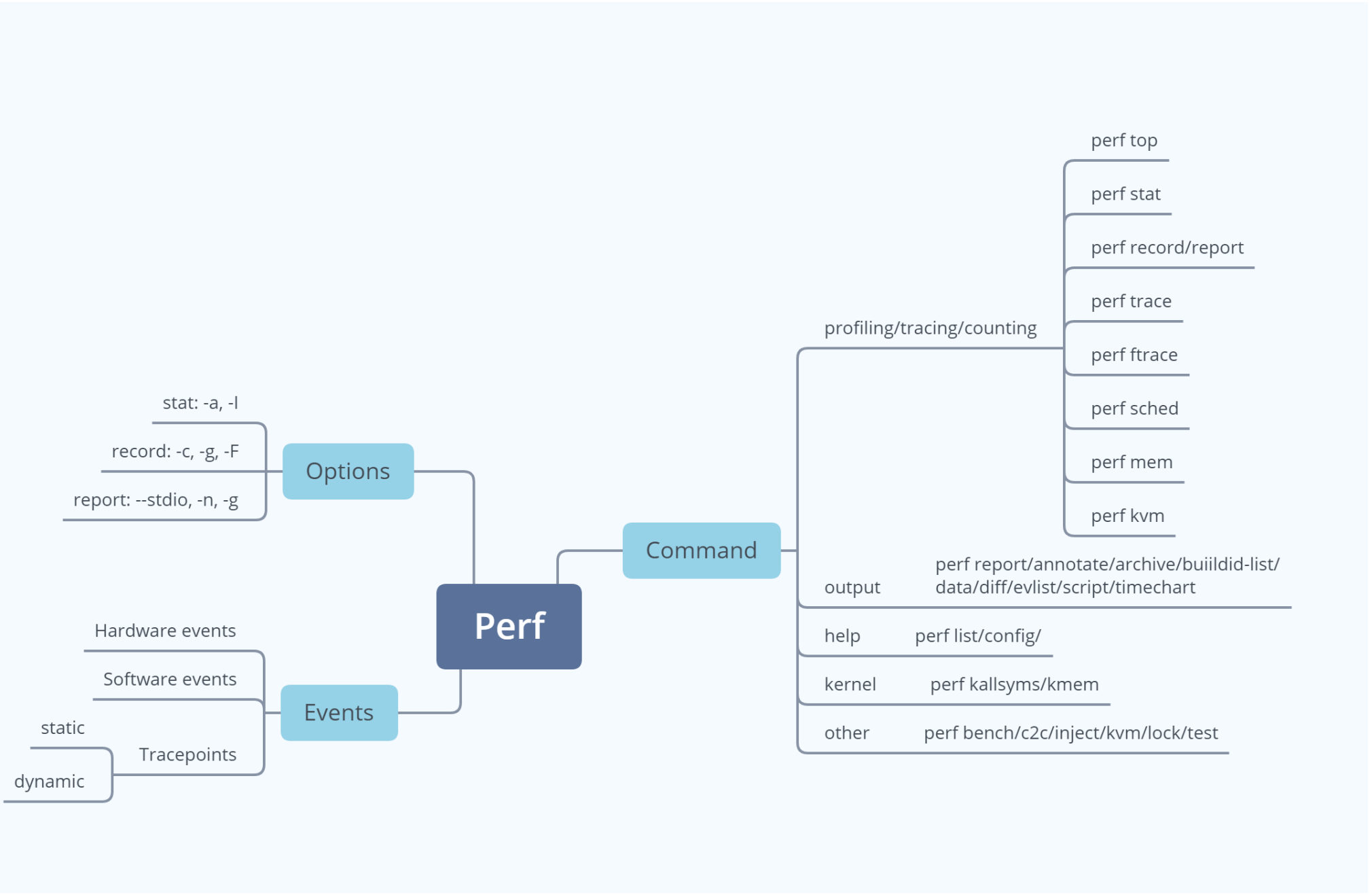
常用perf命令详解

PLCT实验室

2021.05.12

xiaoou@iscas.ac.cn


Perf
Command
profiling/tracing/counting
perf top
perf stat
perf record/report
perf trace
perf ftrace
perf sched
perf mem
perf kvm
output
perf report/annotate/archive/buiildid-list/
data/diff/evlist/script/timechart
help
perf list/config/
kernel
perf kallsyms/kmem
other
perf bench/c2c/inject/kvm/lock/test
Options
stat: -a, -I
record: -c, -g, -F
report: --stdio, -n, -g
Events
Hardware events
Software events
Tracepoints
static
dynamic

Perf stat

- stat means statistic or counting.
  其实就是统计、计数。

- 常用的选项:
  - -a: counting for entire system.
  - -I: Print count deltas every N milliseconds (minimum: 1ms)

- 示例

Perf stat

```
$ sudo perf stat -- sleep 5

 Performance counter stats for 'sleep 5':

              0.80 msec task-clock
                 2      context-switches
                 0      cpu-migrations
                66      page-faults
         1,207,378      cycles
           956,744      instructions
           194,457      branches
             8,301      branch-misses

       5.001793125 seconds time elapsed

       0.001309000 seconds user
       0.000000000 seconds sys
```

```
$ sudo perf stat -a -- sleep 5

 Performance counter stats for 'system wide':

        40,004.73 msec cpu-clock                 #    7.999 CPUs utilized
           24,365      context-switches          #    0.609 K/sec
              725      cpu-migrations            #    0.018 K/sec
            1,898      page-faults               #    0.047 K/sec
    3,164,476,122      cycles                    #    0.079 GHz
    1,154,486,211      instructions              #    0.36  insn per cycle
      227,876,164      branches                  #    5.696 M/sec
       10,232,351      branch-misses             #    4.49% of all branches

       5.001270050 seconds time elapsed
```

Perf stat

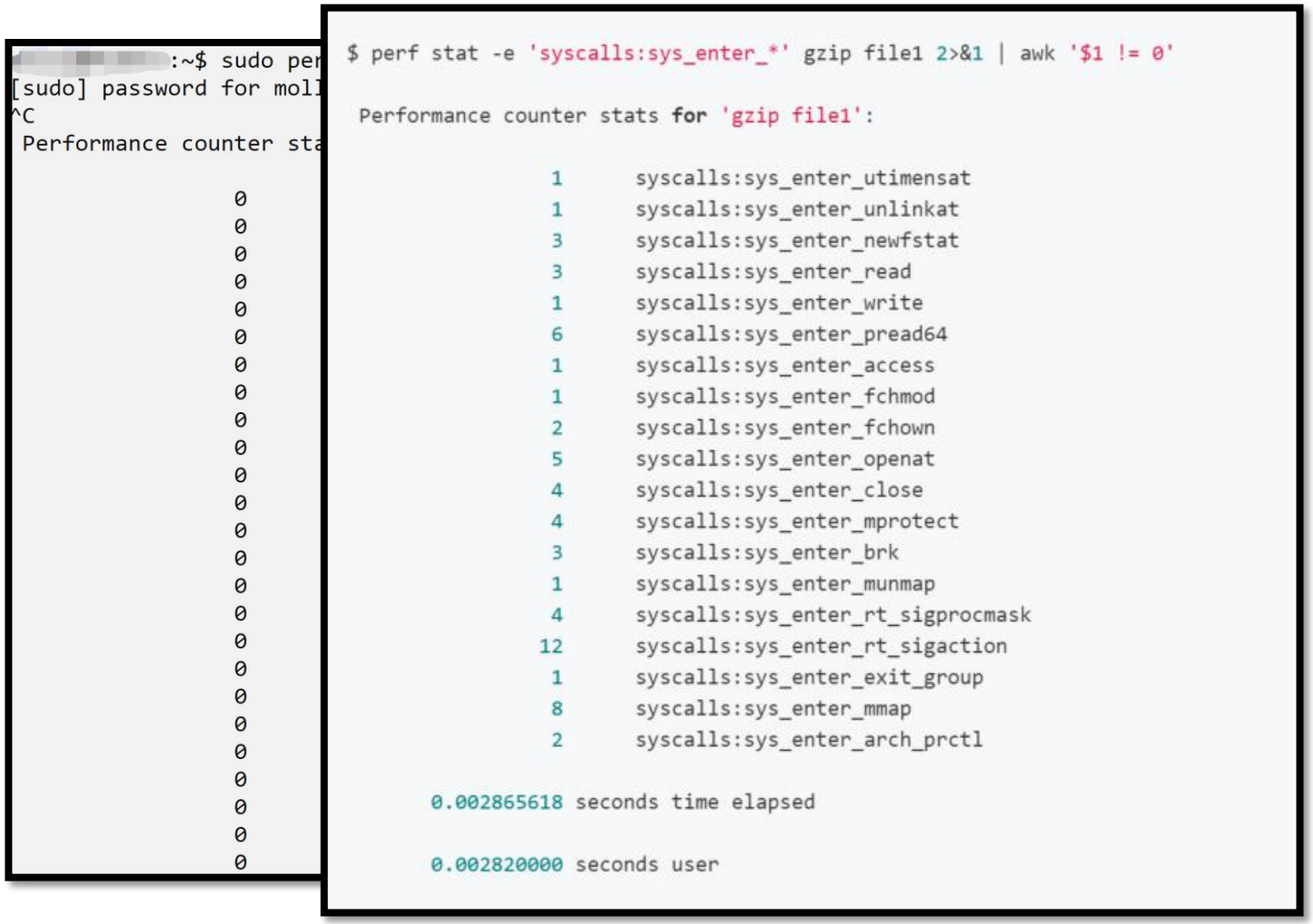
```
$ sudo perf stat -a -I 1000 -- sleep 5
#           time             counts unit events
     1.000839330          8,008.90 msec cpu-clock                 #    8.009 CPUs utilized
     1.000839330             3,403      context-switches          #    0.425 K/sec
     1.000839330                94      cpu-migrations            #    0.012 K/sec
     1.000839330                20      page-faults               #    0.002 K/sec
     1.000839330       210,234,095      cycles                    #    0.026 GHz
     1.000839330        64,704,097      instructions              #    0.31  insn per cycle
     1.000839330        13,586,458      branches                  #    1.696 M/sec
     1.000839330         1,261,647      branch-misses             #    9.29% of all branches
     2.002420701          8,011.78 msec cpu-clock                 #    8.012 CPUs utilized
     2.002420701             3,651      context-switches          #    0.456 K/sec
     2.002420701               149      cpu-migrations            #    0.019 K/sec
     2.002420701                24      page-faults               #    0.003 K/sec
     2.002420701       233,669,364      cycles                    #    0.029 GHz
     2.002420701        86,825,807      instructions              #    0.37  insn per cycle
     2.002420701        18,082,802      branches                  #    2.257 M/sec
     2.002420701         1,391,495      branch-misses             #    7.70% of all branches
     3.003743750          8,011.68 msec cpu-clock                 #    8.012 CPUs utilized
     3.003743750             3,570      context-switches          #    0.446 K/sec
     3.003743750               103      cpu-migrations            #    0.013 K/sec
     3.003743750                15      page-faults               #    0.002 K/sec
     3.003743750       230,822,074      cycles                    #    0.029 GHz
     3.003743750        84,471,781      instructions              #    0.37  insn per cycle
     3.003743750        17,557,372      branches                  #    2.191 M/sec
     3.003743750         1,393,917      branch-misses             #    7.94% of all branches
     4.005320355          8,013.28 msec cpu-clock                 #    8.013 CPUs utilized
```

Perf stat

- 使用通配符来统计一类事件:

  perf stat -e **'syscalls:sys_enter_*'** -p PID

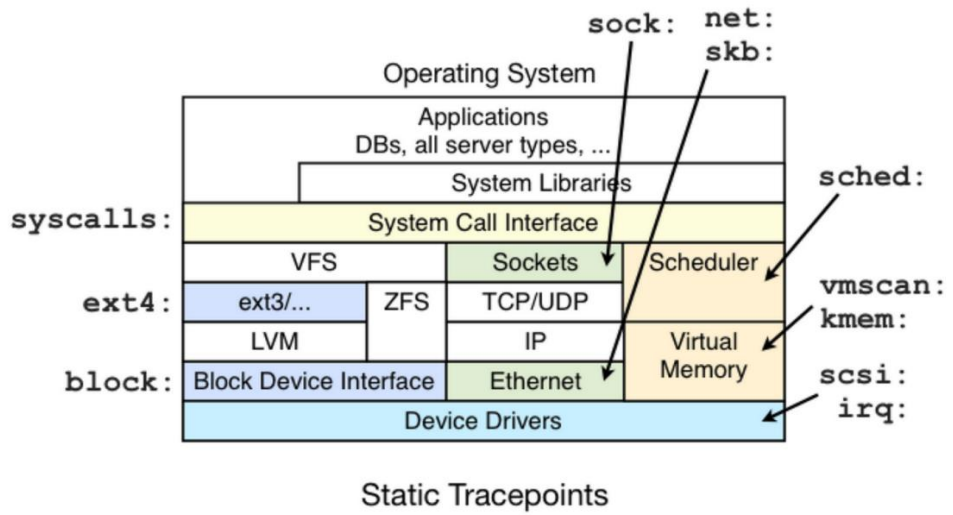
Static tracepoints



Static tracepoint

```
$ perf list | awk -F: '/Tracepoint event/ { lib[$1]++ } END {
>     for (l in lib) { printf "  %-16.16s %d\n", l, lib[l] } }' | sort | column
  alarmtimer     4       gpio           2       iwlwifi_msg    5       page_isolation 1       sock           3
  asoc           13      gvt            14      iwlwifi_ucode  2       pagemap        2       spi            7
  block          18      hda            5       jbd2           17      page_pool      4       swiotlb        1
  bpf_test_run   1       hda_controller 6       kmem           13      percpu         5       sync_trace     1
  bridge         4       hda_intel      4       kvm            76      power          22      syscalls       670
  cfg80211       173     huge_memory    4       kvmmmu         16      printk         1       task           2
  cgroup         13      hwmon          3       libata         6       pwm            2       tcp            7
  clk            16      hyperv         5       mac80211       126     qdisc          4       thermal        5
  compaction     14      i2c            4       mac80211_msg   3       random         15      thermal_power_ 2
  cpuhp          3       i915           40      mce            1       ras            6       timer          13
  cros_ec        2       initcall       3       mdio           1       raw_syscalls   2       tlb            1
  devfreq        1       intel_iommu    7       mei            3       rcu            1       udp            1
  devlink        5       interconnect   2       migrate        1       regmap         15      v4l2           6
  dma_fence      7       iocost         5       mmap           1       regulator      11      vb2            4
  drm            3       iomap          8       mmc            2       resctrl        3       vmscan         18
  exceptions     2       iommu          7       module         5       rpm            5       vsyscall       1
  ext4           105     io_uring       14      msr            3       rseq           2       wbt            4
  fib            1       irq            5       napi           1       rtc            12      workqueue      4
  fib6           1       irq_matrix     12      neigh          7       sched          24      writeback      34
  filelock       12      irq_vectors    34      net            18      scsi           5       x86_fpu        11
  filemap        4       iwlwifi        4       nmi            1       signal         2       xdp            12
  fs_dax         14      iwlwifi_data   2       nvme           4       skb            3       xen            27
  ftrace         2       iwlwifi_io     10      oom            8       smbus          4       xhci-hcd       53
```

- Syscalls: system call enter and exits.
- Ext4: file system events.
- Block: block device events. (硬盘、光盘、U盘等)
- Sock：进程间通信事件
- Sched：CPU调度事件
- Kmem：内核内存分配事件



Static Tracepoints

Perf stat

- Raw PMC counters
  - Using raw PMC counters, eg, counting unhalted core cycles (Core cycles when core is not halted):
  - perf stat -e r003c -a sleep 5 # r == raw mode

- raw PMC counters的编码请查询CPU开发者手册：

1. the Intel 64 and IA-32 Architectures Software Developer's Manual Volume 3B: System Programming Guide, Part 2

- https://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-software-developer-vol-3b-part-2-manual.html

2. the BIOS and Kernel Developer's Guide (BKDG) For AMD Family 10h Processors

- https://www.amd.com/system/files/TechDocs/31116.pdf

Table 18-1. UMask and Event Select Encodings for Pre-Defined Architectural Performance Events

Bit Position CPUID.AH.EBX	Event Name	UMask	Event Select
0	UnHalted Core Cycles	00H	3CH
1	Instruction Retired	00H	C0H
2	UnHalted Reference Cycles	01H	3CH
3	LLC Reference	4FH	2EH
4	LLC Misses	41H	2EH

18-4 Vol. 3B

PERFORMANCE MONITORING

Table 18-1. UMask and Event Select Encodings for Pre-Defined Architectural Performance Events

5	Branch Instruction Retired	00H	C4H
6	Branch Misses Retired	00H	C5H

Perf record

```
# sudo perf list 'block:*'
  block:block_touch_buffer                         [Tracepoint event]
  block:block_dirty_buffer                         [Tracepoint event]
  block:block_rq_abort                             [Tracepoint event]
  block:block_rq_requeue                           [Tracepoint event]
  block:block_rq_complete                          [Tracepoint event]
  block:block_rq_insert                            [Tracepoint event]
  block:block_rq_issue                             [Tracepoint event]
  block:block_bio_bounce                           [Tracepoint event]
  block:block_bio_complete                         [Tracepoint event]
  block:block_bio_backmerge                        [Tracepoint event]
[...]
```

```
$ sudo perf record -e block:block_rq_complete -a sleep 5
[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 1.696 MB perf.data (15 samples) ]

$ sudo perf script
         swapper     0 [003] 1124548.300098: block:block_rq_complete: 259,0 WS () 892332376 + 96 [0]
 glean.dispatche 97501 [000] 1124548.300291: block:block_rq_complete: 259,0 W () 970412512 + 8 [0]
 glean.dispatche 97501 [000] 1124548.300673: block:block_rq_complete: 259,0 W () 970412520 + 8 [0]
         swapper     0 [003] 1124548.300900: block:block_rq_complete: 259,0 FF () 18446744073709551615 + 0 [0]
 glean.dispatche 97501 [000] 1124548.300908: block:block_rq_complete: 259,0 W () 970412528 + 8 [0]
 glean.dispatche 97501 [000] 1124548.301124: block:block_rq_complete: 259,0 W () 970412536 + 8 [0]
         swapper     0 [003] 1124548.301210: block:block_rq_complete: 259,0 WFS () 892332472 + 8 [0]
         swapper     0 [003] 1124548.301217: block:block_rq_complete: 259,0 WFS () 892332472 + 0 [0]
 glean.dispatche 97501 [000] 1124548.301333: block:block_rq_complete: 259,0 W () 970412544 + 8 [0]
 glean.dispatche 97501 [000] 1124548.301534: block:block_rq_complete: 259,0 W () 970412552 + 8 [0]
 glean.dispatche 97501 [000] 1124548.301735: block:block_rq_complete: 259,0 W () 970412560 + 8 [0]
 glean.dispatche 97501 [000] 1124548.301981: block:block_rq_complete: 259,0 W () 970412568 + 8 [0]
 glean.dispatche 97501 [000] 1124548.302188: block:block_rq_complete: 259,0 W () 970412576 + 8 [0]
 glean.dispatche 97501 [000] 1124548.302387: block:block_rq_complete: 259,0 W () 970412584 + 8 [0]
 glean.dispatche 97501 [000] 1124548.302582: block:block_rq_complete: 259,0 W () 970412592 + 8 [0]
```

每列数据分别代表： command, pid, cpu, time, event, storage device major and minor number, type of I/O, block command details, storage device offset + size of I/O (in sectors), errors.

Perf record

- Perf script显示每一列具体字段的含义，需要我们自己猜测验证，可能的字段包括：

Fields: comm,tid,pid,time,cpu,event,**trace**,ip,sym,dso,addr,symoff,srcline,period,iregs,uregs,brstack,brstacksym,flags,bpf-**output**,brstackinsn,brstackoff,callindent,insn,insnlen,synth,phys_addr,metric,misc,ipc

- 使用-F来指定显示的字段
  - sudo perf script -F comm,pid

```
        swapper     0
glean.dispatche 97501
glean.dispatche 97501
        swapper     0
glean.dispatche 97501
glean.dispatche 97501
        swapper     0
        swapper     0
glean.dispatche 97501
glean.dispatche 97501
glean.dispatche 97501
glean.dispatche 97501
glean.dispatche 97501
glean.dispatche 97501
glean.dispatche 97501
```

Perf record

- 跟踪新的进程的创建

```
$ sudo perf record -e sched:sched_process_exec -a
^C[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 2.507 MB perf.data (141 samples) ]

$ sudo perf report -n --sort comm,dso,sym --stdio

# Total Lost Samples: 0
#
# Samples: 141  of event 'sched:sched_process_exec'
# Event count (approx.): 141
#
# Overhead       Samples  Command          Shared Object      Symbol
# ........  ............  ...............  .................  ................
#
    69.50%            98  ip               [kernel.kallsyms]  [k] exec_binprm
     6.38%             9  sh               [kernel.kallsyms]  [k] exec_binprm
     5.67%             8  grep             [kernel.kallsyms]  [k] exec_binprm
     3.55%             5  gpg              [kernel.kallsyms]  [k] exec_binprm
     1.42%             2  cat              [kernel.kallsyms]  [k] exec_binprm
     1.42%             2  nautilus         [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  01-ifupdown      [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  Default          [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  anacron          [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  chrome           [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  dirname          [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  fprintd          [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  fwupdmgr         [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  gdm-session-wor  [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  google-chrome-s  [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  gpu-manager      [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  mkdir            [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  modprobe         [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  nm-dispatcher    [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  prime-switch     [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  readlink         [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  udevadm          [kernel.kallsyms]  [k] exec_binprm
     0.71%             1  which            [kernel.kallsyms]  [k] exec_binprm
```

Perf record

- 采样的频率，通过-F来指定
  (默认频率4000Hz？)

```
sudo perf record -F 31500 -a -- sleep 10

Samples: 137K of event 'cycles', Event count (approx.): 7,442,392,583
```

```
sudo perf record -F 4000 -a -- sleep 10

Samples: 4K of event 'cycles', Event count (approx.): 571,912,377
```

```
sudo perf record -a -- sleep 10

Samples: 6K of event 'cycles', Event count (approx.): 1,095,030,466
```

Perf record

- -c： Event period to sample.
  - 对比perf record -e minor-faults -ag -- sleep 10
    和perf record -e minor-faults -c 1 -ag -- sleep 10
  - 前者： Samples: 65 of event 'minor-faults', Event count (approx.): 3583
  - 后者： Samples: 716 of event 'minor-faults', Event count (approx.): 716
  - 前者不是所有的minor-faults都被采样了，而后者加上-c 1则所有的minor fualts都被跟踪到。

Perf record

- -g：记录函数调用栈

```
$ sudo perf record -e cpu-clock -a -- sleep 5

$ sudo perf report --stdio -n
# To display the perf.data header info, please use --header/--header-or
#
#
# Total Lost Samples: 0
#
# Samples: 31K of event 'cpu-clock'
# Event count (approx.): 7815500000
#
# Overhead       Samples  Command          Shared Object       Symbol
# ........  ............  ...............  ..................  ........
#
    99.92%         31236  swapper          [kernel.kallsyms]   [k] nati
     0.03%             9  swapper          [kernel.kallsyms]   [k] esta
     0.01%             3  client_linux_am  [kernel.kallsyms]   [k] _raw
     0.00%             1  client_linux_am  [kernel.kallsyms]   [k] do_s
     0.00%             1  client_linux_am  [kernel.kallsyms]   [k] fini
     0.00%             1  client_linux_am  client_linux_amd64  [.] 0x00
     0.00%             1  client_linux_am  client_linux_amd64  [.] 0x00
     0.00%             1  client_linux_am  client_linux_amd64  [.] 0x00
     0.00%             1  perf             [kernel.kallsyms]   [k] secu
     0.00%             1  perf             [kernel.kallsyms]   [k] smp_
     0.00%             1  sleep            [kernel.kallsyms]   [k] secu
     0.00%             1  swapper          [kernel.kallsyms]   [k] __sl
     0.00%             1  swapper          [kernel.kallsyms]   [k] __so
     0.00%             1  swapper          [kernel.kallsyms]   [k] find_next_and_bit
     0.00%             1  swapper          [kernel.kallsyms]   [k] finish_task_switch
     0.00%             1  swapper          [kernel.kallsyms]   [k] tick_nohz_idle_exit
     0.00%             1  swapper          [unknown]           [k] 0xffffc90000004003
```

```
$ sudo perf reco

$ sudo perf repo
# To display the
#
#
# Total Lost Sam
#
# Samples: 36K o
# Event count (a
#
# Children
# ........  ....
#
    99.92%     0
            |
            ---s
               |
               |
               |
               |
               |
               |
               |
               |
               |
    99.92%     0
            |
            ---c
               d
               |
               --99.90%--default_idle
                         native_safe_halt
```

```
$ sudo perf report --stdio -n -g folded

# To display the perf.data header info, please use --header/--header-only options.
#
#
# Total Lost Samples: 0
#
# Samples: 36K of event 'cpu-clock'
# Event count (approx.): 9076750000
#
# Children      Self       Samples  Command          Shared Object       Symbol
# ........  ........  ............  ...............  ..................  ........................................
#
    99.92%     0.00%             0  swapper          [kernel.kallsyms]   [k] secondary_startup_64
89.48% secondary_startup_64;start_secondary;cpu_startup_entry;do_idle;default_idle;native_safe_halt
10.41% secondary_startup_64;start_kernel;cpu_startup_entry;do_idle;default_idle;native_safe_halt
    99.92%     0.00%             0  swapper          [kernel.kallsyms]   [k] cpu_startup_entry
99.89% cpu_startup_entry;do_idle;default_idle;native_safe_halt
    99.92%     0.00%             0  swapper          [kernel.kallsyms]   [k] do_idle
99.89% do_idle;default_idle;native_safe_halt
    99.90%     0.00%             0  swapper          [kernel.kallsyms]   [k] default_idle
99.89% default_idle;native_safe_halt
    99.90%    99.89%         36267  swapper          [kernel.kallsyms]   [k] native_safe_halt
89.48% secondary_startup_64;start_secondary;cpu_startup_entry;do_idle;default_idle;native_safe_halt
10.41% secondary_startup_64;start_kernel;cpu_startup_entry;do_idle;default_idle;native_safe_halt
    89.51%     0.00%             0  swapper          [kernel.kallsyms]   [k] start_secondary
89.48% start_secondary;cpu_startup_entry;do_idle;default_idle;native_safe_halt
    10.41%     0.00%             0  swapper          [kernel.kallsyms]   [k] start_kernel
10.41% start_kernel;cpu_startup_entry;do_idle;default_idle;native_safe_halt
     0.03%     0.00%             0  client_linux_am  client_linux_amd64  [.] 0x000000000040ef22
     0.03%     0.00%             0  client_linux_am  client_linux_amd64  [.] 0x00000000004570c3
     0.03%     0.03%            10  client_linux_am  [kernel.kallsyms]   [k] _raw_spin_unlock_irqrestore
```

cpu-clock和cycles事件的区别

cpu-clock	Cycles
软件事件	硬件事件
可以用来表示程序执行经过的真实时间，而无论CPU处于什么状态（Pn（n非0）或者是C状态）	而CPU cycles则用来表示执行程序指令花费的时钟周期数，如果CPU处于Pn（n非0）或者是C状态，则cycles的产生速度会减慢。
如果你想查看哪些代码消耗的真实时间多，则可以使用cpu-clock事件；而如果你想查看哪些代码消耗的时钟周期多，则可以使用CPU cycles事件。	

THE END

THANKS FOR WATCHING