KIFÜ

A MAGYARORSZÁGI **DIGITALIZÁCIÓ** SZOLGÁLATÁBAN

Nyomkövetés Linux környezetben

Wágner Ferenc
KIFÜ

Networkshop 2018

Aggasztó log üzenetek

A központi virtualizációs clusterünkben:

```
vhbl08 corosync[3687]: [TOTEM ] A processor failed, forming new configuration.
vhbl03 corosync[3890]: [TOTEM ] A processor failed, forming new configuration.
vhbl07 corosync[3805]: [MAIN  ] Corosync main process was not scheduled for
    4317.0054 ms (threshold is 2400.0000 ms). Consider token timeout increase.
```

- májusban heti 5
- augusztusban már napi 5!

A cluster tagság tényleges széthullása súlyos üzemeltetési probléma lenne:

- szolgáltatáskiesés
- adatvesztés

A közösség (levelezési lista) tippel

- Túlterhelt a gép $\Rightarrow$ 3200% CPU idle

A közösség (levelezési lista) tippel

- Túlterhelt a gép $\Rightarrow$ 3200% CPU idle
- Virtuális gépben futunk, és túlterhelt a host $\Rightarrow$ nem

A közösség (levelezési lista) tippel

- Túlterhelt a gép $\Rightarrow$ 3200% CPU idle
- Virtuális gépben futunk, és túlterhelt a host $\Rightarrow$ nem
- Nincs lockolva a memória (ismert hiba) $\Rightarrow$ nem használjuk a démonizáló kódot

A közösség (levelezési lista) tippel

- Túlterhelt a gép $\Rightarrow$ 3200% CPU idle
- Virtuális gépben futunk, és túlterhelt a host $\Rightarrow$ nem
- Nincs lockolva a memória (ismert hiba) $\Rightarrow$ nem használjuk a démonizáló kódot
- Csak növeljük meg a token timeoutot $\Rightarrow$ gagyi

A közösség (levelezési lista) tippel

- Túlterhelt a gép $\Rightarrow$ 3200% CPU idle
- Virtuális gépben futunk, és túlterhelt a host $\Rightarrow$ nem
- Nincs lockolva a memória (ismert hiba) $\Rightarrow$ nem használjuk a démonizáló kódot
- Csak növeljük meg a token timeoutot $\Rightarrow$ gagyi
- KSM page-locking hiba $\Rightarrow$ csak nagyon régi kernelekben

A közösség (levelezési lista) tippel

- Túlterhelt a gép $\Rightarrow$ 3200% CPU idle
- Virtuális gépben futunk, és túlterhelt a host $\Rightarrow$ nem
- Nincs lockolva a memória (ismert hiba) $\Rightarrow$ nem használjuk a démonizáló kódot
- Csak növeljük meg a token timeoutot $\Rightarrow$ gagyi
- KSM page-locking hiba $\Rightarrow$ csak nagyon régi kernelekben
- replikált cLVM mirror log lassú $\Rightarrow$ mi?

A közösség (levelezési lista) tippel

- Túlterhelt a gép $\Rightarrow$ 3200% CPU idle
- Virtuális gépben futunk, és túlterhelt a host $\Rightarrow$ nem
- Nincs lockolva a memória (ismert hiba) $\Rightarrow$ nem használjuk a démonizáló kódot
- Csak növeljük meg a token timeoutot $\Rightarrow$ gagyi
- KSM page-locking hiba $\Rightarrow$ csak nagyon régi kernelekben
- replikált cLVM mirror log lassú $\Rightarrow$ mi?

Nem értjük.
Talán csak mérési (program)hiba?

A gdb most nem opció, mert leállítja a programot.

Hova tegyük a szondát?

```
$ perf probe -x /usr/sbin/corosync --source ~/corosync-2.4.2 \
             --line timer_function_scheduler_timeout
```

Hova tegyük a szondát?

```
$ perf probe -x /usr/sbin/corosync --source ~/corosync-2.4.2 \
             --line timer_function_scheduler_timeout
 0  static void timer_function_scheduler_timeout (void *data)
 1  {
      struct scheduler_pause_timeout_data *timeout_data = (struct scheduler_pause_timeout_data *)data;
      unsigned long long tv_current;
      unsigned long long tv_diff;

 6    tv_current = qb_util_nano_current_get ();

 8    if (timeout_data->tv_prev == 0) {
        /* Initial call -> just pretent everything is ok */
        timeout_data->tv_prev = tv_current;
        timeout_data->max_tv_diff = 0;
      }

      tv_diff = tv_current - timeout_data->tv_prev;
17    timeout_data->tv_prev = tv_current;

19    if (tv_diff > timeout_data->max_tv_diff) {
20      log_printf (LOGSYS_LEVEL_WARNING, "Corosync main process was not scheduled for %0.4f ms "
            "(threshold is %0.4f ms). Consider token timeout increase.",
            (float)tv_diff / QB_TIME_NS_IN_MSEC, (float)timeout_data->max_tv_diff / QB_TIME_NS_IN_MSEC);
      }
      /* Set next threshold, because token_timeout can change */
28    timeout_data->max_tv_diff = timeout_data->totem_config->token_timeout * QB_TIME_NS_IN_MSEC * 0.8;
29    qb_loop_timer_add (corosync_poll_handle,
        QB_LOOP_MED,
31      timeout_data->totem_config->token_timeout * QB_TIME_NS_IN_MSEC / 3,
        timeout_data,
        timer_function_scheduler_timeout,
        &timeout_data->handle);
35  }
```

Hova tegyük a szondát?

```
$ perf probe -x /usr/sbin/corosync --source ~/corosync-2.4.2 \
             --line timer_function_scheduler_timeout
 0  static void timer_function_scheduler_timeout (void *data)
 1  {
      struct scheduler_pause_timeout_data *timeout_data = (struct scheduler_pause_timeout_data *)data;
      unsigned long long tv_current;
      unsigned long long tv_diff;

 6    tv_current = qb_util_nano_current_get ();

 8    if (timeout_data->tv_prev == 0) {
        /* Initial call -> just pretent everything is ok */
        timeout_data->tv_prev = tv_current;
        timeout_data->max_tv_diff = 0;
      }

      tv_diff = tv_current - timeout_data->tv_prev;
17    timeout_data->tv_prev = tv_current;

⇒19   if (tv_diff > timeout_data->max_tv_diff) {
20      log_printf (LOGSYS_LEVEL_WARNING, "Corosync main process was not scheduled for %0.4f ms "
            "(threshold is %0.4f ms). Consider token timeout increase.",
            (float)tv_diff / QB_TIME_NS_IN_MSEC, (float)timeout_data->max_tv_diff / QB_TIME_NS_IN_MSEC);
      }
      /* Set next threshold, because token_timeout can change */
28    timeout_data->max_tv_diff = timeout_data->totem_config->token_timeout * QB_TIME_NS_IN_MSEC * 0.8;
29    qb_loop_timer_add (corosync_poll_handle,
        QB_LOOP_MED,
31      timeout_data->totem_config->token_timeout * QB_TIME_NS_IN_MSEC / 3,
        timeout_data,
        timer_function_scheduler_timeout,
        &timeout_data->handle);
35 }
```

Nosza!

Ügyesen használja a debug infót:

```
$ perf probe -x /usr/sbin/corosync --definition \
             'timer_function_scheduler_timeout:19 tv_diff'
```

Nosza!

Ügyesen használja a debug infót:

```
$ perf probe -x /usr/sbin/corosync --definition \
             'timer_function_scheduler_timeout:19 tv_diff'
p:probe_corosync/timer_function_scheduler_timeout
                 /usr/sbin/corosync:0x26097 tv_diff=%si:x64
```

Nosza!

Ügyesen használja a debug infót:

```
$ perf probe -x /usr/sbin/corosync --definition \
             'timer_function_scheduler_timeout:19 tv_diff'
p:probe_corosync/timer_function_scheduler_timeout
                 /usr/sbin/corosync:0x26097 tv_diff=%si:x64
```

És tényleg, a tárgykód:

```
(gdb) disassemble/m timer_function_scheduler_timeout
868             if (tv_diff > timeout_data->max_tv_diff) {
   0x0000000000026097 <+39>:    cmp    %rcx,%rsi
   0x000000000002609a <+42>:    jbe    0x260e9 <t_f_s_t+121>
869                     log_printf (LOGSYS_LEVEL_WARNING, "Corosync main…
```

Nosza!

Ügyesen használja a debug infót:

```
$ perf probe -x /usr/sbin/corosync --definition \
             'timer_function_scheduler_timeout:19 tv_diff'
p:probe_corosync/timer_function_scheduler_timeout
                 /usr/sbin/corosync:0x26097 tv_diff=%si:x64
```

És tényleg, a tárgykód:

```
(gdb) disassemble/m timer_function_scheduler_timeout
868             if (tv_diff > timeout_data->max_tv_diff) {
   0x0000000000026097 <+39>:    cmp    %rcx,%rsi
   0x000000000002609a <+42>:    jbe    0x260e9 <t_f_s_t+121>
869                     log_printf (LOGSYS_LEVEL_WARNING, "Corosync main…
```

Használjuk, de decimálisan!

```
# perf probe -x /usr/sbin/corosync --add \
             'timer_function_scheduler_timeout:19 tv_diff:u64'
Added new event:
  probe_corosync:timer_function_scheduler_timeout
    (on t_f_s_t:19 in /usr/sbin/corosync with tv_diff_u64)
You can now use it in all perf tools, such as:
  perf record -e probe_corosync:timer_function_scheduler_timeout -aR sleep 1
```

Milyen társaságba keveredtünk?

```
# perf list
  instructions                                                [Hardware event]
  context-switches OR cs                                      [Software event]
  cstate_pkg/c7-residency/                                    [Kernel PMU event]
  ext4:ext4_sync_fs                                           [Tracepoint event]
  kvm:kvm_cpuid                                               [Tracepoint event]
  module:module_load                                          [Tracepoint event]
  probe_corosync:timer_function_scheduler_timeout             [Tracepoint event]
  sched:sched_switch                                          [Tracepoint event]
  syscalls:sys_enter_open                                     [Tracepoint event]
  syscalls:sys_exit_open                                      [Tracepoint event]
  xfs:xfs_rename                                              [Tracepoint event]
```

Milyen társaságba keveredtünk?

```
# perf list
  instructions                                       [Hardware event]
  context-switches OR cs                             [Software event]
  cstate_pkg/c7-residency/                           [Kernel PMU event]
  ext4:ext4_sync_fs                                  [Tracepoint event]
  kvm:kvm_cpuid                                      [Tracepoint event]
  module:module_load                                 [Tracepoint event]
  probe_corosync:timer_function_scheduler_timeout    [Tracepoint event]
  sched:sched_switch                                 [Tracepoint event]
  syscalls:sys_enter_open                            [Tracepoint event]
  syscalls:sys_exit_open                             [Tracepoint event]
  xfs:xfs_rename                                     [Tracepoint event]

# perf record -e probe_corosync:timer_function_scheduler_timeout \
              -a sleep 3
[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 0.932 MB perf.data (3 samples) ]
```

Lássuk, mit lőttünk!

```
# chown wferi:wferi perf.data
$ perf script
corosync  3772 [016] 9310439.316062:
  probe_corosync:timer_function_scheduler_timeout:
  (563dffa81097) tv_diff_u64=1000006409

corosync  3772 [016] 9310440.316053:
  probe_corosync:timer_function_scheduler_timeout:
  (563dffa81097) tv_diff_u64=1000018919

corosync  3772 [016] 9310441.316045:
  probe_corosync:timer_function_scheduler_timeout:
  (563dffa81097) tv_diff_u64=1000017916
```

A mért időkülönbségek pontosan követik az időbélyegeket

$\Rightarrow$ a jelenség nem mérési hiba

Miért nem ütemeződik a Corosync?

- maximális (99) prioritású realtime folyamat
- ami nem nyúl diszkhez
- egy lényegében tétlen 16 magos gépen

?

Miért nem ütemeződik a Corosync?

- maximális (99) prioritású realtime folyamat
- ami nem nyúl diszkhez
- egy lényegében tétlen 16 magos gépen

?

```
# perf record -e probe_corosync:timer_function_scheduler_timeout \
              -e sched:sched_switch -p 3796 sleep 30
```

Miért nem ütemeződik a Corosync?

- maximális (99) prioritású realtime folyamat
- ami nem nyúl diszkhez
- egy lényegében tétlen 16 magos gépen

?

```
# perf record -e probe_corosync:timer_function_scheduler_timeout \
              -e sched:sched_switch -p 3796 sleep 30
$ perf script
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4564.26: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4564.38: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1000017434
4564.38: sched:sched_switch: corosync:3796 [0] S ==> kworker/16:21:1698 [120]
4564.46: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4564.48: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4565.38: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1000017354
4565.81: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4566.29: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4566.38: sched:sched_switch: corosync:3796 [0] D ==> swapper/16:0 [120]
4566.39: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1009431772
```

Prioritás-kvíz

```
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
```

Prioritás-kvíz

```
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
```

Hányféle prioritást tud kiírni a ps parancs?

Prioritás-kvíz

```
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
```

Hányféle prioritást tud kiírni a ps parancs?

- Egyet: a nice értéket

Prioritás-kvíz

```
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
```

Hányféle prioritást tud kiírni a ps parancs?

- Egyet: a nice értéket
- Kettőt: a POSIX real time priority-t is

Prioritás-kvíz

```
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
```

Hányféle prioritást tud kiírni a ps parancs?

- Egyet: a nice értéket
- Kettőt: a POSIX real time priority-t is
- Hármat: meg még az ütemezési osztályt (RR, FIFO, stb.)

Prioritás-kvíz

```
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
```

Hányféle prioritást tud kiírni a ps parancs?

- Egyet: a nice értéket
- Kettőt: a POSIX real time priority-t is
- Hármat: meg még az ütemezési osztályt (RR, FIFO, stb.)
- Négyet: és a kernel fenti numerikus reprezentációját is

Prioritás-kvíz

```
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
```

Hányféle prioritást tud kiírni a ps parancs?

- Egyet: a nice értéket
- Kettőt: a POSIX real time priority-t is
- Hármat: meg még az ütemezési osztályt (RR, FIFO, stb.)
- Négyet: és a kernel fenti numerikus reprezentációját is
- Hetet: ráadásul három Irix/HP-UX/Sun legacy értéket

Prioritás-kvíz

```
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
```

Hányféle prioritást tud kiírni a ps parancs?

- Egyet: a nice értéket
- Kettőt: a POSIX real time priority-t is
- Hármat: meg még az ütemezési osztályt (RR, FIFO, stb.)
- Négyet: és a kernel fenti numerikus reprezentációját is
- Hetet: ráadásul három Irix/HP-UX/Sun legacy értéket
- Tizet: plusz FOO, BAR és BAZ

Prioritás-kvíz

```
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
```

Hányféle prioritást tud kiírni a ps parancs?

- Egyet: a nice értéket
- Kettőt: a POSIX real time priority-t is
- Hármat: meg még az ütemezési osztályt (RR, FIFO, stb.)
- Négyet: és a kernel fenti numerikus reprezentációját is
- Hetet: ráadásul három Irix/HP-UX/Sun legacy értéket
- Tizet: plusz FOO, BAR és BAZ

```
$ ps -eo pid,class,nice,rtprio,priority,opri,pri_foo,pri_bar,pri_baz,\
            pri,pri_api,comm
  PID CLS  NI RTPRIO PRI PRI FOO BAR BAZ PRI API COMMAND
    9 FF    -     99 -100 -40 -120 -99 0 139  99 migration/0
   26 TS    5      -  25  85   5  26 125  14 -26 ksmd
   27 TS   19      -  39  99  19  40 139   0 -40 khugepaged
  777 TS    1      -  21  81   1  22 121  18 -22 rtkit-daemon
  778 TS    0      -  20  80   0  21 120  19 -21 rsyslogd
  854 RR    -     99 -100 -40 -120 -99 0 139  99 corosync
```

Miért nem ütemeződik a Corosync?

- maximális (99) prioritású realtime folyamat
- ami nem nyúl diszkhez
- egy lényegében tétlen 16 magos gépen

?

```
# perf record -e probe_corosync:timer_function_scheduler_timeout \
              -e sched:sched_switch -p 3796 sleep 30
$ perf script
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4564.26: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4564.38: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1000017434
4564.38: sched:sched_switch: corosync:3796 [0] S ==> kworker/16:21:1698 [120]
4564.46: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4564.48: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4565.38: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1000017354
4565.81: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4566.29: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4566.38: sched:sched_switch: corosync:3796 [0] D ==> swapper/16:0 [120]
4566.39: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1009431772
```

Miért nem ütemeződik a Corosync?

- maximális (99) prioritású realtime folyamat
- ami nem nyúl diszkhez
- egy lényegében tétlen 16 magos gépen

?

```
# perf record -e probe_corosync:timer_function_scheduler_timeout \
              -e sched:sched_switch -p 3796 sleep 30
$ perf script
4564.23: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4564.26: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4564.38: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1000017434
4564.38: sched:sched_switch: corosync:3796 [0] S ==> kworker/16:21:1698 [120]
4564.46: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4564.48: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4565.38: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1000017354
4565.81: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4566.29: sched:sched_switch: corosync:3796 [0] S ==> swapper/16:0 [120]
4566.38: sched:sched_switch: corosync:3796 [0] D ==> swapper/16:0 [120]
4566.39: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1009431772
```

Az a D állapot meglepő (disk sleep?!) és itt három nagyságrendnyi anomália mellett jelent meg!

Találtunk valamit?

Nézzük csak a legalább 1.001 s késleltetés előtti váltásokat!

```
# perf record -e probe_corosync:timer_function_scheduler_timeout \
              --filter 'tv_diff_u64 >= 1000100000' \
              -e sched:sched_switch -p 3796 sleep 30
$ perf script | grep -B1 tv_diff
```

Találtunk valamit?

Nézzük csak a legalább 1.001 s késleltetés előtti váltásokat!

```
# perf record -e probe_corosync:timer_function_scheduler_timeout \
              --filter 'tv_diff_u64 >= 1000100000' \
              -e sched:sched_switch -p 3796 sleep 30
$ perf script | grep -B1 tv_diff
6329.27: sched:sched_switch: corosync:3796 [0] D ==> swapper/16:0 [120]
6329.27: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1001193436
--
6332.27: sched:sched_switch: corosync:3796 [0] D ==> swapper/16:0 [120]
6332.27: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1001102890
--
6335.27: sched:sched_switch: corosync:3796 [0] D ==> swapper/16:0 [120]
6335.27: p_c:t_f_s_t: (55cc5e2c3097) tv_diff_u64=1001067313
```

Valószínű, hogy megtaláltuk az anomália közvetlen okát!

Kövessük a nyomot!

Stack trace-t szeretnénk a „kiütemezés” pillanatáról

De triggereket a `perf` eszköz még nem kezel $\Rightarrow$ közvetlenül kell használnunk a tracefs-t

Kövessük a nyomot!

Stack trace-t szeretnénk a „kiütemezés” pillanatáról

De triggereket a perf eszköz még nem kezel ⇒ közvetlenül kell használnunk a tracefs-t

```
# cd /sys/kernel/debug/tracing
# echo 3796 >set_event_pid

# echo 'traceoff if tv_diff_u64 >= 1000100000'
        >events/probe_corosync/timer_function_scheduler_timeout/trigger
# echo 1 >events/probe_corosync/timer_function_scheduler_timeout/enable

# echo 'prev_state == 2'                 >events/sched/sched_switch/filter
# echo 'stacktrace if prev_state == 2' >events/sched/sched_switch/trigger
# echo 1                                 >events/sched/sched_switch/enable

# echo >trace
# echo 1 >tracing_on
# cat tracing_on
0
```

Drasztikusan csökkent a kernelből kiolvasandó adatmennyiség!

Ki a gyilkos?

```
# cat trace
# tracer: nop
# entries-in-buffer/entries-written: 5/5   #P:32
#     TASK-PID   CPU#          TIMESTAMP  FUNCTION
  corosync-3796  [000] d... 10147.577289: t_f_s_t: tv_diff_u64=1000014780
  corosync-3796  [000] d... 10148.577356: t_f_s_t: tv_diff_u64=1000017148
  corosync-3796  [000] d... 10149.577391: sched_switch: prev_comm=corosync
                                prev_pid=3796 prev_prio=0 prev_state=D ==>
                                next_comm=swapper/0 next_pid=0 next_prio=120
  corosync-3796  [000] d... 10149.577403: <stack trace>
 => sender
 => i_ipmi_request
 => hrtimer_start_range_ns
 => wait_for_completion
 => default_wake_function
 => ipmi_heartbeat
 => ipmi_unlocked_ioctl
 => do_vfs_ioctl
 => default_wake_function
 => SyS_ioctl
 => system_call_fast_compare_end
  corosync-3796  [000] d... 10149.578431: t_f_s_t: tv_diff_u64=1001026856
```

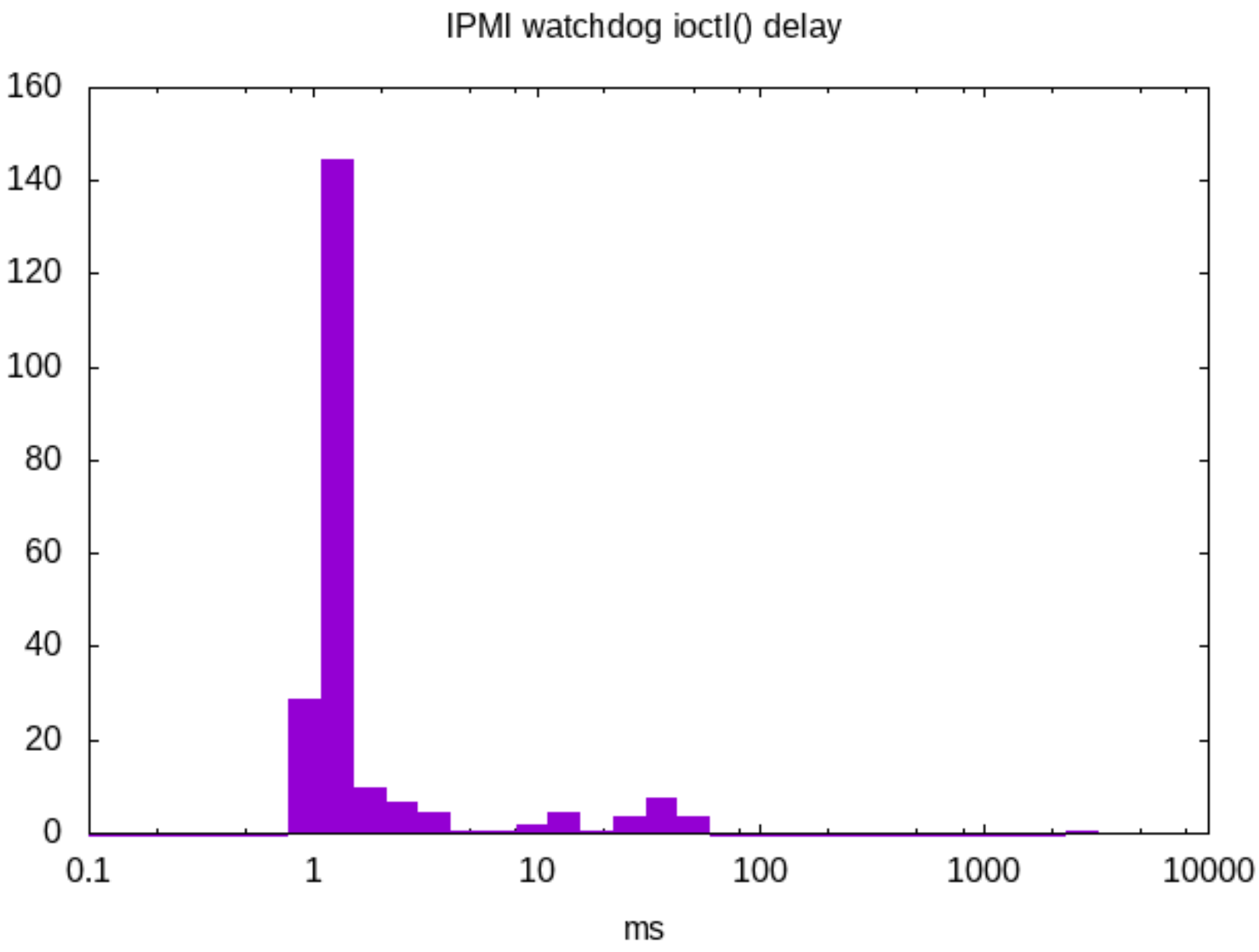
Rabosítás

```
# echo 1 >events/syscalls/sys_enter_ioctl/enable
# echo >trace
# echo 1 >tracing_on
# cat trace
corosync-3796  [000] .... 309122.625332: sys_ioctl(fd: 10, cmd: 80045705,
                                                    arg: 55cc5e4e5390)
corosync-3796  [000] d... 309122.625350: sched_switch: prev_comm=corosync
                             prev_pid=3796 prev_prio=0 prev_state=D ==>
                             next_comm=swapper/0 next_pid=0 next_prio=120
corosync-3796  [000] d... 309122.625361: <stack trace>
  [...]
corosync-3796  [000] d... 309122.626391: t_f_s_t: tv_diff_u64=1001024970
```

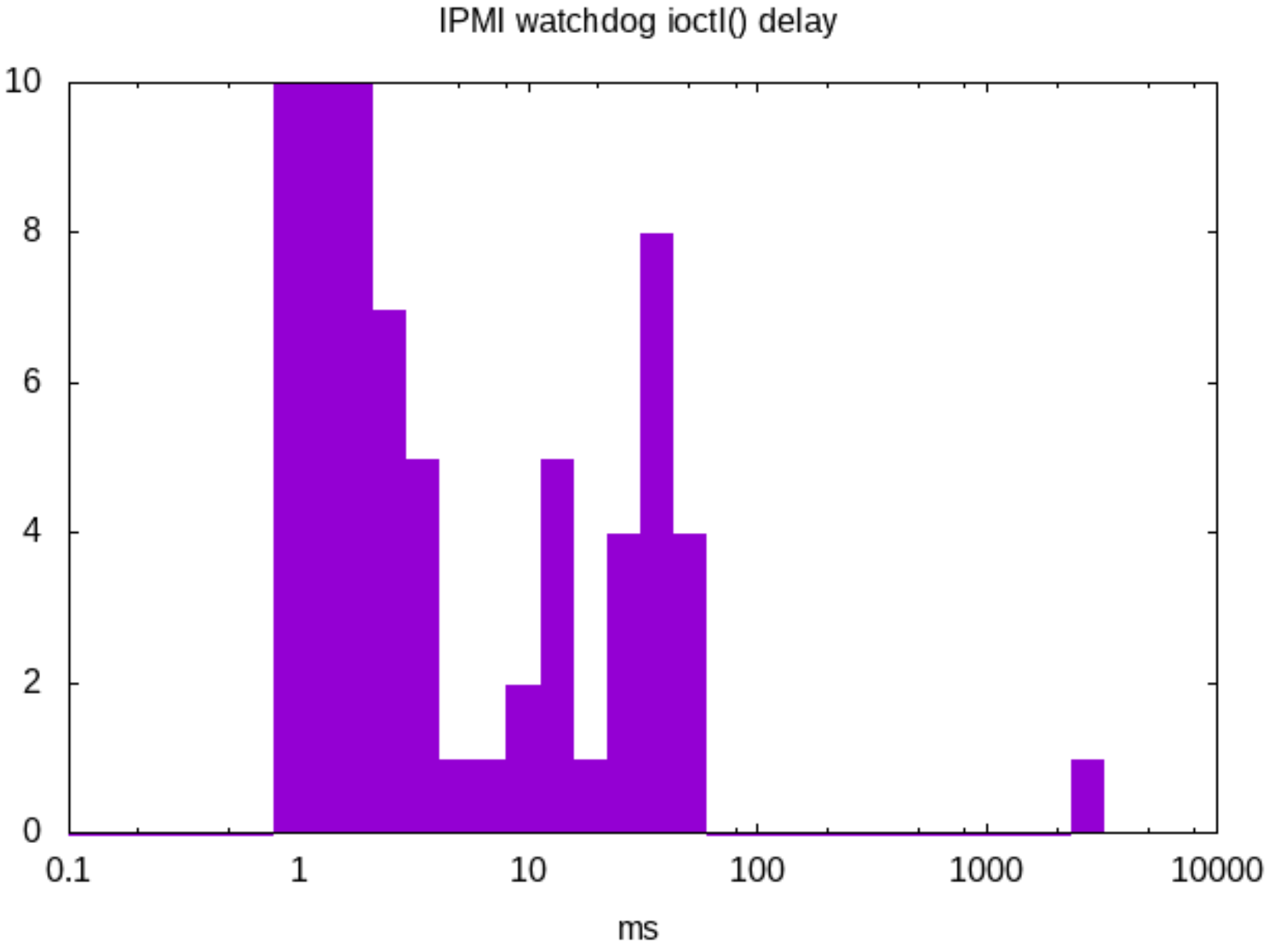
Rabosítás

```
# echo 1 >events/syscalls/sys_enter_ioctl/enable
# echo >trace
# echo 1 >tracing_on
# cat trace
corosync-3796  [000] .... 309122.625332: sys_ioctl(fd: 10, cmd: 80045705,
                                                    arg: 55cc5e4e5390)
corosync-3796  [000] d... 309122.625350: sched_switch: prev_comm=corosync
                             prev_pid=3796 prev_prio=0 prev_state=D ==>
                             next_comm=swapper/0 next_pid=0 next_prio=120
corosync-3796  [000] d... 309122.625361: <stack trace>
  [...]
corosync-3796  [000] d... 309122.626391: t_f_s_t: tv_diff_u64=1001024970
# ls -l /proc/3796/fd/16
l-wx------ 1 root root 64 Apr  1 18:29 /proc/3796/fd/16 -> /dev/watchdog
```

Tanulság


IPMI watchdog ioctl() delay
160
140
120
100
80
60
40
20
0
0.1
1
10
100
1000
10000
ms

Tanulság



1079 minta. A szerver IPMI implementációja *nagyon* gáz.

Perspektíva

- A klasszikus eszközök (`top`, `vmstat`, stb.) csak átlagokat mutatnak.

Perspektíva

- A klasszikus eszközök (`top`, `vmstat`, stb.) csak átlagokat mutatnak.
- A `perf` eszköz kényelmes, de itt-ott még korlátozott.

Perspektíva

- A klasszikus eszközök (`top`, `vmstat`, stb.) csak átlagokat mutatnak.
- A `perf` eszköz kényelmes, de itt-ott még korlátozott.
- A trace eventekhez már eBPF programok csatolhatók, vagyis például az előző hisztogramot elkészíthettük volna a kernelben!

KIFÜ

Köszönöm a figyelmet!

www.kifu.gov.hu

BPF in action

```
$ sudo tcpdump ip -d
(000) ldh      [12]
(001) jeq      #0x800           jt 2        jf 3
(002) ret      #262144
(003) ret      #0
```