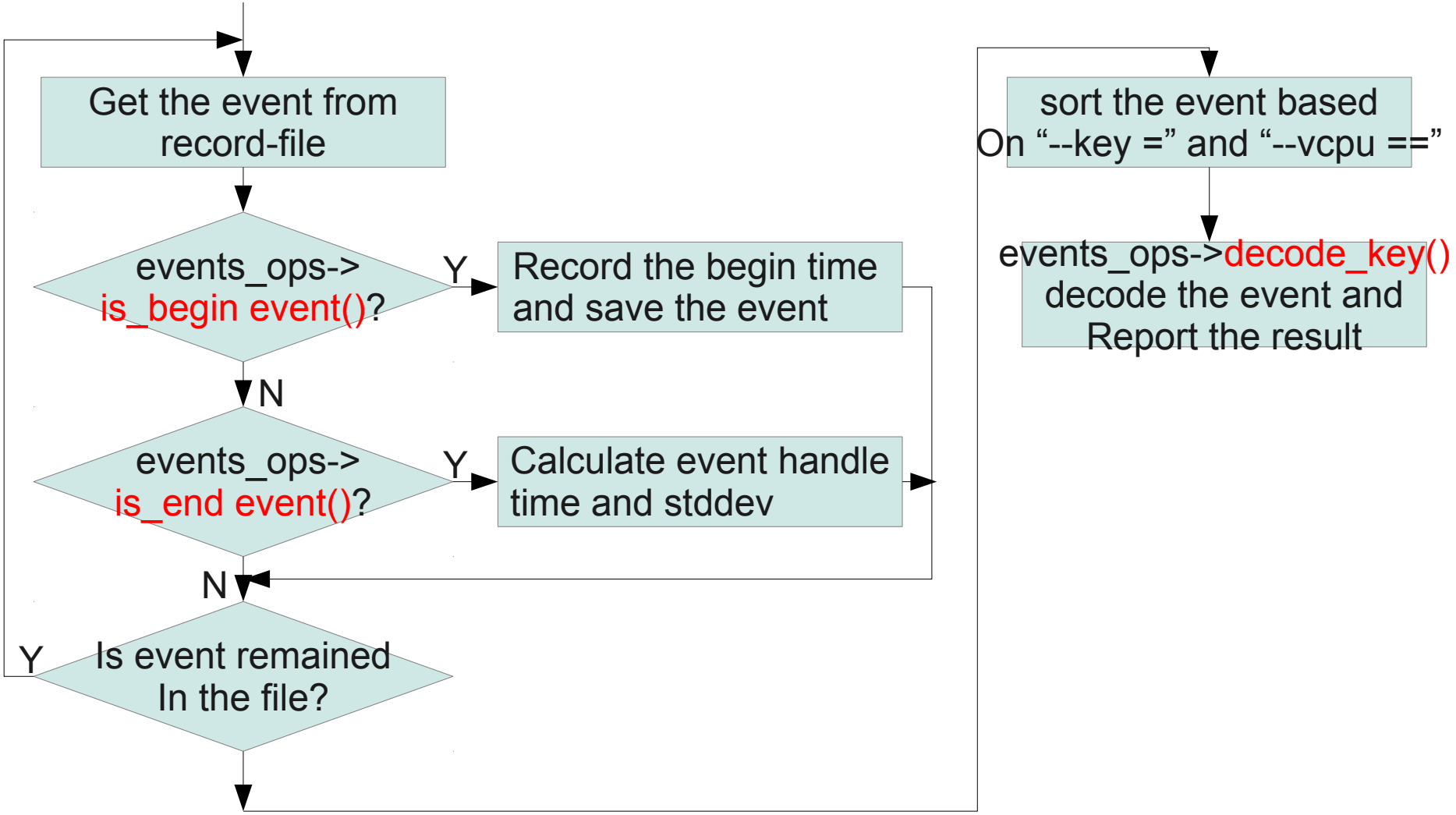
--event = mmio

```
static struct kvm_events_ops ioport_events = {
        .is_begin_event = ioport_event_begin,
        .is_end_event = ioport_event_end,
        .decode_key = ioport_event_decode_key,
        .name = "IO Port Access"
};
```

--event = ioport

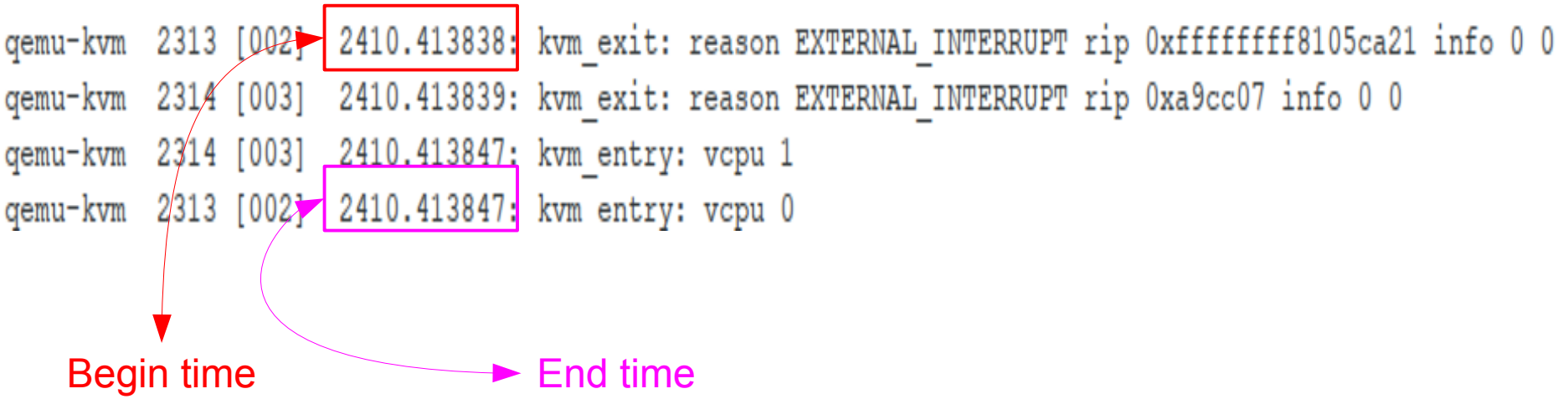
perf kvm-events

- Implementation: work flow of analysis events



perf kvm-events

- Implementation: work flow of analysis events



event handle time = end time – begin time = 2410.413847 – 2410.413838 = 0.000009 (s)

perf kvm-events

- Recognize begin event and end event
  - In the current code, some tracepoints can help us to detect the begin time and the end time, but it is not exact (old kernel).
  - So, we add some tracepoints to make things better.

perf kvm-events

- Recognize begin and end event for vm-exit
  - Begin: kvm_exit
  - End: kvm_entry
  - Both events exist in the currently KVM code, perf kvm-events easily works for old-kernel and new-kernel

```
qemu-kvm  2313 [002]  2410.413838: kvm_exit: reason EXTERNAL_INTERRUPT rip 0xffffffff8105ca21 info 0 0
qemu-kvm  2314 [003]  2410.413839: kvm_exit: reason EXTERNAL_INTERRUPT rip 0xa9cc07 info 0 0
qemu-kvm  2314 [003]  2410.413847: kvm_entry: vcpu 1
qemu-kvm  2313 [002]  2410.413847: kvm_entry: vcpu 0
```

perf kvm-events

- Recognize begin and end event for MMIO access
  - In current code, we have the tracepoint named kvm_mmio to trace mmio access

```
#define kvm_trace_symbol_mmio \
        { KVM_TRACE_MMIO_READ_UNSATISFIED, "unsatisfied-read" }, \
        { KVM_TRACE_MMIO_READ, "read" }, \
        { KVM_TRACE_MMIO_WRITE, "write" }

TRACE_EVENT(kvm_mmio,
        TP_PROTO(int type, int len, u64 gpa, u64 val),
        TP_ARGS(type, len, gpa, val),
```

    - KVM_TRACE_MMIO_READ_UNSATISFIED: need return to userspace to emulation MMIO access
    - KVM_TRACE_MMIO_READ: the mmio read emulation has finished
    - KVM_TRACE_MMIO_WRITE: the mmio write emulation is begin

perf kvm-events

- Recognize begin and end event for MMIO access
  - So, for the old kernel:
    - The handle time of MMIO read emulation is from kvm_exit to kvm_mmio(KVM_TRACE_MMIO_READ, ...)
    - The handle time of MMIO write emulation is from kvm_mmio(KVM_TRACE_MMIO_WRITE, ...) to kvm_entry
  - This is just a approximate way since lots of overload is included

perf kvm-events

- Recognize begin and end event for MMIO access
  - In order to let the result to be more exact, we add two tracepoints to trace the mmio begin and end, that is, kvm_mmio_begin and kvm_mmio_done
  - They are used to trace MMIO begin and end time in ther new kernel

perf kvm-events

- Recognize begin and end event for MMIO access

```
qemu-kvm  2314 [003]  2410.310919: kvm_mmio_begin: vcpu 1 mmio write gpa 0xfee00380
qemu-kvm  2313 [002]  2410.310920: kvm_mmio_done: vcpu 0
qemu-kvm  2314 [003]  2410.310921: kvm_mmio: mmio write len 4 gpa 0xfee00380 val 0xe94e
qemu-kvm  2313 [002]  2410.310924: kvm_entry: vcpu 0
qemu-kvm  2314 [003]  2410.310924: kvm_mmio_done: vcpu 1
qemu-kvm  2314 [003]  2410.310987: kvm_entry: vcpu 1
```

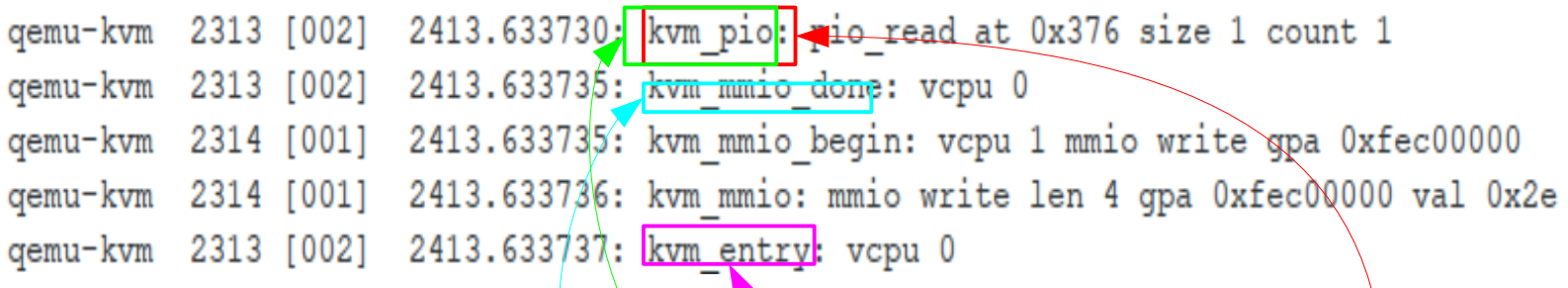
  - For old kernel: handle time = end time – begin time = 2410.310987 – 2410.310921 = **0.000066** (s)
  - For new kernel: handle time = end time – begin time = 2410.310924 – 2410.310919 = **0.000005** (s)

perf kvm-events

- Recognize begin and end event for IO port access
  - In current code, we have the tracepoint named kvm_pio to trace io port access which indicates the io access emulation is begin
  - So, for the old kernel, the io access emulation time is from kvm_pio to kvm_entry
  - In the new kernel, the handle time is from kvm_pio to kvm_mmio_done, it is more exact than old kernel

perf kvm-events

- Recognize begin and end event for IO Port access



  - For old kernel: handle time = end time – begin time = 2413.633737 – 2413.633730 = **0.000007** (s)
  - For new kernel: handle time = end time – begin time = 2413.633735 – 2413.633730 = **0.000005** (s)

Future work

- Pull perf kvm-events to mainline
- Refine perf kvm-events, let it to support more events (e.g. MMU events...)
- perf kvm-events support on PowerPC
- Per VM tracking support

Questions?

- Any questions/comments are welcome!

Thanks! :)