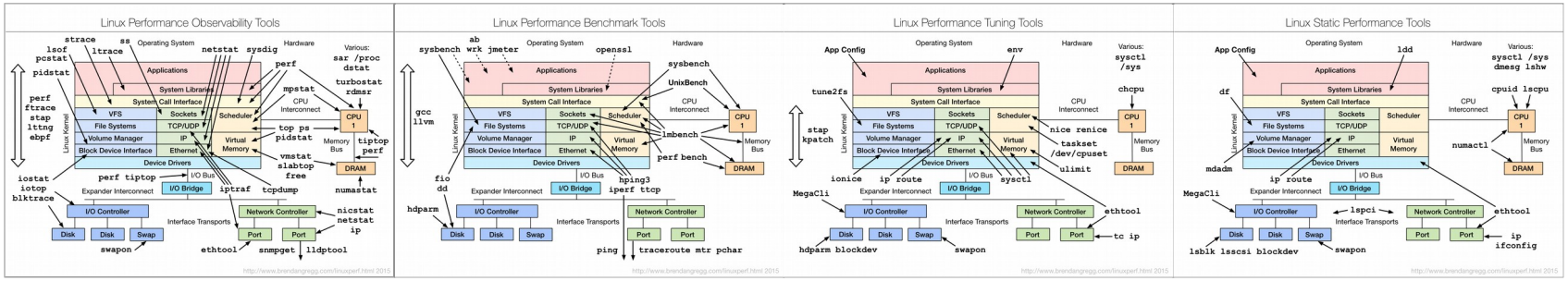
Linux Tools

This Tutorial

- A tour of many Linux performance tools
  - To show you what can be done
  - With guidance for how to do it

Observability Benchmarking Tuning Static Tuning



Agenda

- Methodologies
- Tools
- Tool Types:
  - Observability
  - Benchmarking
  - Tuning
  - Static
- Profiling
- Tracing

Methodologies

Methodologies

- Objectives:
  - Recognize the Streetlight Anti-Method
  - Perform the Workload Characterization Method
  - Perform the USE Method
  - Learn how to start with the questions, before using tools
  - Be aware of other methodologies

Methodologies

- There are dozens of performance tools for Linux
  - Packages: sysstat, procps, coreutils, …
  - Commercial products
- Methodologies can provide guidance for choosing and using tools effectively
- A starting point, a process, and an ending point

ex:1:my sys is slow

```
mohit@nomind:~/Work/UltraLowLatency$ $WORK_HOME/TOOLS/GeneralTools/lab002
```

```
top - 12:47:30 up  3:48,  1 user,  load average: 0.45, 0.27, 0.26
Tasks: 273 total,   1 running, 271 sleeping,   0 stopped,   1 zombie
%Cpu(s):  2.1 us,  1.0 sy,  0.0 ni, 96.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 16338544 total,  5903736 free,  3973800 used,  6461008 buff/cache
KiB Swap: 17576956 total, 17576956 free,        0 used. 11404860 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 1188 root      20   0  249948  99528  71228 S   8.3  0.6   2:40.59 Xorg
 2107 mohit     20   0 2101036 355172  75700 S   8.0  2.2   1:15.51 compiz
 7118 mohit     20   0  628668  31988  26492 S   3.0  0.2   0:00.09 gnome-screensho
 6388 mohit     20   0 1858316 525064 132680 S   1.7  3.2   1:58.55 wpp
 1213 root     -51   0       0      0      0 S   1.3  0.0   0:34.69 irq/40-nvidia
 1917 mohit     20   0  524108  28476  21800 S   0.7  0.2   0:02.42 bamfdaemon
 2001 mohit     20   0  577168  49620  25984 S   0.7  0.3   0:09.32 unity-panel-ser
 3850 mohit     20   0  662252  40588  28012 S   0.7  0.2   0:04.56 gnome-terminal-
```

ex:1:my sys is slow

```
mohit@nomind:~/Work/UltraLowLatency$ $WORK_HOME/TOOLS/GeneralTools/lab002
```

```
Total DISK READ :       0.00 B/s | Total DISK WRITE :       3.89 K/s
Actual DISK READ:       0.00 B/s | Actual DISK WRITE:       0.00 B/s
  TID  PRIO  USER     DISK READ  DISK WRITE  SWAPIN     IO>    COMMAND
  810 be/4 syslog     0.00 B/s    3.89 K/s  0.00 %  0.00 % rsyslogd -n [rs:main Q:Reg]
    1 be/4 root       0.00 B/s    0.00 B/s  0.00 %  0.00 % init splash
    2 be/4 root       0.00 B/s    0.00 B/s  0.00 %  0.00 % [kthreadd]
    3 be/4 root       0.00 B/s    0.00 B/s  0.00 %  0.00 % [ksoftirqd/0]
 2052 be/4 mohit      0.00 B/s    0.00 B/s  0.00 %  0.00 % indicator-messages-service [dconf worker]
    5 be/0 root       0.00 B/s    0.00 B/s  0.00 %  0.00 % [kworker/0:0H]
 2054 be/4 mohit      0.00 B/s    0.00 B/s  0.00 %  0.00 % indicator-session-service [gmain]
    7 be/4 root       0.00 B/s    0.00 B/s  0.00 %  0.00 % [rcu_sched]
    8 be/4 root       0.00 B/s    0.00 B/s  0.00 %  0.00 % [rcu_bh]
```

Anti--Methodologies

- The lack of a deliberate methodology…

Street Light Anti–Method

1. Pick observability tools that are:
   - Familiar
   - Found on the Internet
   - Found at random
2. Run tools
3. Look for obvious issues

Drunk Man Anti–Method

- Tune things at random until the problem goes away

Blame Someone Else Anti–Method

1. Find a system or environment component you are not responsible for
2. Hypothesize that the issue is with that component
3. Redirect the issue to the responsible team
4. When proven wrong, go to 1

Actual Methodologies

- Problem Statement Method
- Workload Characterization Method
- USE Method
- Off-CPU Analysis
- CPU Profile Method
- RTFM Method
- Active Benchmarking *(covered later)*
- Static Performance Tuning *(covered later)*
- …

Problem Statement Method

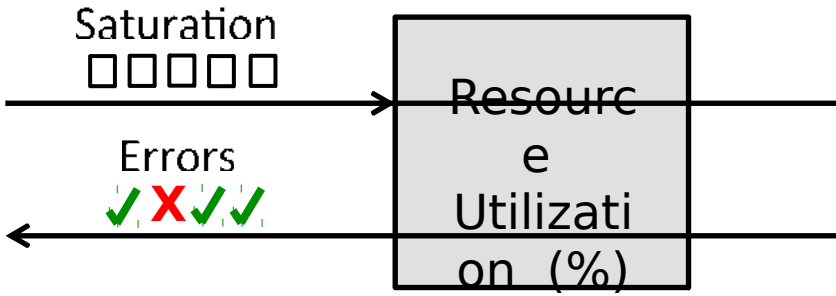
1. What makes you **think** there is a performance problem?
2. Has this system **ever** performed well?
3. What has **changed** recently? (Software? Hardware? Load?)
4. Can the performance degradation be expressed in terms of **latency** or run time?
5. Does the problem affect **other** people or applications (or is it just you)?
6. What is the **environment**? Software, hardware, instance types? Versions? Configuration?

Workload Characterization Method

1. **Who** is causing the load? PID, UID, IP addr, ...
2. **Why** is the load called? code path, stack trace
3. **What** is the load? IOPS, tput, type, r/w
4. **How** is the load changing over time?

The USE Method

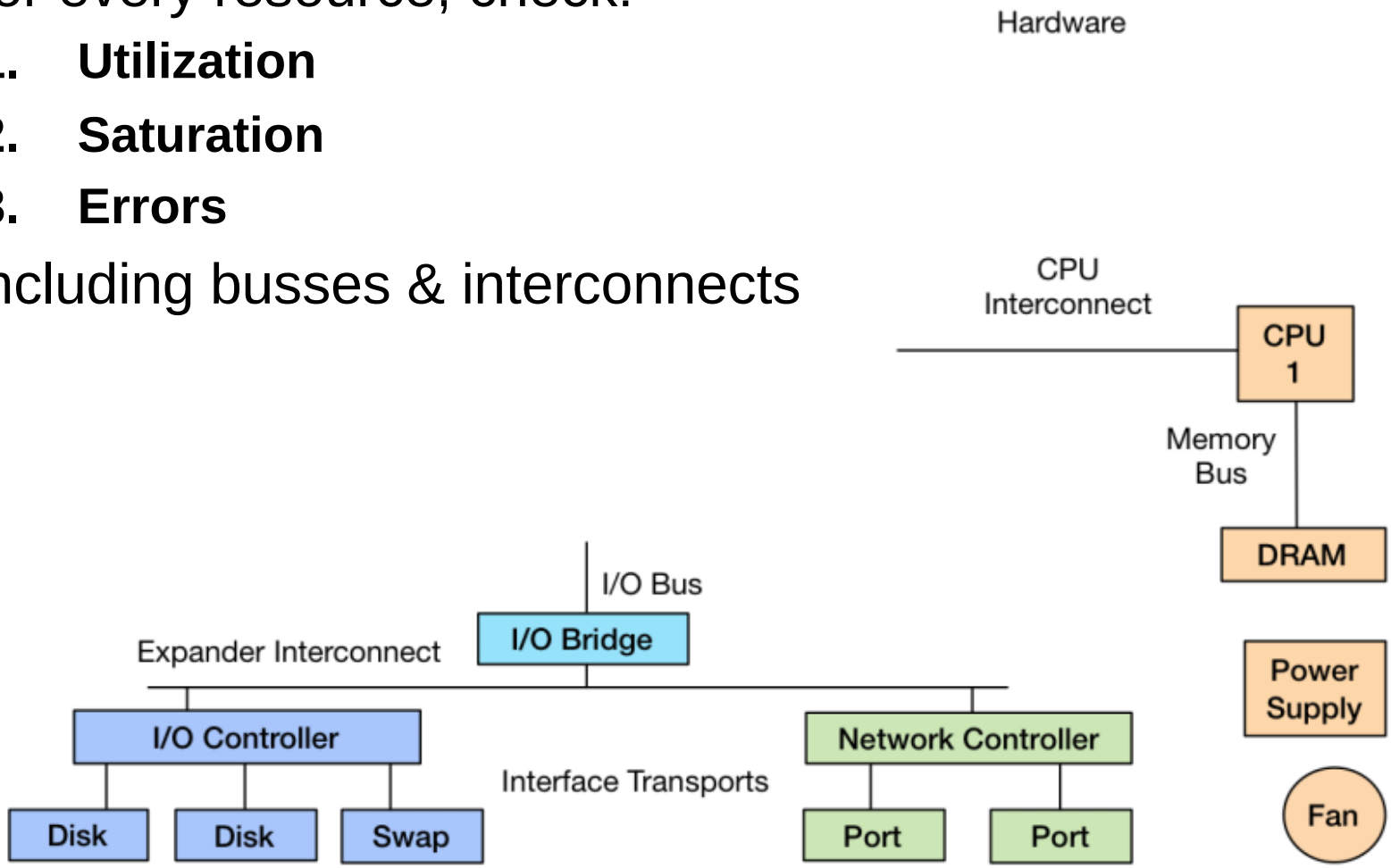
- For every resource, check:
  1. **Utilization**
  2. **Saturation**
  3. **Errors**
- Definitions:
  - Utilization: busy time
  - Saturation: queue length or queued time
  - Errors: easy to interpret (objective)
- Helps if you have a functional (block) diagram of your system / software / environment, showing all resources



Start with the questions, then find the tools

USE Method for Hardware

- For every resource, check:
  1. **Utilization**
  2. **Saturation**
  3. **Errors**
- Including busses & interconnects



Linux USE Method Example

USE Method: Linux Performance Checklist

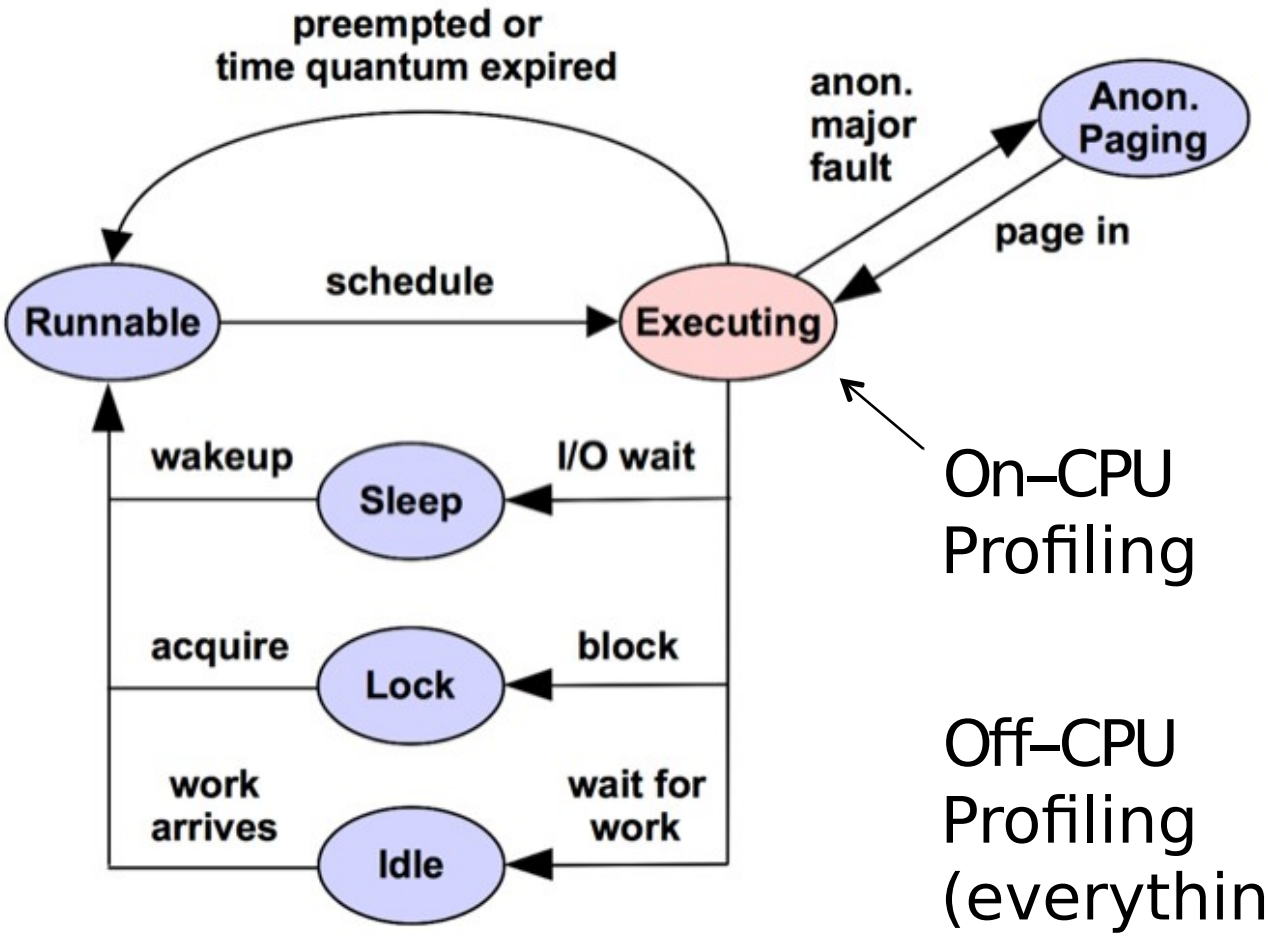
The USE Method provides a strategy for performing a complete check of system health, identifying common bottlenecks and errors. For each system resource, metrics for utilization, saturation and errors are identified and checked. Any issues discovered are then investigated using further strategies.

This is an example USE-based metric list for Linux operating systems (eg, Ubuntu, CentOS, Fedora). This is primarily intended for system administrators of the physical systems, who are using command line tools. Some of these metrics can be found in remote monitoring tools.

Physical Resources

component	type	metric
CPU	utilization	system-wide: `vmstat 1`, "us" + "sy" + "st"; `sar -u`, sum fields except "%idle" and "%iowait"; `dstat -c`, sum fields except "idl" and "wai"; per-cpu: `mpstat -P ALL 1`, sum fields except "%idle" and "%iowait"; `sar -P ALL`, same as `mpstat`; per-process: `top`, "%CPU"; `htop`, "CPU%"; `ps -o pcpu`; `pidstat 1`, "%CPU"; per-kernel-thread: `top`/`htop` ("K" to toggle), where VIRT == 0 (heuristic). [1]
CPU	saturation	system-wide: `vmstat 1`, "r" > CPU count [2]; `sar -q`, "runq-sz" > CPU count; `dstat -p`, "run" > CPU count; per-process: /proc/PID/schedstat 2nd field (sched_info.run_delay); `perf sched latency` (shows "Average" and "Maximum" delay per-schedule); dynamic tracing, eg, SystemTap schedtimes.stp "queued(us)" [3]
CPU	errors	`perf` (LPE) if processor specific error events (CPC) are available; eg, AMD64's "04Ah Single-bit ECC Errors Recorded by Scrubber" [4]
Memory capacity	utilization	system-wide: `free -m`, "Mem:" (main memory), "Swap:" (virtual memory); `vmstat 1`, "free" (main memory), "swap" (virtual memory); `sar -r`, "%memused"; `dstat -m`, "free"; `slabtop -s c` for kmem slab usage; per-process: `top`/`htop`, "RES" (resident main memory), "VIRT" (virtual memory), "Mem" for system-wide summary
Memory capacity	saturation	system-wide: `vmstat 1`, "si"/"so" (swapping); `sar -B`, "pgscank" + "pgscand" (scanning); `sar -W`; per-process: 10th field (min_flt) from /proc/PID/stat for minor-fault rate, or dynamic tracing [5]; OOM killer: `dmesg | grep killed`
Memory	errors	`dmesg` for physical failures; dynamic tracing, eg, SystemTap uprobes for failed malloc()s

Off–CPU Analysis



On–CPU Profiling

Off–CPU Profiling (everything else)

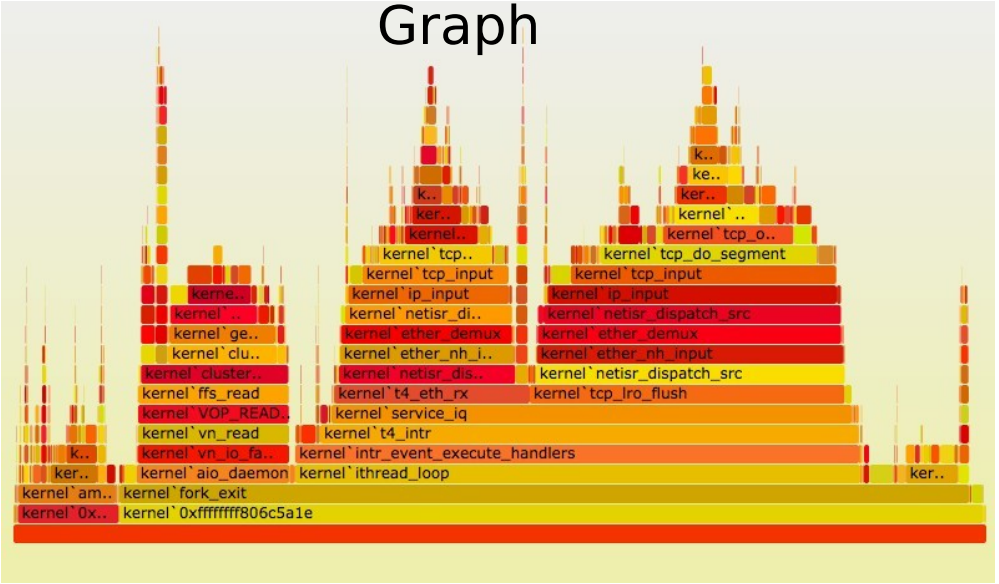
Thread State Transition Diagram

CPU Profile Method

1. Take a CPU profile
2. Understand all software in profile > 1%

- Discovers a wide range of performance issues by their CPU usage
- Narrows software to study

Flame Graph



RTFM Method

- How to understand performance tools or metrics:
  1. Man pages
  2. Books
  3. Web search
  4. Co-workers
  5. Prior talk slides/video (this one!)
  6. Support services
  7. Source code
  8. Experimentation
  9. Social

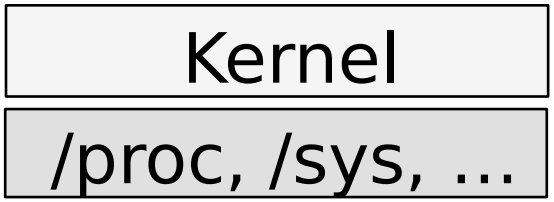
Tools

Tools

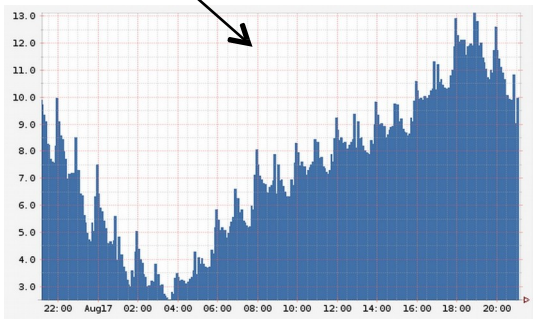
- Objectives:
  - Perform the USE Method for resource utilization
  - Perform Workload Characterization for disks, network
  - Perform the CPU Profile Method using flame graphs
  - Have exposure to various observability tools:
    - Basic: vmstat, iostat, mpstat, ps, top, …
    - Intermediate: tcpdump, netstat, nicstat, pidstat, sar, …
    - Advanced: ss, slaptop, perf_events, …
  - Perform Active Benchmarking
  - Understand tuning risks
  - Perform Static Performance Tuning

Command Line Tools

- Useful to study even if you never use them: GUIs and commercial products often use the same interfaces




```
$ vmstat 1
procs -----------memory---------- ---swap-- …
r  b    swpd   free    buff   cache    si    so …
9  0       0 29549320   29252 9299060    0     …
2  0       0 29547876   29252 9299332    0     …
4  0       0 29548124   29252 9299460    0     …
5  0       0 29548840   29252 9299592    0     …
```

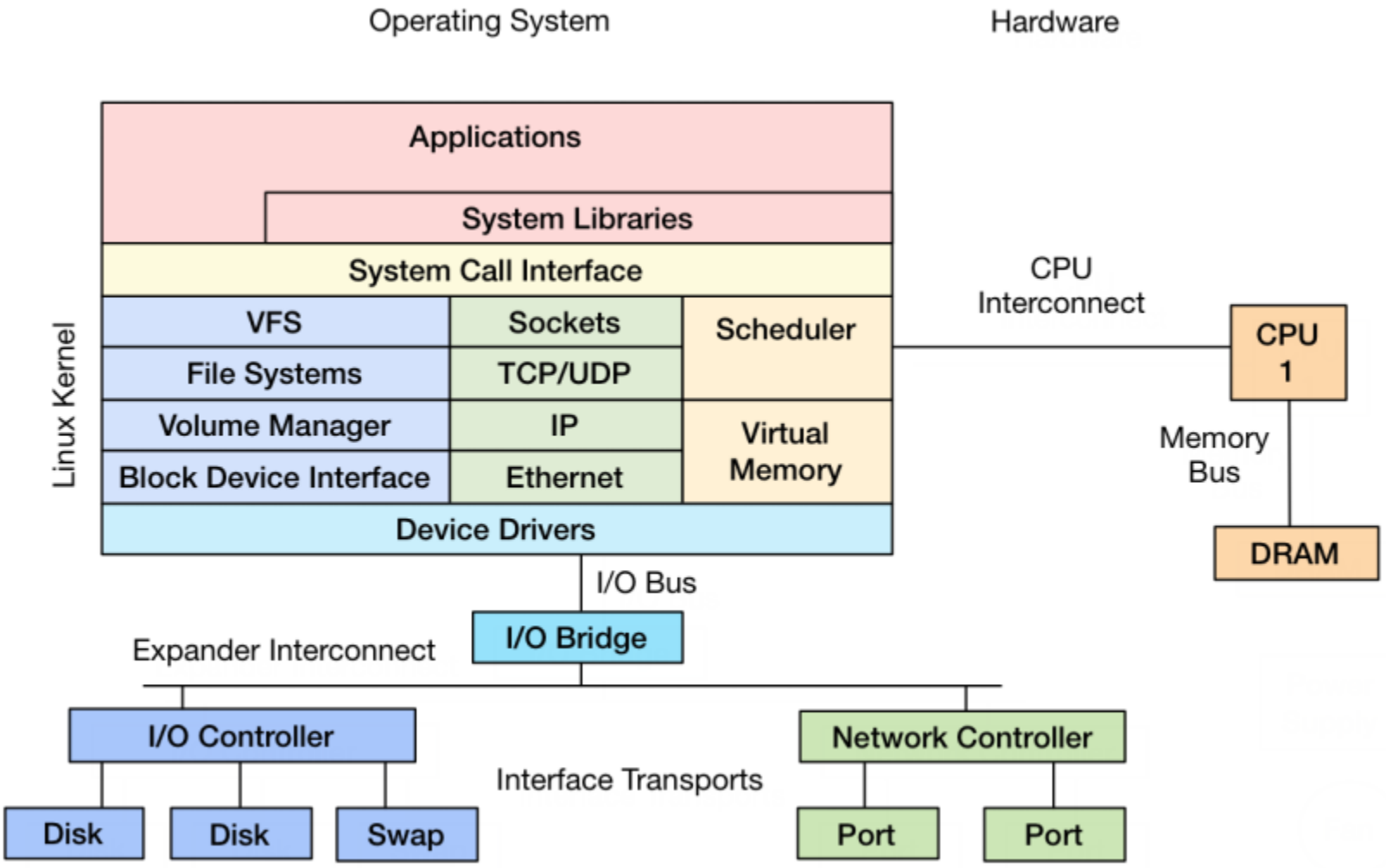


Tool Types

Type	Characteristic
Observability	Watch activity. Safe, usually, depending on resource overhead.
Benchmarking	Load test. Caution: production tests can cause issues due to contention.
Tuning	Change. Danger: changes could hurt performance, now or later with load.
Static	Check configuration. Should be safe.

Observability Tools

How do you measure these?



Observability Tools: Basic

- uptime
- top (or htop)
- ps
- vmstat
- iostat
- mpstat
- free

uptime

- One way to print *load averages*:

```
$ uptime
 07:42:06 up    8:16,  1 user, load average: 2.27, 2.84, 2.91
```

- A measure of resource demand: CPUs + disks
  - Other OSes only show CPUs: easier to interpret
- Exponentially-damped moving averages
- Time constants of 1, 5, and 15 minutes
  - Historic trend without the line graph
- Load > # of CPUs, may mean CPU saturation
  - Don’t spend more than 5 seconds studying these

top (or htop)

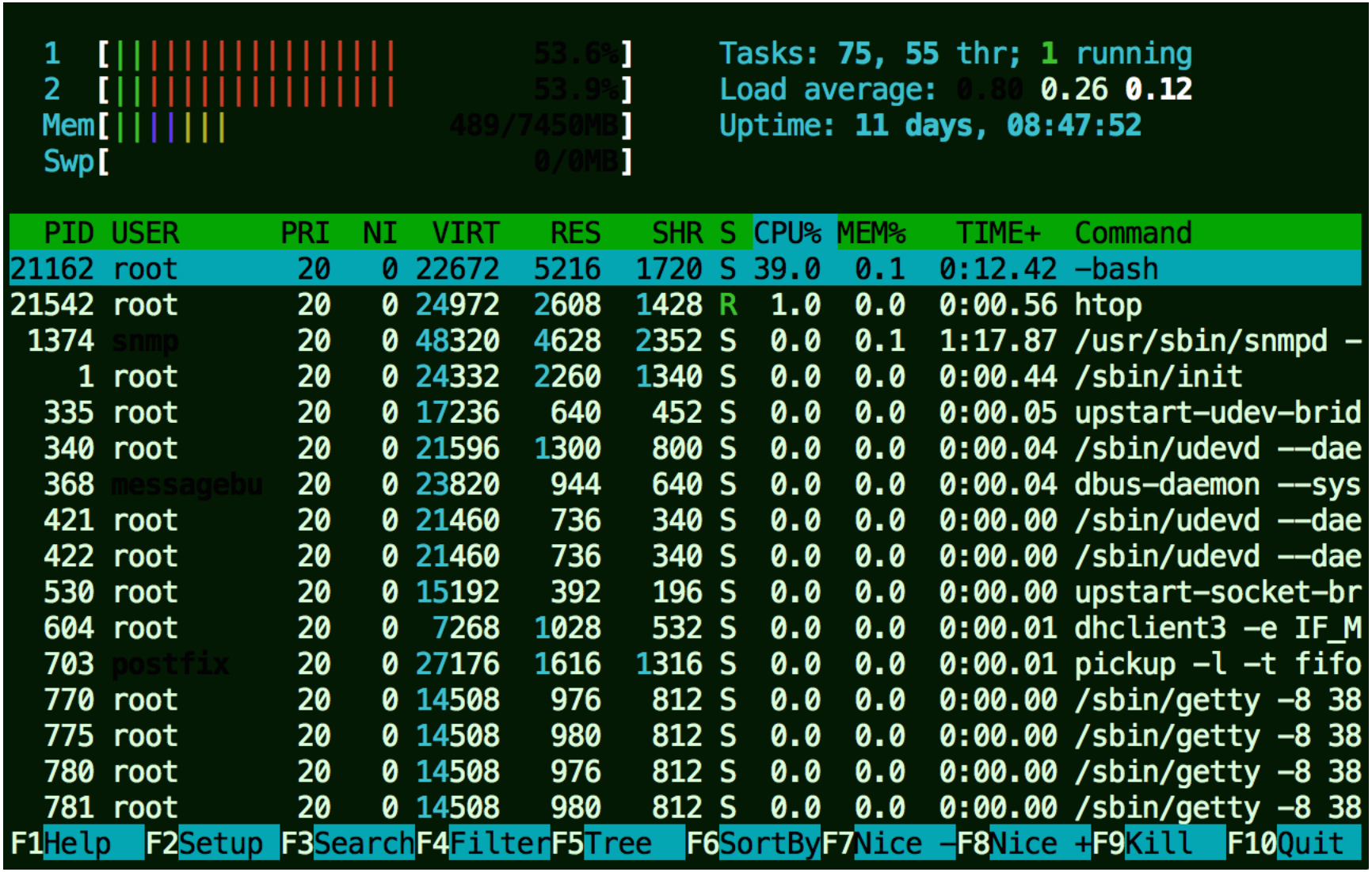
- System and per-process interval summary:

```
top - 12:47:30 up  3:48,  1 user,  load average: 0.45, 0.27, 0.26
Tasks: 273 total,   1 running, 271 sleeping,   0 stopped,   1 zombie
%Cpu(s):  2.1 us,  1.0 sy,  0.0 ni, 96.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 16338544 total,  5903736 free,  3973800 used,  6461008 buff/cache
KiB Swap: 17576956 total, 17576956 free,        0 used. 11404860 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 1188 root      20   0  249948  99528  71228 S   8.3  0.6   2:40.59 Xorg
 2107 mohit     20   0 2101036 355172  75700 S   8.0  2.2   1:15.51 compiz
 7118 mohit     20   0  628668  31988  26492 S   3.0  0.2   0:00.09 gnome-screensho
 6388 mohit     20   0 1858316 525064 132680 S   1.7  3.2   1:58.55 wpp
 1213 root     -51   0       0      0      0 S   1.3  0.0   0:34.69 irq/40-nvidia
 1917 mohit     20   0  524108  28476  21800 S   0.7  0.2   0:02.42 bamfdaemon
 2001 mohit     20   0  577168  49620  25984 S   0.7  0.3   0:09.32 unity-panel-ser
 3850 mohit     20   0  662252  40588  28012 S   0.7  0.2   0:04.56 gnome-terminal-
```

- %CPU is summed across all CPUs
- Can miss short-lived processes (atop won’t)
- Can consume noticeable CPU to read /proc

htop


1 [|||||||||||||||| 53.6%]
2 [|||||||||||||||| 53.9%]
Mem[||||||| 489/7450MB]
Swp[0/0MB]
Tasks: 75, 55 thr; 1 running
Load average: 0.80 0.26 0.12
Uptime: 11 days, 08:47:52
PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
21162 root 20 0 22672 5216 1720 S 39.0 0.1 0:12.42 -bash
21542 root 20 0 24972 2608 1428 R 1.0 0.0 0:00.56 htop
1374 snmp 20 0 48320 4628 2352 S 0.0 0.1 1:17.87 /usr/sbin/snmpd -
1 root 20 0 24332 2260 1340 S 0.0 0.0 0:00.44 /sbin/init
335 root 20 0 17236 640 452 S 0.0 0.0 0:00.05 upstart-udev-brid
340 root 20 0 21596 1300 800 S 0.0 0.0 0:00.04 /sbin/udevd --dae
368 messagebu 20 0 23820 944 640 S 0.0 0.0 0:00.04 dbus-daemon --sys
421 root 20 0 21460 736 340 S 0.0 0.0 0:00.00 /sbin/udevd --dae
422 root 20 0 21460 736 340 S 0.0 0.0 0:00.00 /sbin/udevd --dae
530 root 20 0 15192 392 196 S 0.0 0.0 0:00.00 upstart-socket-br
604 root 20 0 7268 1028 532 S 0.0 0.0 0:00.01 dhclient3 -e IF_M
703 postfix 20 0 27176 1616 1316 S 0.0 0.0 0:00.01 pickup -l -t fifo
770 root 20 0 14508 976 812 S 0.0 0.0 0:00.00 /sbin/getty -8 38
775 root 20 0 14508 980 812 S 0.0 0.0 0:00.00 /sbin/getty -8 38
780 root 20 0 14508 976 812 S 0.0 0.0 0:00.00 /sbin/getty -8 38
781 root 20 0 14508 980 812 S 0.0 0.0 0:00.00 /sbin/getty -8 38
F1Help F2Setup F3SearchF4FilterF5Tree F6SortByF7Nice -F8Nice +F9Kill F10Quit

ps

- Process status listing (eg, “ASCII art forest”):

```
ps -ef f
UID        PID  PPID  C STIME TTY      STAT   TIME CMD
root         2     0  0 08:59 ?        S      0:00 [kthreadd]
root         3     2  0 08:59 ?        S      0:00  \_ [ksoftirqd/0]
root         5     2  0 08:59 ?        S<     0:00  \_ [kworker/0:0H]
root         7     2  0 08:59 ?        S      0:15  \_ [rcu_sched]
root      1167     1  0 08:59 ?        SLsl   0:00 /usr/sbin/lightdm
root      1188  1167  1 08:59 tty7     Ss+    4:50  \_ /usr/lib/xorg/Xorg -core :0 -seat seat0 -auth /var/run/lightdm/root/:0 -nolisten tcp vt7
root      1438  1167  0 08:59 ?        Sl     0:00  \_ lightdm --session-child 12 19
mohit     1759  1438  0 08:59 ?        Ss     0:00      \_ /sbin/upstart --user
mohit     1838  1759  0 08:59 ?        S      0:00         \_ upstart-udev-bridge --daemon --user
mohit     1846  1759  0 08:59 ?        Ss     0:04         \_ dbus-daemon --fork --session --address=unix:abstract=/tmp/dbus-28JQaqarzK
```

- Custom fields:

```
ps -eo user,sz,rss,minflt,majflt,pcpu,args
USER        SZ   RSS MINFLT MAJFLT %CPU COMMAND
root     30023  6248  16246     55  0.0 /sbin/init splash
root         0     0      0      0  0.0 [kthreadd]
root         0     0      0      0  0.0 [ksoftirqd/0]
```

vmstat

- Virtual memory statistics and more:

```
mohit@nomind:~/Work/UltraLowLatency$ vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0      0 5990120 206860 6218784    0    0    16    30   75   50  1  0 98  0  0
mohit@nomind:~/Work/UltraLowLatency$ vmstat 2
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0      0 5990012 206864 6218780    0    0    16    30   75   50  1  0 98  0  0
 0  0      0 5989624 206864 6218784    0    0     0     0  340 1207  0  0 100  0  0
 0  0      0 5989004 206864 6218784    0    0     0   156  339 1292  0  0 100  0  0
 0  0      0 5989840 206864 6218784    0    0     0     0  311 1152  0  0 100  0  0
 0  0      0 5989840 206868 6218780    0    0     0    18  347 1288  0  0 100  0  0
```

- First output line has *some* summary since boot values
  - Should be all; partial is confusing
- High level CPU summary
  - “r” is runnable tasks

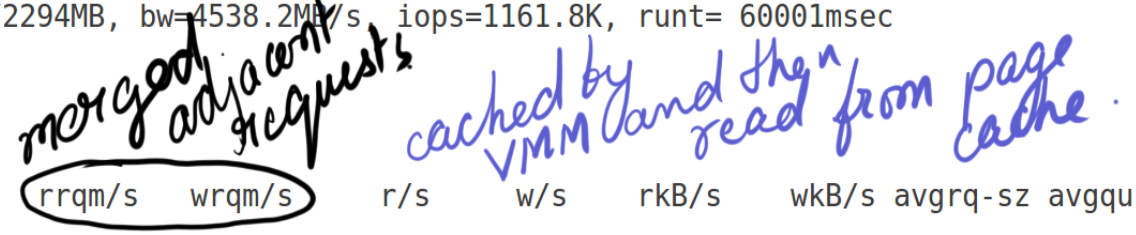
iostat

- Block I/O (disk) stats. 1st output is since boot.

```
mohit@nomind:~/Work/UltraLowLatency$ fio --name=global --rw=read --name=job1 --runtime=60 --time_based --size
job1: (g=0): rw=read, bs=4K-4K/4K-4K/4K-4K, ioengine=sync, iodepth=1
fio-2.2.10
Starting 1 process
Jobs: 1 (f=1): [R(1)] [100.0% done] [4437MB/0KB/0KB /s] [1136K/0/0 iops] [eta 00m:00s]
job1: (groupid=0, jobs=1): err= 0: pid=14259: Mon Apr 17 19:00:01 2017
  read : io=272294MB, bw=4538.2MB/s, iops=1161.8K, runt= 60001msec


Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await
  svctm  %util
sda               0.00     0.00 1378.00    0.00 348024.00     0.00   505.11     0.74    0.53    0.53    0.00
  0.31  43.40
Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await
  svctm  %util
sda               0.00     4.50  697.00    1.50 176264.00    26.00   504.77     0.38    0.55    0.55    1.33
  0.32  22.20
Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await
  svctm  %util
sda               0.00     0.00    0.00    6.00     0.00    28.00     9.33     0.00    0.00    0.00    0.00
  0.00   0.00
Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await
  svctm  %util
sda               0.00     0.00    0.00    1.00     0.00     8.00    16.00     0.00    0.00    0.00    0.00
  0.00   0.00

```

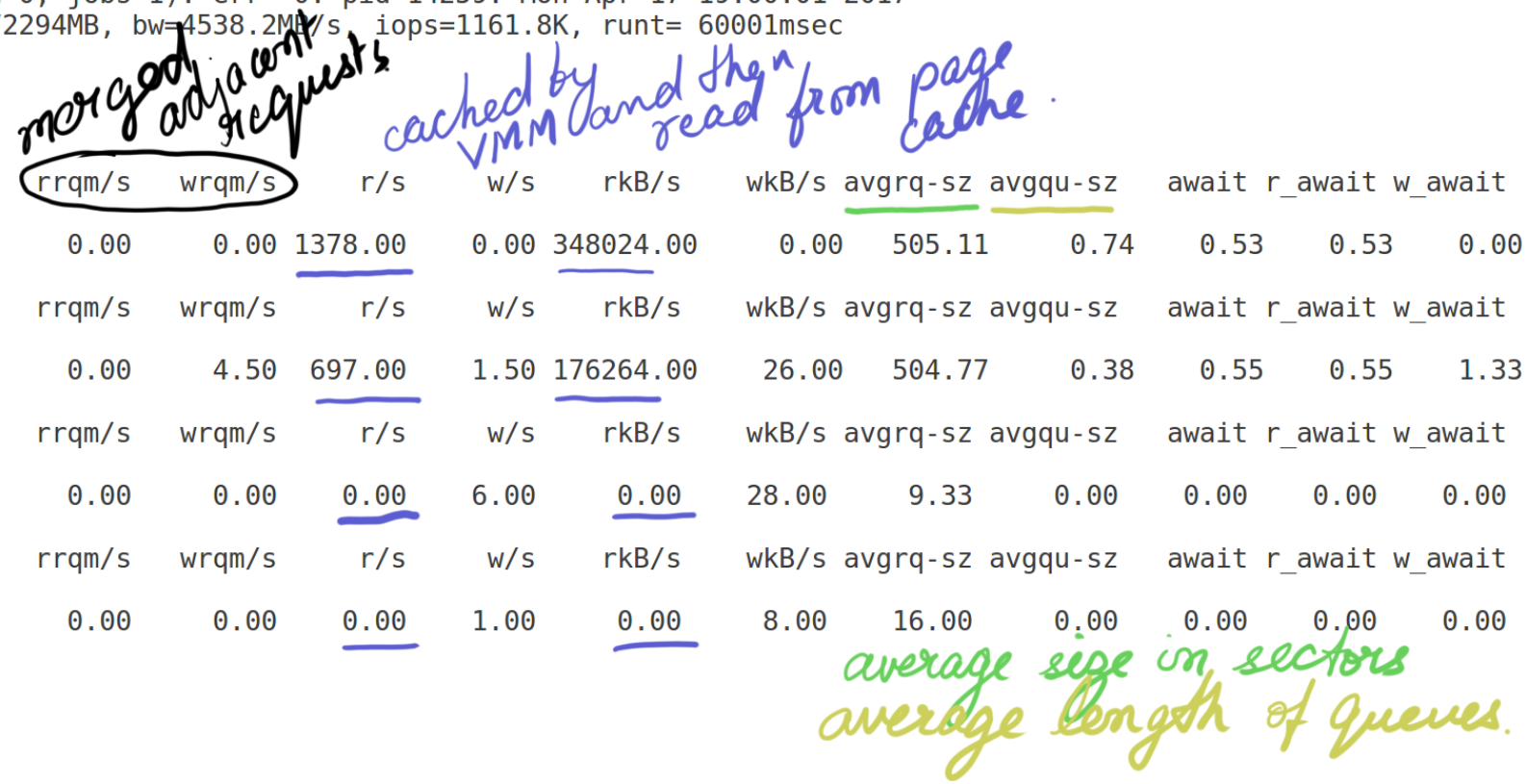


iostat

- Block I/O (disk) stats. 1st output is since boot.

```
mohit@nomind:~/Work/UltraLowLatency$ fio --name=global --rw=read --name=job1 --runtime=60 --time_based --size=1
job1: (g=0): rw=read, bs=4K-4K/4K-4K/4K-4K, ioengine=sync, iodepth=1
fio-2.2.10
Starting 1 process
Jobs: 1 (f=1): [R(1)] [100.0% done] [4437MB/0KB/0KB /s] [1136K/0/0 iops] [eta 00m:00s]
job1: (groupid=0, jobs=1): err= 0: pid=14259: Mon Apr 17 19:00:01 2017
  read : io=272294MB, bw=4538.2MB/s, iops=1161.8K, runt= 60001msec


Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await
  svctm  %util
sda               0.00     0.00 1378.00    0.00 348024.00     0.00   505.11     0.74    0.53    0.53    0.00
  0.31  43.40
Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await
  svctm  %util
sda               0.00     4.50  697.00    1.50 176264.00    26.00   504.77     0.38    0.55    0.55    1.33
  0.32  22.20
Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await
  svctm  %util
sda               0.00     0.00    0.00    6.00     0.00    28.00     9.33     0.00    0.00    0.00    0.00
  0.00   0.00
Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await
  svctm  %util
sda               0.00     0.00    0.00    1.00     0.00     8.00    16.00     0.00    0.00    0.00    0.00
  0.00   0.00
```



iostat

- Block I/O (disk) stats. 1st output is since boot.

```
mohit@nomind:~/Work/UltraLowLatency$ fio --name=global --rw=randread --name=job1 --runtime=60 --time_based --size=1g
job1: (g=0): rw=randread, bs=4K-4K/4K-4K/4K-4K, ioengine=sync, iodepth=1
fio-2.2.10
Starting 1 process
Jobs: 1 (f=1): [r(1)] [100.0% done] [3798MB/0KB/0KB /s] [972K/0/0 iops] [eta 00m:00s]
job1: (groupid=0, jobs=1): err= 0: pid=17090: Mon Apr 17 21:06:26 2017


mohit@nomind:~/Work/UltraLowLatency$ iostat -xkdz 2
Linux 4.4.0-72-generic (nomind)         Monday 17 April 2017    _x86_64_        (8 CPU)

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.15     2.92   28.95    2.87   500.83   241.48    46.66     0.03    0.84    0.19    7.38   0.18   0.59

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     0.00    0.00    0.50     0.00    82.00   328.00     0.00    0.00    0.00    0.00   0.00   0.00

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     6.50    0.00    1.00     0.00    30.00    60.00     0.00    0.00    0.00    0.00   0.00   0.00

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     1.00    0.00    4.00     0.00    20.00    10.00     0.00    0.50    0.00    0.50   0.50   0.20

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     0.00 5569.00    0.00 22276.00     0.00     8.00     0.50    0.09    0.09    0.00   0.09  50.40

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     1.50 9777.00    1.00 39108.00    10.00     8.00     0.91    0.09    0.09    0.00   0.09  91.20

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     0.00 9865.00    0.00 39460.00     0.00     8.00     0.92    0.09    0.09    0.00   0.09  92.20

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     2.00 9799.00    1.50 39196.00    16.00     8.00     0.90    0.09    0.09    1.33   0.09  90.00
```

mpstat

- Multi-processor statistics, per-CPU:

```
  $ mpstat –P ALL 1
[…]
08:06:43   PM    CPU    %usr   %nice    %sys   %iowait    %irq   %soft  %steal  %guest
 08:06:44  PM    all   53.45    0.00    3.77      0.00    0.00    0.39    0.13    0.00
 08:06:44  PM      0   49.49    0.00    3.03      0.00    0.00    1.01    1.01    0.00
 08:06:44  PM      1   51.61    0.00    4.30      0.00    0.00    2.15    0.00    0.00
 08:06:44  PM      2   58.16    0.00    7.14      0.00    0.00    0.00    1.02    0.00
 08:06:44  PM      3   54.55    0.00    5.05      0.00    0.00    0.00    0.00    0.00
 08:06:44  PM      4   47.42    0.00    3.09      0.00    0.00    0.00    0.00    0.00
 08:06:44  PM      5   65.66    0.00    3.03      0.00    0.00    0.00    0.00    0.00
08:06:44   PM      6   50.00    0.00    2.08      0.00    0.00    0.00    0.00    0.00
[…]
```

- Look for unbalanced workloads, hot CPUs.

free

- Main memory usage:

```
mohit@nomind:~/Work/UltraLowLatency$ free -m
              total        used        free      shared  buff/cache   available
Mem:          15955        4473        4940         509        6541       10579
Swap:         17164           0       17164
```

- buffers: block device I/O cache
- cached: virtual page cache
- total=used+free+buff/cache

ex:1:Latency is now much higher...

```
mohit@nomind:~/Work/UltraLowLatency$ $WORK_HOME/TOOLS/GeneralTools/lab005
```

```
mohit@nomind:~/Work/UltraLowLatency$ top

top - 22:28:16 up 13:28,  1 user,  load average: 1.20, 1.04, 0.78
Tasks: 292 total,   1 running, 290 sleeping,   0 stopped,   1 zombie
%Cpu(s):  0.3 us,  0.9 sy,  0.0 ni, 90.9 id,  7.9 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 16338544 total,  4813356 free,  4671124 used,  6854064 buff/cache
KiB Swap: 17576956 total, 17576956 free,        0 used. 10739968 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
17965 mohit     20   0    4224    628    552 D   3.3  0.0   0:13.56 lab005
  194 root      20   0       0      0      0 D   2.0  0.0   0:11.02 jbd2/sda4-8
 3101 mohit     20   0 1363624 532872 338636 S   1.0  3.3   1:13.05 chrome
 5625 mohit     20   0 1107940 268672  83444 S   1.0  1.6   5:45.68 chrome
  296 root       0 -20       0      0      0 S   0.7  0.0   0:02.98 kworker/5:1H
 1188 root      20   0  570920 165388 130568 S   0.7  1.0  14:44.17 Xorg
 1213 root     -51   0       0      0      0 S   0.7  0.0   3:33.16 irq/40-nvidia
```

ex:1:Latency is now much higher...

```
mohit@nomind:~/Work/UltraLowLatency$ $WORK_HOME/TOOLS/GeneralTools/lab005
```

```
mohit@nomind:~/Work/UltraLowLatency$ vmstat 2
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  1      0 4814316 332552 6521520    0    0    59    40   83   70  2  1 97  0  0
 0  1      0 4814224 332552 6521520    0    0     0  8408 2556 7365  1  1 92  7  0
 1  1      0 4814100 332552 6521520    0    0     0  6120 1938 5463  0  1 90  8  0
 0  1      0 4814100 332552 6521520    0    0     0  8190 2481 6993  1  1 92  7  0
 0  1      0 4814068 332552 6521520    0    0     0  8104 2497 7111  1  1 91  7  0
^C
mohit@nomind:~/Work/UltraLowLatency$ mpstat -P ALL 2
Linux 4.4.0-72-generic (nomind)         Monday 17 April 2017    _x86_64_        (8 CPU)

10:29:09  IST  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
10:29:11  IST  all    0.38    0.00    0.32    8.82    0.00    0.00    0.00    0.00    0.00   90.49
10:29:11  IST    0    1.01    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   98.99
10:29:11  IST    1    0.00    0.00    0.00    2.02    0.00    0.00    0.00    0.00    0.00   97.98
10:29:11  IST    2    0.00    0.00    0.51   38.07    0.00    0.00    0.00    0.00    0.00   61.42
10:29:11  IST    3    0.50    0.00    0.50    1.50    0.00    0.00    0.00    0.00    0.00   97.50
10:29:11  IST    4    1.01    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   98.99
10:29:11  IST    5    0.00    0.00    0.52   30.73    0.00    0.00    0.00    0.00    0.00   68.75
10:29:11  IST    6    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
10:29:11  IST    7    0.50    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   99.50

10:29:11  IST  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
10:29:13  IST  all    0.44    0.00    0.88    6.63    0.00    0.00    0.00    0.00    0.00   92.04
10:29:13  IST    0    1.00    0.00    1.00    1.00    0.00    0.00    0.00    0.00    0.00   97.00
10:29:13  IST    1    0.51    0.00    0.51    5.10    0.00    0.00    0.00    0.00    0.00   93.88
10:29:13  IST    2    0.51    0.00    0.51   19.80    0.00    0.00    0.00    0.00    0.00   79.19
10:29:13  IST    3    0.51    0.00    1.52    1.01    0.00    0.00    0.00    0.00    0.00   96.97
10:29:13  IST    4    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
10:29:13  IST    5    0.00    0.00    2.62   26.70    0.00    0.00    0.00    0.00    0.00   70.68
10:29:13  IST    6    0.00    0.00    0.50    0.00    0.00    0.00    0.00    0.00    0.00   99.50
10:29:13  IST    7    1.00    0.00    0.50    0.00    0.00    0.00    0.00    0.00    0.00   98.50
^C
```

ex:1:Latency is now much higher...

```
mohit@nomind:~/Work/UltraLowLatency$ $WORK_HOME/TOOLS/GeneralTools/lab005
```

```
mohit@nomind:~/Work/UltraLowLatency$ sar -n DEV 1
Linux 4.4.0-72-generic (nomind)         Monday 17 April 2017    _x86_64_        (8 CPU)

10:31:04  IST     IFACE   rxpck/s   txpck/s    rxkB/s    txkB/s   rxcmp/s   txcmp/s  rxmcst/s   %ifutil
10:31:05  IST    wlp3s0      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
10:31:05  IST        lo      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00

10:31:05  IST     IFACE   rxpck/s   txpck/s    rxkB/s    txkB/s   rxcmp/s   txcmp/s  rxmcst/s   %ifutil
10:31:06  IST    wlp3s0      1.00      1.00      0.10      0.09      0.00      0.00      0.00      0.00
10:31:06  IST        lo      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
```

ex:1:Latency is now much higher...

```
mohit@nomind:~/Work/UltraLowLatency$ $WORK_HOME/TOOLS/GeneralTools/lab005
```

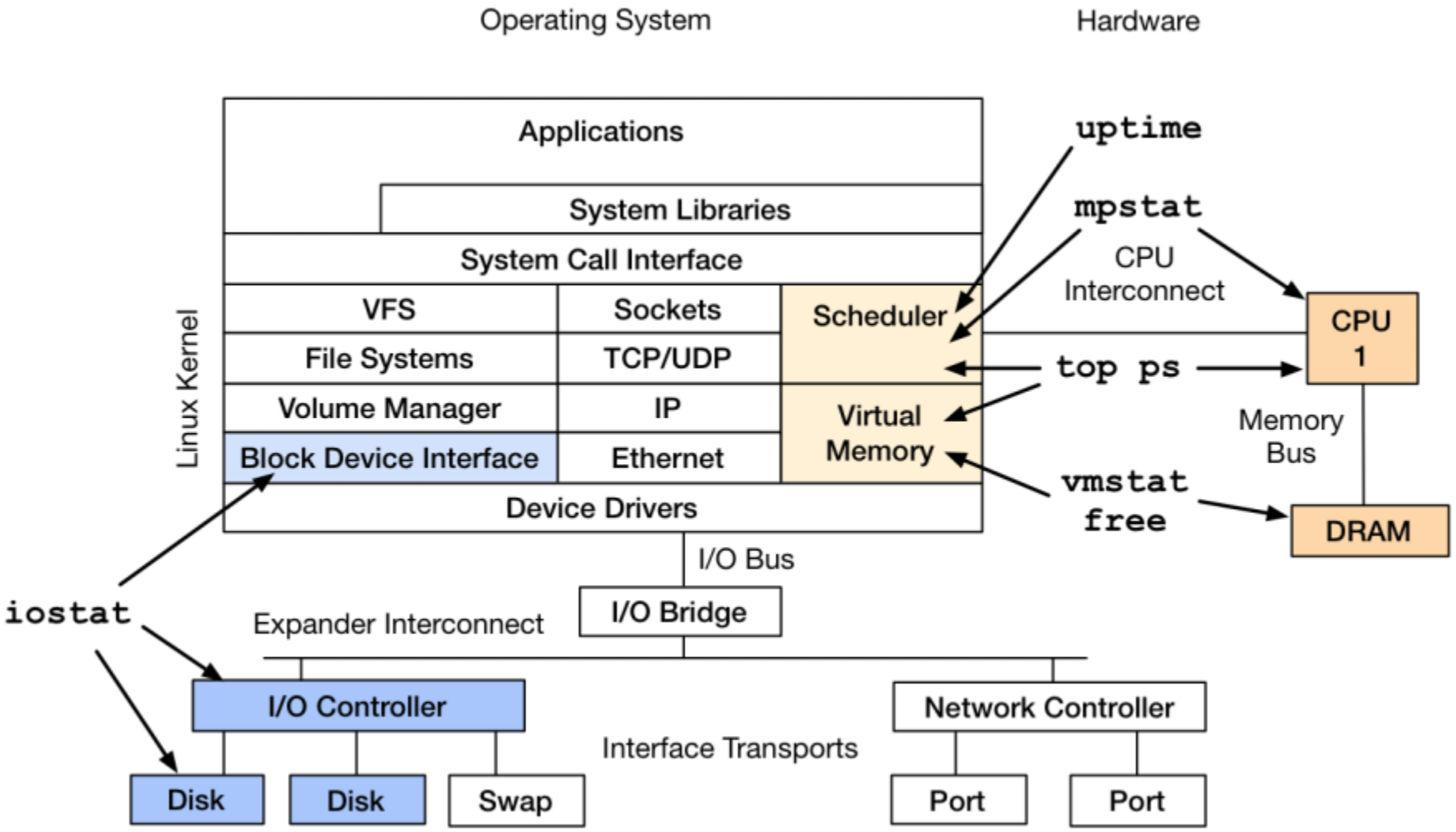
```
mohit@nomind:~/Work/UltraLowLatency$ iostat -dxm 2
Linux 4.4.0-72-generic (nomind)         Monday 17 April 2017    _x86_64_        (8 CPU)

Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.14     6.45   31.22   25.23     0.46     0.34    29.08     0.04    0.72    0.17    1.41   0.36   2.05

Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00   250.00    0.00 1540.00     0.00     7.97    10.60     0.92    0.60    0.00    0.60   0.59  91.60

Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00   250.00    0.00 1567.00     0.00     8.07    10.55     0.95    0.60    0.00    0.60   0.60  94.60
```

Observability Tools: Basic



Observability Tools: Intermediate

- strace
- tcpdump
- netstat
- nicstat
- pidstat
- swapon
- lsof
- sar (and collectl, dstat, etc.)

strace

- System call tracer:

```
mohit@nomind:~/Work/UltraLowLatency$ sudo strace -tttT -p `/bin/ps -fu $USER| grep "lab005" | grep -v "grep" | awk '{print $2}'` 2>&1 | head
-10
strace: Process 23246 attached
1492486478.447812 write(3, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 8192) = 8192 <0.001012>
1492486478.448855 write(3, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 8192) = 8192 <0.002462>
1492486478.451351 write(3, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 8192) = 8192 <0.000981>
1492486478.452407 write(3, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 8192) = 8192 <0.002277>
1492486478.454797 write(3, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 8192) = 8192 <0.001006>
1492486478.455922 write(3, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 8192) = 8192 <0.000996>
1492486478.457036 write(3, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 8192) = 8192 <0.002207>
1492486478.459280 write(3, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 8192) = 8192 <0.001031>
1492486478.460395 write(3, "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0"..., 8192) = 8192 <0.002095>
```

- Eg, -ttt: time (us) since epoch; -T: syscall time (s)
- Translates syscall args
  - Very helpful for solving system usage issues
- Currently has massive overhead (ptrace based)
  - Can slow the target by > 100x. Use extreme caution.

strace:substitutes

- System call tracer:

```
sudo strace -tttT -p `/bin/ps -fu $USER| grep "lab005" | grep -v "grep" | awk '{print $2}'` 2>&1 | head -10
sudo perf stat -e 'syscalls:sys_enter_*' -p `/bin/ps -fu $USER| grep "lab005" | grep -v "grep" | awk '{print $2}'`
sudo perf trace -p `/bin/ps -fu $USER| grep "lab005" | grep -v "grep" | awk '{print $2}'` 2>&1 | head -10
sudo stap -v $WORK_HOME/resources/systemtap/testsuite/systemtap.examples/process/strace.stp -x `/bin/ps -fu $USER| grep "lab005" | grep -v
"grep" | awk '{print $2}'` 2>&1 | head -10
```

tcpdump

- Sniff network packets for post analysis:

```
 1 mohit@nomind:~/Work/UltraLowLatency$ sudo tcpdump -i wlp3s0 -w /tmp/out.tcpdump
 2 tcpdump: listening on wlp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
 3 ^C48 packets captured
 4 48 packets received by filter
 5 0 packets dropped by kernel
 6
 7 mohit@nomind:~/Work/UltraLowLatency$ sudo tcpdump -nr /tmp/out.tcpdump | head
 8 reading from file /tmp/out.tcpdump, link-type EN10MB (Ethernet)
 9 09:16:33.064156 ARP, Request who-has 192.168.1.1 tell 192.168.1.139, length 28
10 09:16:33.064894 ARP, Reply 192.168.1.1 is-at 94:d7:23:05:07:ce, length 46
11 09:16:33.423900 IP 192.168.1.238.37721 > 192.168.1.255.32412: UDP, length 21
12 09:16:33.425021 IP 192.168.1.238.56308 > 192.168.1.255.32414: UDP, length 21
13 09:16:36.856164 IP 192.168.1.139.56620 > 74.125.68.188.5228: Flags [.], ack 314195072
   2488120513], length 0
14 09:16:36.939251 IP 74.125.68.188.5228 > 192.168.1.139.56620: Flags [.], ack 1, win 35
   length 0
15 09:16:38.443099 IP 192.168.1.238.37721 > 192.168.1.255.32412: UDP, length 21
16 09:16:38.444253 IP 192.168.1.238.56308 > 192.168.1.255.32414: UDP, length 21
17 09:16:38.760233 IP 216.58.197.69.443 > 192.168.1.139.41144: UDP, length 42
18 09:16:38.786173 IP 192.168.1.139.41144 > 216.58.197.69.443: UDP, length 39
19
```

- Study packet sequences with timestamps (us)
- CPU overhead optimized (socket ring buffers), but can still be significant. Use caution.

netstat

- Various network protocol statistics using -s:
- A multi-tool:
  - -i: interface stats
  - -r: route table
  - default: list conns
- netstat -p: shows process details!
- Per-second interval with -c

```
netstat -s
Ip:
    376734 total packets received
    0 forwarded
    0 incoming packets discarded
    376606 incoming packets delivered
    232950 requests sent out
    44 outgoing packets dropped
Icmp:
    105 ICMP messages received
    2 input ICMP message failed.
    ICMP input histogram:
        destination unreachable: 105
    122 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        destination unreachable: 122
IcmpMsg:
        InType3: 105
        OutType3: 122
```

nicstat

- Network interface stats, iostat-like output:

```
mohit@nomind:~/Work/UltraLowLatency$ nicstat 1
    Time      Int   rKB/s   wKB/s   rPk/s   wPk/s    rAvs    wAvs %Util    Sat
09:31:42   wlp3s0    4.29    0.34    4.14    2.48  1061.3   139.4  0.00   0.00
09:31:42       lo    0.01    0.01    0.20    0.20   67.95   67.95  0.00   0.00
    Time      Int   rKB/s   wKB/s   rPk/s   wPk/s    rAvs    wAvs %Util    Sat
09:31:43   wlp3s0    0.00    0.00    0.00    0.00    0.00    0.00  0.00   0.00
09:31:43       lo    0.00    0.00    0.00    0.00    0.00    0.00  0.00   0.00
    Time      Int   rKB/s   wKB/s   rPk/s   wPk/s    rAvs    wAvs %Util    Sat
09:31:44   wlp3s0    0.12    0.00    2.00    0.00   63.00    0.00  0.00   0.00
09:31:44       lo    0.00    0.00    0.00    0.00    0.00    0.00  0.00   0.00
```

- Check network throughput and interface %util
- Designed such that USE can be applied

pidstat

- Very useful process stats. eg, by-thread, disk I/O:

```
mohit@nomind:~/Work/UltraLowLatency$ pidstat -t 1
Linux 4.4.0-72-generic (nomind)         Tuesday 18 April 2017   _x86_64_        (8 CPU)

09:38:37  IST   UID      TGID       TID    %usr %system  %guest    %CPU   CPU  Command
09:38:38  IST   127       939         -    0.97    0.00    0.00    0.97     7  influxd
09:38:38  IST   125      1068         -    0.00    0.97    0.00    0.97     2  redis-server
09:38:38  IST   125         -      1068    0.00    0.97    0.00    0.97     2  |__redis-server
09:38:38  IST  1000      2092         -    0.97    0.00    0.00    0.97     1  pulseaudio
09:38:38  IST  1000         -      2092    0.97    0.97    0.00    1.94     1  |__pulseaudio
09:38:38  IST  1000         -      2246    0.97    0.00    0.00    0.97     0  |__alsa-sink-CS420
09:38:38  IST  1000      5625         -    1.94    0.00    0.00    1.94     0  chrome
09:38:38  IST  1000         -      5625    0.97    0.00    0.00    0.97     0  |__chrome
09:38:38  IST  1000     14600         -    0.97    0.00    0.00    0.97     7  compiz
09:38:38  IST  1000     18733         -    1.94    0.00    0.00    1.94     1  chrome
09:38:38  IST  1000     26174         -    0.97    1.94    0.00    2.91     4  pidstat
09:38:38  IST  1000         -     26174    0.97    1.94    0.00    2.91     4  |__pidstat


mohit@nomind:~/Work/UltraLowLatency$ pidstat -d 1
Linux 4.4.0-72-generic (nomind)         Tuesday 18 April 2017   _x86_64_        (8 CPU)
09:39:24  IST   UID       PID   kB_rd/s   kB_wr/s kB_ccwr/s iodelay  Command
09:39:25  IST  1000      2690      0.00      8.00      0.00       0  chrome
```

- I usually prefer this over top(1)

swapon

- Show swap device usage:

```
mohit@nomind:~/Work/UltraLowLatency$ swapon
NAME      TYPE      SIZE USED PRIO
/dev/sda5 partition 16.8G   0B   -1
```

- If you have swap enabled…

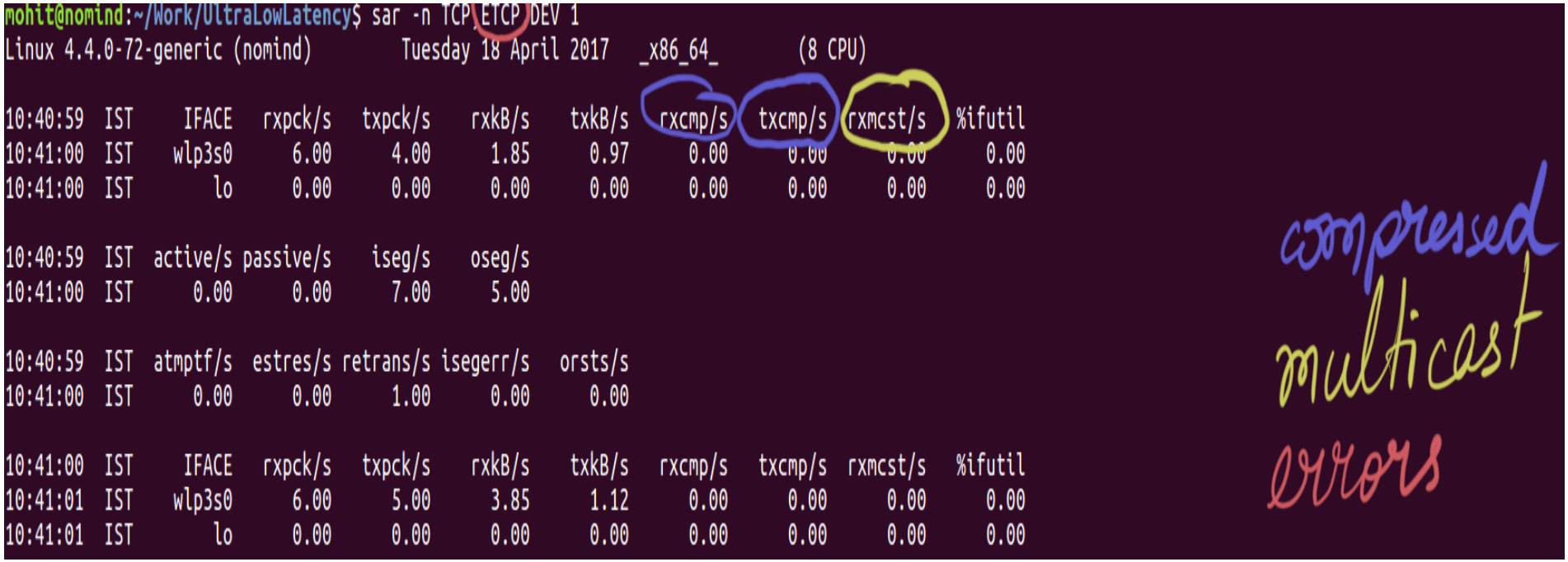
lsof

- More a debug tool, lsof(8) shows file descriptor usage, which for some apps, equals current active network connections:

```
mohit@nomind:~/Work/UltraLowLatency$ lsof -iTCP -sTCP:ESTABLISHED
COMMAND  PID  USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
chrome  2690 mohit   87u  IPv4 140545      0t0  TCP 192.168.1.139:56620->sc-in-f188.1e100.net:5228 (ESTABLISHED)
chrome  2690 mohit  248u  IPv4 115792      0t0  TCP 192.168.1.139:50040->192.168.1.238:32400 (ESTABLISHED)
```

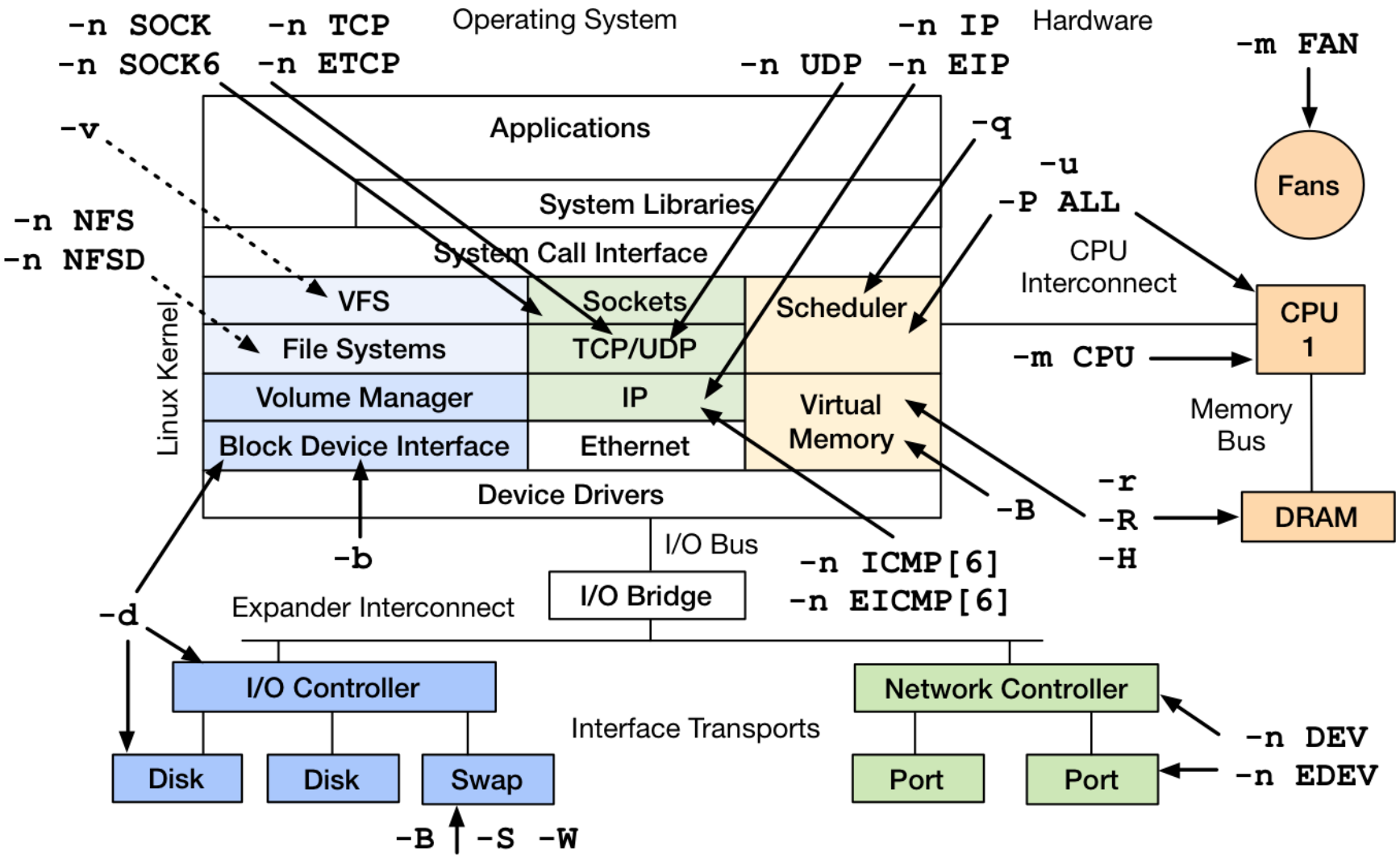
sar

- System Activity Reporter. Many stats, eg:



- Archive or live mode: (interval [count])
- Well designed. Header naming convention, logical groups: TCP, ETCP, DEV, EDEV, …

Observability: sar



Other Tools

- You may also use collectl, atop, dstat, or another measure-all tool
- The tool isn't important – it's important to have *a* way to measure everything
- In cloud environments, you are probably using a monitoring product, developed in-house or commercial.
  - We develop Atlas for cloud-wide monitoring, and Vector for instance-level analysis
  - Same method applies…

ex3:App is taking forever...

```
mohit@nomind:~/Work/UltraLowLatency$ $WORK_HOME/TOOLS/GeneralTools/lab003
```

```
vmstat 1
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 1  0      0 3550156 862764 6684484    0    0    31   412    7   55  1  1 95  3  0
 1  0      0 3551576 862764 6684484    0    0     0     0  768 1267  2 11 87  0  0
```

Some user and sys time

no swapping though

```
pidstat -t 1
Linux 4.4.0-72-generic (nomind)         Tuesday 18 April 2017   _x86_64_        (8 CPU)

11:34:46  IST   UID      TGID       TID    %usr %system  %guest    %CPU   CPU  Command
11:34:47  IST     0      1188         -    1.96    0.00    0.00    1.96     0  Xorg
11:34:47  IST     0         -      1188    1.96    0.00    0.00    1.96     0  |__Xorg
11:34:47  IST  1000     29708         -   12.75   87.25    0.00  100.00     6  lab003
11:34:47  IST  1000         -     29708   12.75   87.25    0.00  100.00     6  |__lab003
11:34:47  IST  1000     29824         -    0.98    1.96    0.00    2.94     7  pidstat
11:34:47  IST  1000         -     29824    0.98    1.96    0.00    2.94     7  |__pidstat
```

a lot more user and sys time

```
iostat -x 1
Linux 4.4.0-72-generic (nomind)         Tuesday 18 April 2017   _x86_64_        (8 CPU)

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
```

but no disc IO

```
 sar -n DEV 1
Linux 4.4.0-72-generic (nomind)         Tuesday 18 April 2017   _x86_64_        (8 CPU)

11:35:53  IST     IFACE   rxpck/s   txpck/s    rxkB/s    txkB/s   rxcmp/s   txcmp/s  rxmcst/s   %ifutil
11:35:54  IST    wlp3s0      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
11:35:54  IST        lo      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00

11:35:54  IST     IFACE   rxpck/s   txpck/s    rxkB/s    txkB/s   rxcmp/s   txcmp/s  rxmcst/s   %ifutil
11:35:55  IST    wlp3s0      4.00      2.00      0.30      0.85      0.00      0.00      0.00      0.00
11:35:55  IST        lo      4.00      4.00      0.20      0.20      0.00      0.00      0.00      0.00
```

and no network IO as well

ex3:App is taking forever...

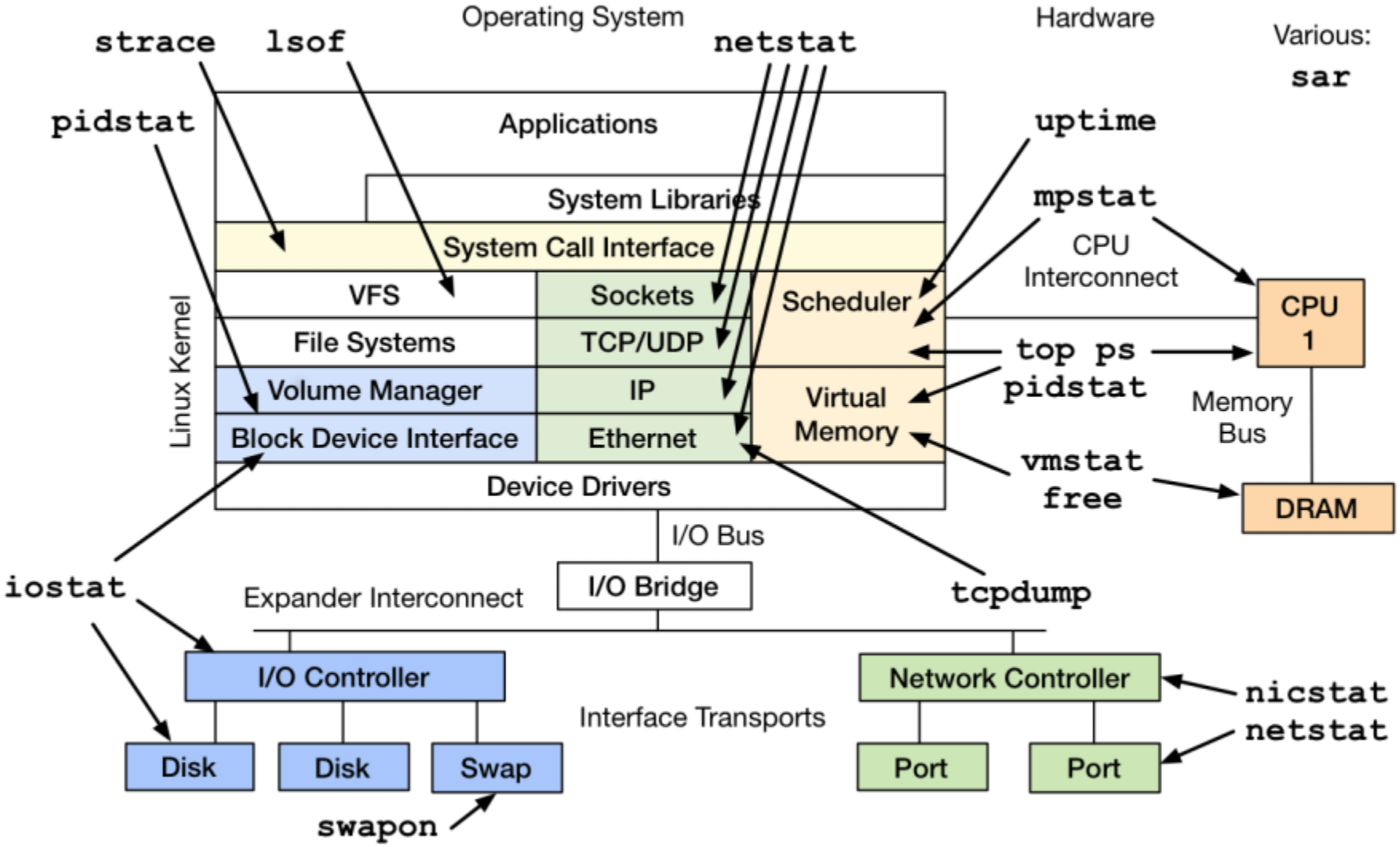
```
mohit@nomind:~/Work/UltraLowLatency$ $WORK_HOME/TOOLS/GeneralTools/lab003
```

```
mohit@nomind:~/Work/UltraLowLatency$ sudo perf trace -p `pgrep lab003` 2>&1 | head -10
[sudo] password for mohit:
     0.004 ( 0.004 ms): read(fd: 3</home/mohit/Work/UltraLowLatency/lab003.data>, buf: 0x7fff452c1097) = 0
     0.006 ( 0.001 ms): read(fd: 3</home/mohit/Work/UltraLowLatency/lab003.data>, buf: 0x7fff452c1097) = 0
     0.008 ( 0.001 ms): read(fd: 3</home/mohit/Work/UltraLowLatency/lab003.data>, buf: 0x7fff452c1097) = 0
     0.011 ( 0.001 ms): read(fd: 3</home/mohit/Work/UltraLowLatency/lab003.data>, buf: 0x7fff452c1097) = 0
     0.013 ( 0.001 ms): read(fd: 3</home/mohit/Work/UltraLowLatency/lab003.data>, buf: 0x7fff452c1097) = 0
     0.015 ( 0.001 ms): read(fd: 3</home/mohit/Work/UltraLowLatency/lab003.data>, buf: 0x7fff452c1097) = 0
     0.017 ( 0.001 ms): read(fd: 3</home/mohit/Work/UltraLowLatency/lab003.data>, buf: 0x7fff452c1097) = 0
     0.019 ( 0.001 ms): read(fd: 3</home/mohit/Work/UltraLowLatency/lab003.data>, buf: 0x7fff452c1097) = 0
     0.022 ( 0.001 ms): read(fd: 3</home/mohit/Work/UltraLowLatency/lab003.data>, buf: 0x7fff452c1097) = 0
     0.024 ( 0.001 ms): read(fd: 3</home/mohit/Work/UltraLowLatency/lab003.data>, buf: 0x7fff452c1097) = 0
mohit@nomind:~/Work/UltraLowLatency$
mohit@nomind:~/Work/UltraLowLatency$
mohit@nomind:~/Work/UltraLowLatency$
```

stuck in a loop sequestring 0 bytes

notice the use of perf version

Observability Tools: Intermediate



Advanced Observability Tools

- Misc:
  - ltrace, ss, iptraf, ethtool, snmpget, lldptool, iotop, blktrace, slabtop, /proc, pcstat
- CPU Performance Counters:
  - perf_events, tiptop, rdmsr
- Advanced Tracers:
  - perf_events, ftrace, eBPF, SystemTap, ktap, LTTng, dtrace4linux, sysdig
- Some selected demos…

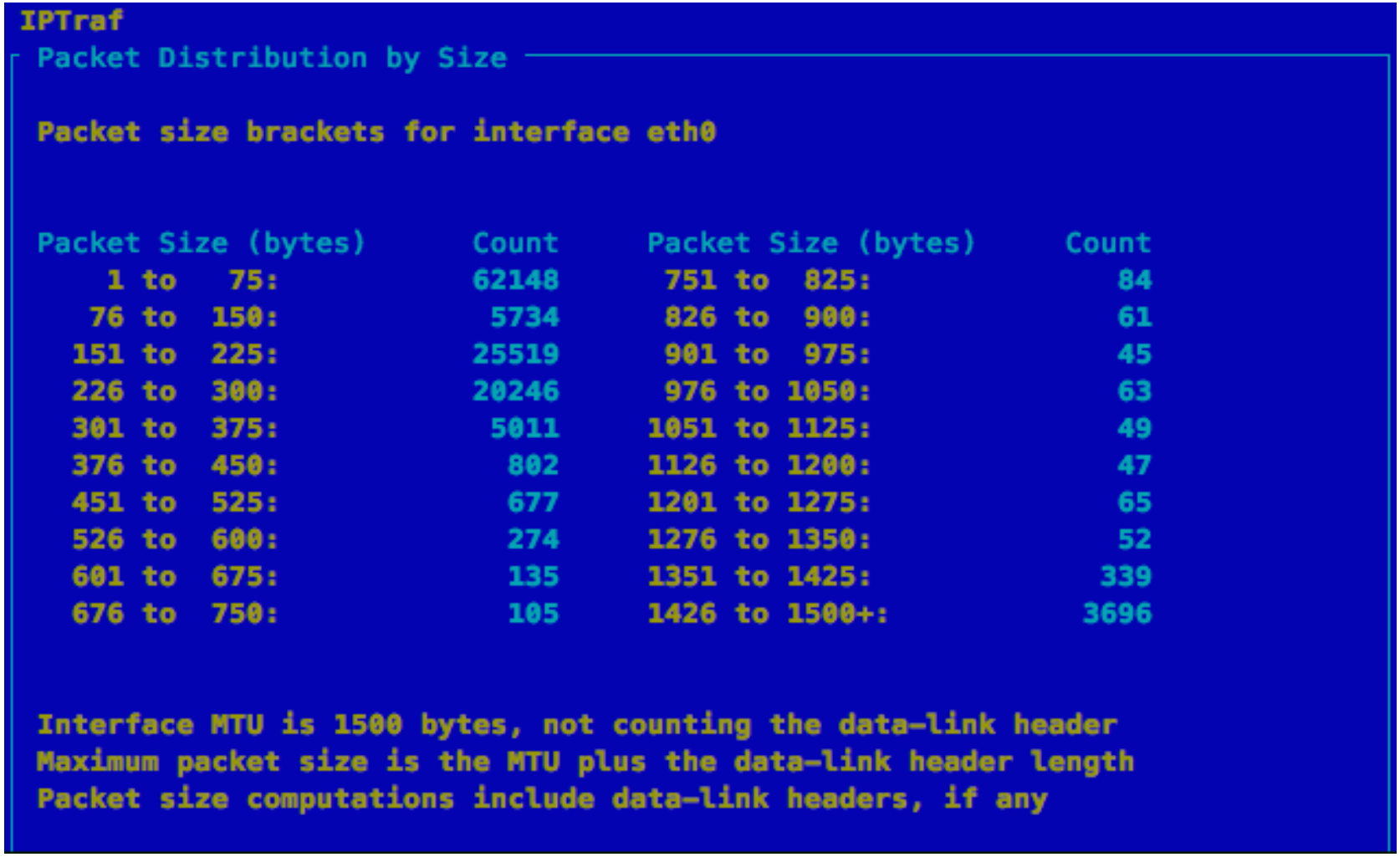
SS

- More socket statistics:

```
1 mohit@nomind:~/Work/UltraLowLatency$ ss -mop
2 Netid State      Recv-Q Send-Q                    Local Address:Port                                  Peer
  Address:Port
3 u_seq ESTAB      0      0                               @0001c 32035                                      *
  32036                 users:(("chrome",pid=2690,fd=11))
4         skmem:(r0,rb212992,t0,tb212992,f0,w0,o0,bl0)
5 u_str ESTAB      0      0                                * 232098                                         *
  227900              users:(("gimp-2.8",pid=31043,fd=17))
6         skmem:(r0,rb212992,t0,tb212992,f0,w0,o0,bl0)
7 u_str ESTAB      0      0                                * 225164                                         *
  227114              users:(("soffice.bin",pid=30321,fd=14))
8         skmem:(r0,rb212992,t0,tb212992,f0,w0,o0,bl0)
9 u_str ESTAB      0      0                                * 93107                                          *
  95887               users:(("compiz",pid=14600,fd=21))
10        skmem:(r0,rb212992,t0,tb212992,f0,w0,o0,bl0)
11
12
13 mohit@nomind:~/Work/UltraLowLatency$ ss -i
14 Netid State      Recv-Q Send-Q                    Local Address:Port                                  Peer
  Address:Port
15 u_seq ESTAB      0      0                               @0001c 32035                                      *
  32036
16 u_str ESTAB      0      0                                * 232098                                         * 227900
```

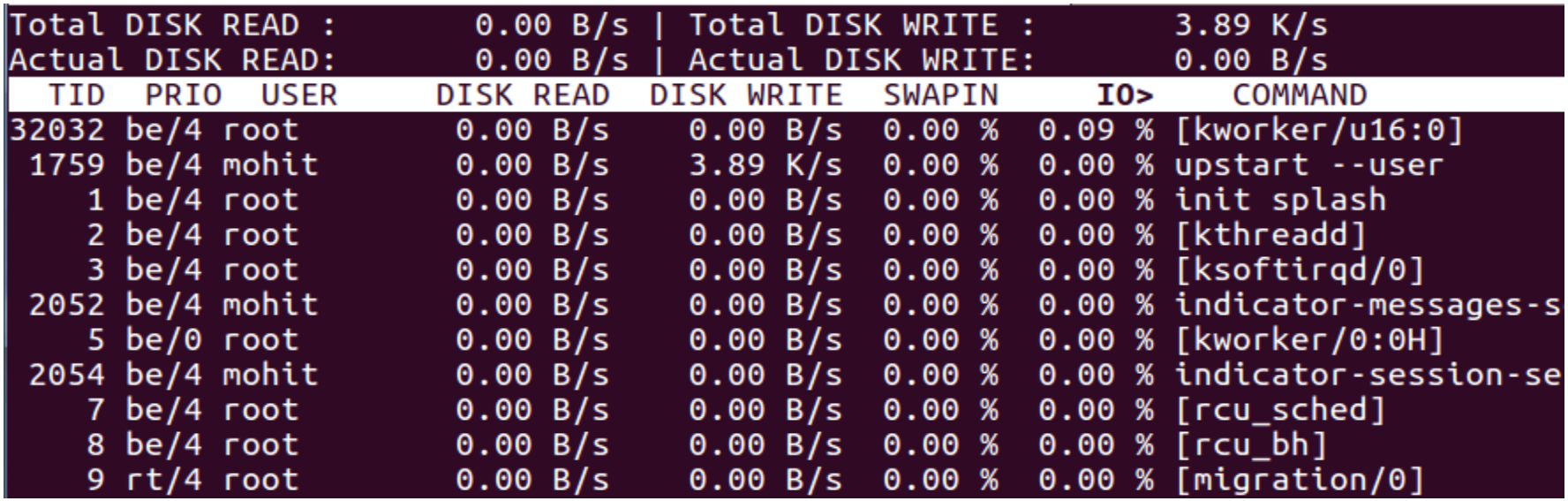
m = socket kernel memory
p = process using it
i = tcp internal state

iptraf


IPTraf
Packet Distribution by Size
Packet size brackets for interface eth0
Packet Size (bytes) Count Packet Size (bytes) Count
1 to 75: 62148 751 to 825: 84
76 to 150: 5734 826 to 900: 61
151 to 225: 25519 901 to 975: 45
226 to 300: 20246 976 to 1050: 63
301 to 375: 5011 1051 to 1125: 49
376 to 450: 802 1126 to 1200: 47
451 to 525: 677 1201 to 1275: 65
526 to 600: 274 1276 to 1350: 52
601 to 675: 135 1351 to 1425: 339
676 to 750: 105 1426 to 1500+: 3696
Interface MTU is 1500 bytes, not counting the data-link header
Maximum packet size is the MTU plus the data-link header length
Packet size computations include data-link headers, if any

iotop

- Block device I/O (disk) by process:



```
Total DISK READ :       0.00 B/s | Total DISK WRITE :       3.89 K/s
Actual DISK READ:       0.00 B/s | Actual DISK WRITE:       0.00 B/s
  TID  PRIO  USER     DISK READ  DISK WRITE  SWAPIN     IO>    COMMAND
32032 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.09 % [kworker/u16:0]
 1759 be/4 mohit       0.00 B/s    3.89 K/s  0.00 %  0.00 % upstart --user
    1 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % init splash
    2 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [kthreadd]
    3 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [ksoftirqd/0]
 2052 be/4 mohit       0.00 B/s    0.00 B/s  0.00 %  0.00 % indicator-messages-s
    5 be/0 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [kworker/0:0H]
 2054 be/4 mohit       0.00 B/s    0.00 B/s  0.00 %  0.00 % indicator-session-se
    7 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [rcu_sched]
    8 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [rcu_bh]
    9 rt/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [migration/0]
```

- Needs kernel support enabled
  – CONFIG_TASK_IO_ACCOUNTING

slabtop

- Kernel slab allocator memory usage:

```
 Active / Total Objects (% used)    : 2341588 / 2360312 (99.2%)
 Active / Total Slabs (% used)      : 73054 / 73054 (100.0%)
 Active / Total Caches (% used)     : 78 / 122 (63.9%)
 Active / Total Size (% used)       : 613464.27K / 617828.50K (99.3%)
 Minimum / Average / Maximum Object : 0.01K / 0.26K / 18.50K

  OBJS ACTIVE  USE OBJ SIZE  SLABS OBJ/SLAB CACHE SIZE NAME
968916 968916 100%    0.10K  24844       39     99376K buffer_head
516285 516102  99%    0.19K  24585       21     98340K dentry
270930 270506  99%    1.05K   9031       30    288992K ext4_inode_cache
127194 126857  99%    0.04K   1247      102      4988K ext4_extent_status
 71936  69658  96%    0.06K   1124       64      4496K kmalloc-64
 53456  53347  99%    0.61K   2056       26     32896K proc_inode_cache
 49408  44989  91%    0.03K    386      128      1544K kmalloc-32
 48496  46856  96%    0.57K   1732       28     27712K radix_tree_node
 44940  43049  95%    0.20K   2247       20      8988K vm_area_struct
 34442  34099  99%    0.12K   1013       34      4052K kernfs_node_cache
 18615  16901  90%    0.08K    365       51      1460K anon_vma
 18480  18480 100%    0.55K    660       28     10560K inode_cache
 16575  16028  96%    0.05K    195       85       780K ftrace_event_field
 14848  13339  89%    0.25K    464       32      3712K kmalloc-256
 14553  13446  92%    0.19K    693       21      2772K kmalloc-192
 11004  10689  97%    0.09K    262       42      1048K kmalloc-96
  8032   7445  92%    0.50K    251       32      4016K kmalloc-512
  7680   7680 100%    0.01K     15      512        60K kmalloc-8
```

pcstat

- Show page cache residency by file:

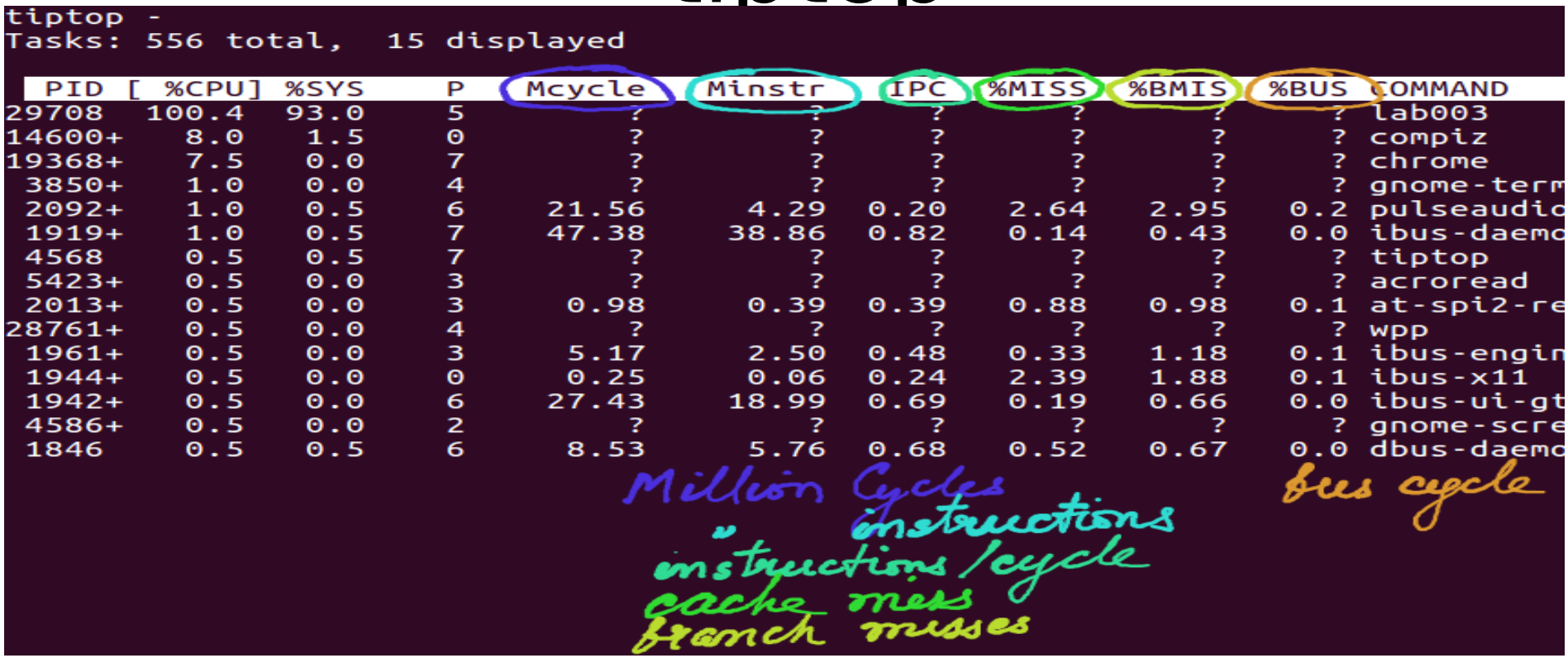
```
# ./pcstat data0*
|----------+----------------+------------+-----------+---------|
| Name     | Size           | Pages      | Cached    | Percent |
|----------+----------------+------------+-----------+---------|
| data00   | 104857600      | 25600      | 25600     | 100.000 |
| data01   | 104857600      | 25600      | 25600     | 100.000 |
| data02   | 104857600      | 25600      | 4080      | 015.938 |
| data03   | 104857600      | 25600      | 25600     | 100.000 |
| data04   | 104857600      | 25600      | 16010     | 062.539 |
| data05   | 104857600      | 25600      | 0         | 000.000 |
|----------+----------------+------------+-----------+---------|
```

- Uses the mincore(2) syscall. Useful for database performance analysis.

perf_events

- Provides the "perf" command
- In Linux source code: tools/perf
  - Usually pkg added by linux-tools-common, etc.
- **Multi-tool** with many capabilities
  - CPU profiling
  - PMC profiling
  - Static & dynamic tracing
- *(Covered seperately)*

tiptop



- IPC by process, %MISS, %BUS
- Needs some love. perfmon2 library integration?
- Still can’t use it in clouds yet (needs PMCs enabled)

rdmsr

- Model Specific Registers (MSRs), unlike PMCs, can be read by default in Xen guests
  - Timestamp clock, temp, power, …
  - Use rdmsr(1) from the msr-tools package to read them

```
root@nomind:/home/mohit/Work/Linux/resources/msr-cloud-tools# ./showboost
CPU MHz     : 3582
Turbo MHz   : 0 (10 active)
Turbo Ratio : 0% (10 active)
CPU 0 summary every 5 seconds...

TIME        C0_MCYC      C0_ACYC        UTIL  RATIO    MHz
19:11:04    107774990    85574420         0%    79%   2844
19:11:09    100347459    60763011         0%    60%   2168
19:11:14    149731187    122672362        0%    81%   2934
19:11:24    89840534     64148047         0%    71%   2557
19:11:29    70662409     37820871         0%    53%   1917

root@nomind:/home/mohit/Work/Linux/resources/msr-cloud-tools# rdmsr -p2 0x10
117a57197c245

root@nomind:/home/mohit/Work/Linux/resources/msr-cloud-tools# ./cputemp
CPU1 CPU2 CPU3 CPU4 CPU5 CPU6 CPU7 CPU8
62 62 60 61 62 63 61 60
```

Real CPU frequencies queried by their MSR mask (Check the code).

Timestamp or (RDTSC)

CPU temperature

More Advanced Tools...

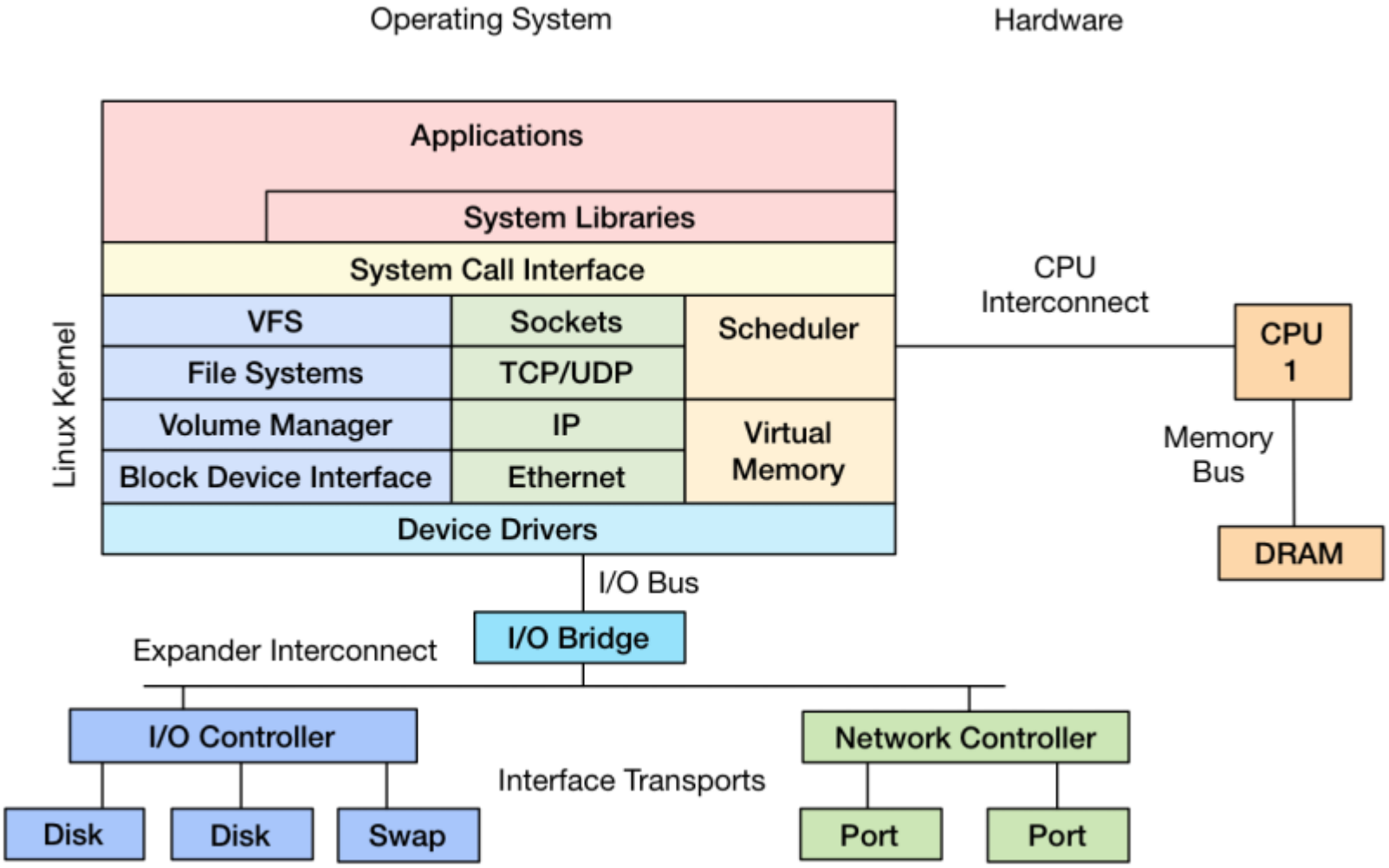
- Some others worth mentioning:

Tool	Description
ltrace	Library call tracer
ethtool	Mostly interface tuning; some stats
snmpget	SNMP network host statistics
lldptool	Can get LLDP broadcast stats
blktrace	Block I/O event tracer
/proc	Many raw kernel counters
pmu–tools	On– and off–core CPU counter tools

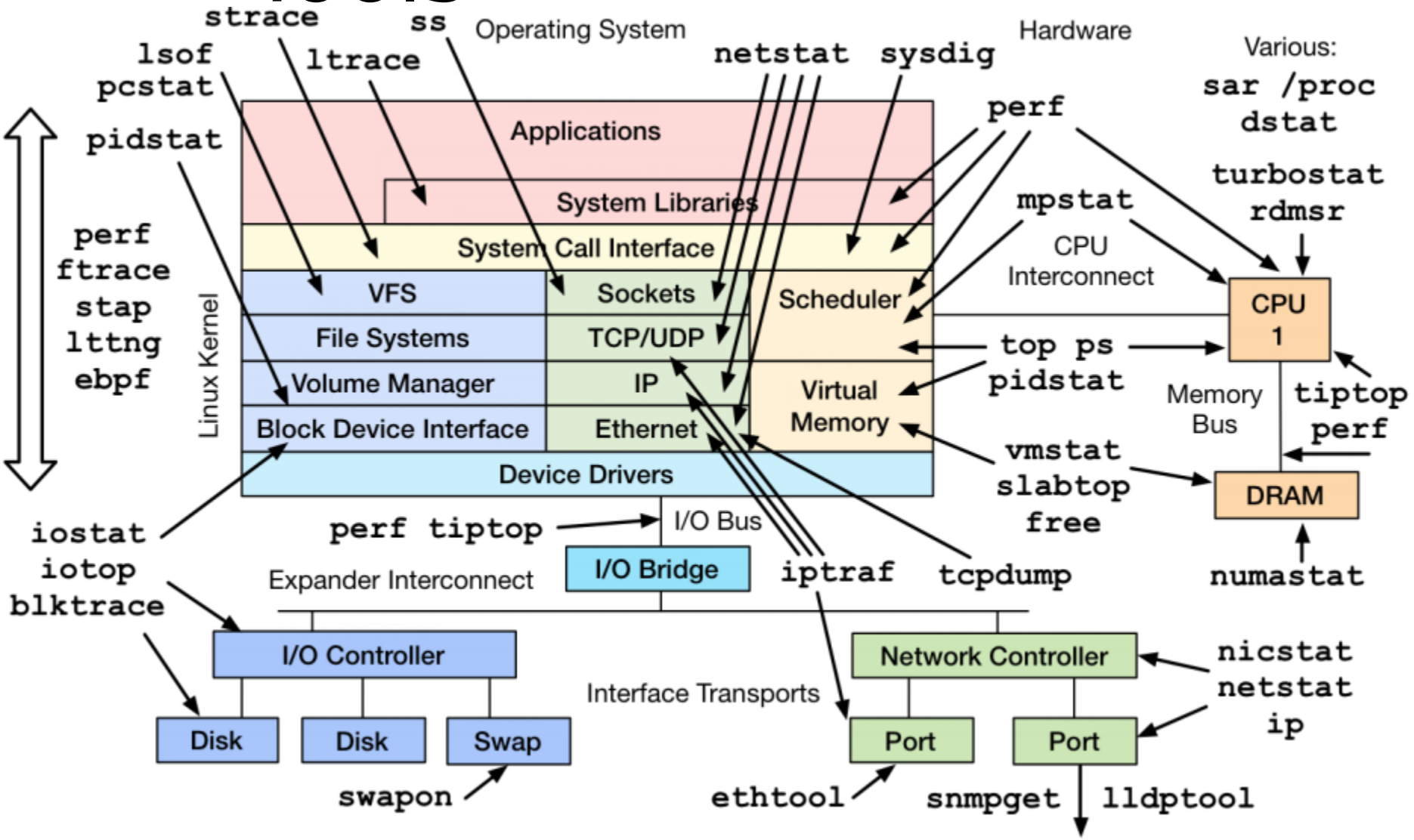
Advanced Tracers

- Many options on Linux:
  - perf_events, ftrace, eBPF, SystemTap, ktap, LTTng, dtrace4linux, sysdig
- Most can do static and dynamic tracing
  - Static: pre-defined events (tracepoints)
  - Dynamic: instrument any software (kprobes, uprobes).
    Custom metrics on-demand. *Catch all*.
- Many are in-development

Linux Observability Tools



Linux Observability Tools



Benchmarking Tools

Benchmarking Tools

- Multi:
  - UnixBench, lmbench, sysbench, perf bench
- FS/disk:
  - dd, hdparm, fio
- App/lib:
  - ab, wrk, jmeter, openssl
- Networking:
  - ping, hping3, iperf, ttcp, traceroute, mtr, pchar

Benchmarking

- Most(~100%) of benchmarks are wrong
- Results are usually misleading:
  you benchmark A, but actually measure B, and conclude you measured C
- Common mistakes:
  - Testing the wrong target: eg, FS cache instead of disk
  - Choosing the wrong target: eg, disk instead of FS cache
    … doesn't resemble real world usage
  - Invalid results: eg, bugs
- The energy needed to refute benchmarks is multiple orders of magnitude bigger than to run them

Active Benchmarking (Method)

1. Run the benchmark for hours
2. While running, analyze and confirm the performance limiter using *observability tools*
   - Disk benchmark: run iostat, …
   - CPU benchmark: run pidstat, perf, flame graphs, …
   - …

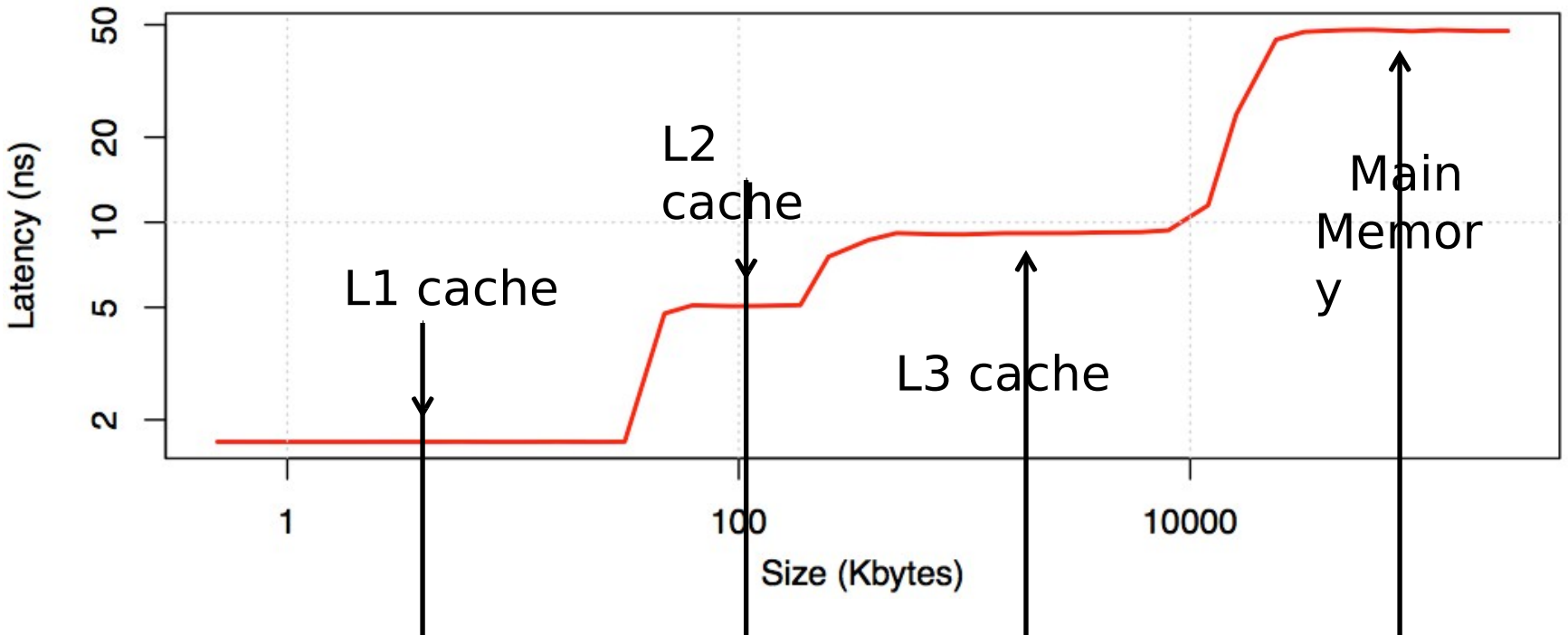
- Answer the question: why isn't the result 10x?

We just covered the observability tools – use them!

lmbench

- CPU, memory, and kernel micro-benchmarks
- Eg, memory latency by stride size:

```
$ lat_mem_rd 100m 128 > out.latencies
some R processing…
```



fio

- FS or disk I/O micro-benchmarks

```
mohit@nomind:~/Work/UltraLowLatency$ fio --name=global --rw=randread --name=job1 --runtime=60 --time_based
job1: (g=0): rw=randread, bs=4K-4K/4K-4K/4K-4K, ioengine=sync, iodepth=1
fio-2.2.10
Starting 1 process
Jobs: 1 (f=1): [r(1)] [100.0% done] [3508MB/0KB/0KB /s] [898K/0/0 iops] [eta 00m:00s]
job1: (groupid=0, jobs=1): err= 0: pid=12011: Tue Apr 18 19:42:07 2017
  read : io=122446MB, bw=2040.8MB/s, iops=522429, runt= 60001msec
    clat (usec): min=0, max=7872, avg= 1.54, stdev= 9.45
     lat (usec): min=0, max=7872, avg= 1.57, stdev= 9.45
    clat percentiles (usec):
     |  1.00th=[    0],  5.00th=[    0], 10.00th=[    0], 20.00th=[    0],
     | 30.00th=[    0], 40.00th=[    1], 50.00th=[    1], 60.00th=[    1],
     | 70.00th=[    1], 80.00th=[    1], 90.00th=[    1], 95.00th=[    1],
     | 99.00th=[    1], 99.50th=[   92], 99.90th=[  111], 99.95th=[  113],
     | 99.99th=[  119]
    bw (MB  /s): min=   33, max= 3762, per=99.41%, avg=2028.71, stdev=1801.83
    lat (usec) : 2=99.02%, 4=0.06%, 10=0.02%, 20=0.06%, 50=0.01%
    lat (usec) : 100=0.42%, 250=0.42%, 500=0.01%, 750=0.01%, 1000=0.01%
    lat (msec) : 2=0.01%, 4=0.01%, 10=0.01%
```

- Results include basic latency distribution

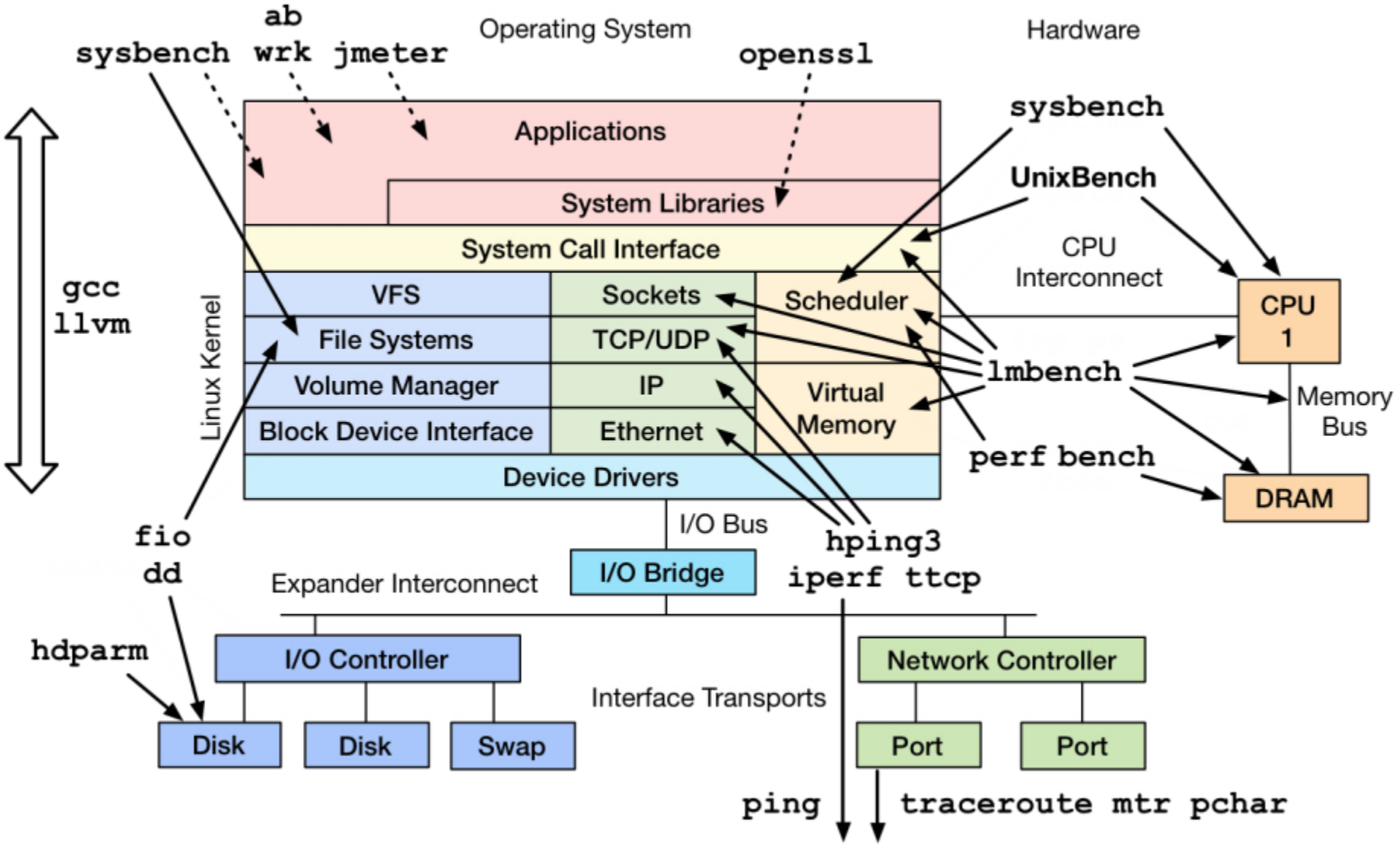
pchar

- Traceroute with bandwidth per hop!

```
$ pchar 10.71.83.1
[…]
 4: 10.110.80.1 (10.110.80.1)
    Partial loss:      0 / 5 (0%)
    Partial char:      rtt = 9.351109 ms, (b = 0.004961 ms/B), r2 = 0.184105
                       stddev rtt = 4.967992, stddev b = 0.006029
    Partial queueing:  avg = 0.000000 ms (0 bytes)
    Hop char:          rtt = --.--- ms, bw = 1268.975773 Kbps
    Hop queueing:      avg = 0.000000 ms (0 bytes)
 5: 10.193.43.181 (10.193.43.181)
    Partial loss:      0 / 5 (0%)
    Partial char:      rtt = 25.461597 ms, (b = 0.011934 ms/B), r2 = 0.228707
                       stddev rtt = 10.426112, stddev b = 0.012653
    Partial queueing:  avg = 0.000000 ms (0 bytes)
    Hop char:          rtt = 16.110487 ms, bw = 1147.210397 Kbps
    Hop queueing:      avg = 0.000000 ms (0 bytes)
[…]
```

- Needs love. Based on pathchar (Linux 2.0.30).

Benchmarking Tools



Tuning Tools

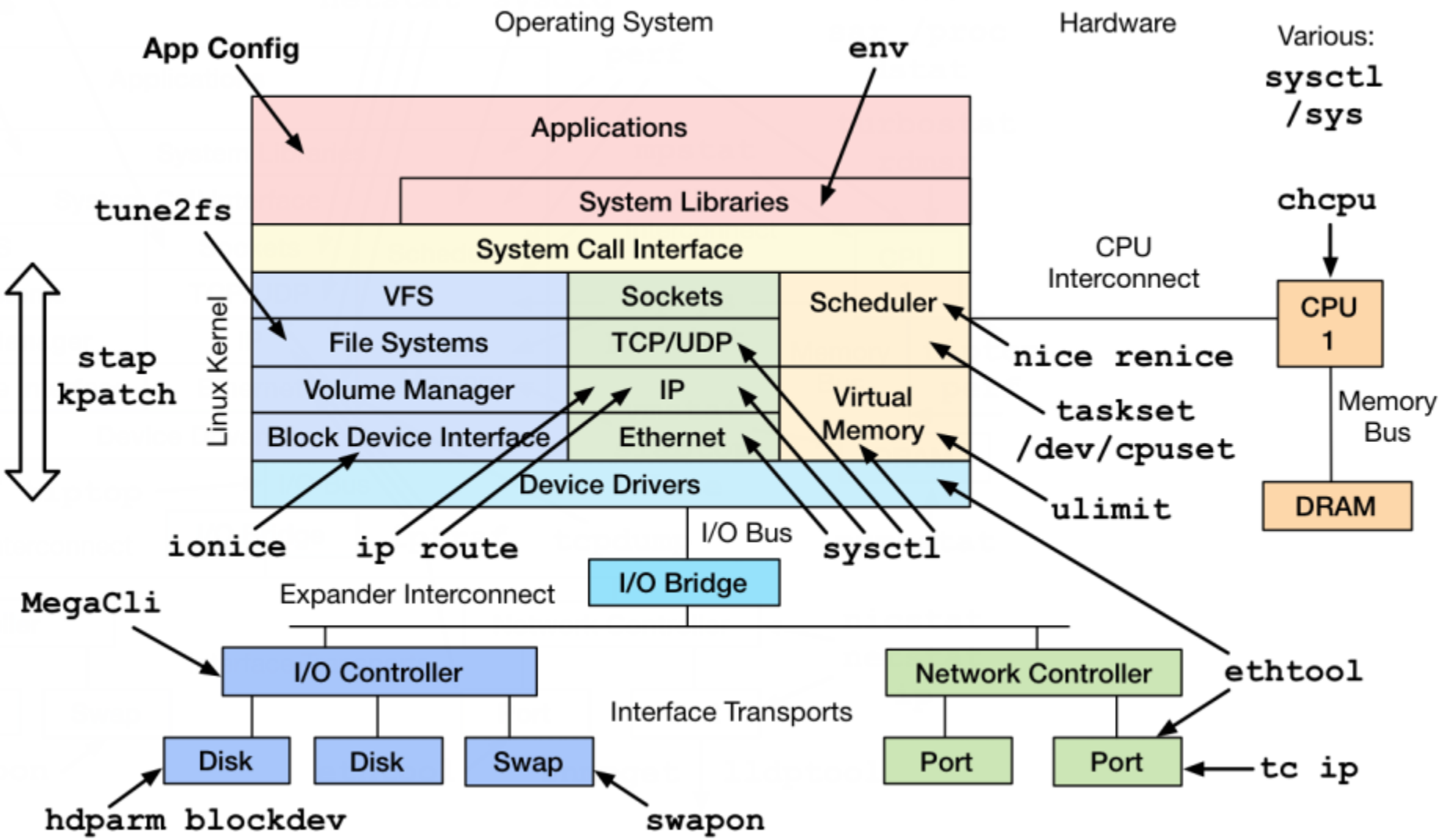
Tuning Tools

- Generic interfaces:
  - sysctl, /sys
- Many areas have custom tuning tools:
  - Applications: their own config
  - CPU/scheduler: nice, renice, taskset, ulimit, chcpu
  - Storage I/O: tune2fs, ionice, hdparm, blockdev, …
  - Network: ethtool, tc, ip, route
  - Dynamic patching: stap, kpatch

Tuning Methods

- Scientific Method:
  1. Question
  2. Hypothesis
  3. Prediction
  4. Test
  5. Analysis
- Any observational or benchmarking tests you can try before tuning?
- Consider risks, and see previous tools

Tuning Tools



Static Tools

Static Tools

- Static Performance Tuning: check the static state and configuration of the system
  - CPU types & flags
  - CPU frequency scaling config
  - Storage devices
  - File system capacity
  - File system and volume configuration
  - Route table
  - State of hardware
  - etc.
- What can be checked on a system without load
- Methodology by Richard Elling (2000)

CPU Types & Flags

```
mohit@nomind:~/Work/UltraLowLatency$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 70
model name      : Intel(R) Core(TM) i7-4870HQ CPU @ 2.50GHz
stepping        : 1
microcode       : 0xf
cpu MHz         : 3618.457
cache size      : 6144 KB
physical id     : 0
siblings        : 8
core id         : 0
cpu cores       : 4
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse
p_good nopl xtopology nonstop_tsc aperfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fm
avx f16c rdrand lahf_lm abm epb tpr_shadow vnmi flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms
```

CPU Frequency Scaling

- Kernel may be configured to dynamically modify CPU frequency:

```
mohit@nomind:~/Work/UltraLowLatency$ cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_min_freq
800000
mohit@nomind:~/Work/UltraLowLatency$ cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_max_freq
3700000
mohit@nomind:~/Work/UltraLowLatency$ cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
powersave
```

  - See Documentation/cpu-freq/governors.txt, and scaling_governor == performance
- Not to be confused with Intel Turbo Boost (which is H/W)

Storage Devices

```
mohit@nomind:~/Work/UltraLowLatency$ cat /proc/scsi/scsi
Attached devices:
Host: scsi0 Channel: 00 Id: 00 Lun: 00
  Vendor: ATA      Model: APPLE SSD SM0512 Rev: JA1Q
  Type:   Direct-Access                    ANSI  SCSI revision: 05
Host: scsi1 Channel: 00 Id: 00 Lun: 00
  Vendor: APPLE    Model: SD Card Reader   Rev: 3.00
  Type:   Direct-Access                    ANSI  SCSI revision: 06
```

- Micro-benchmarking disks (not file systems!) is also useful for understanding their characteristics

Routing Table

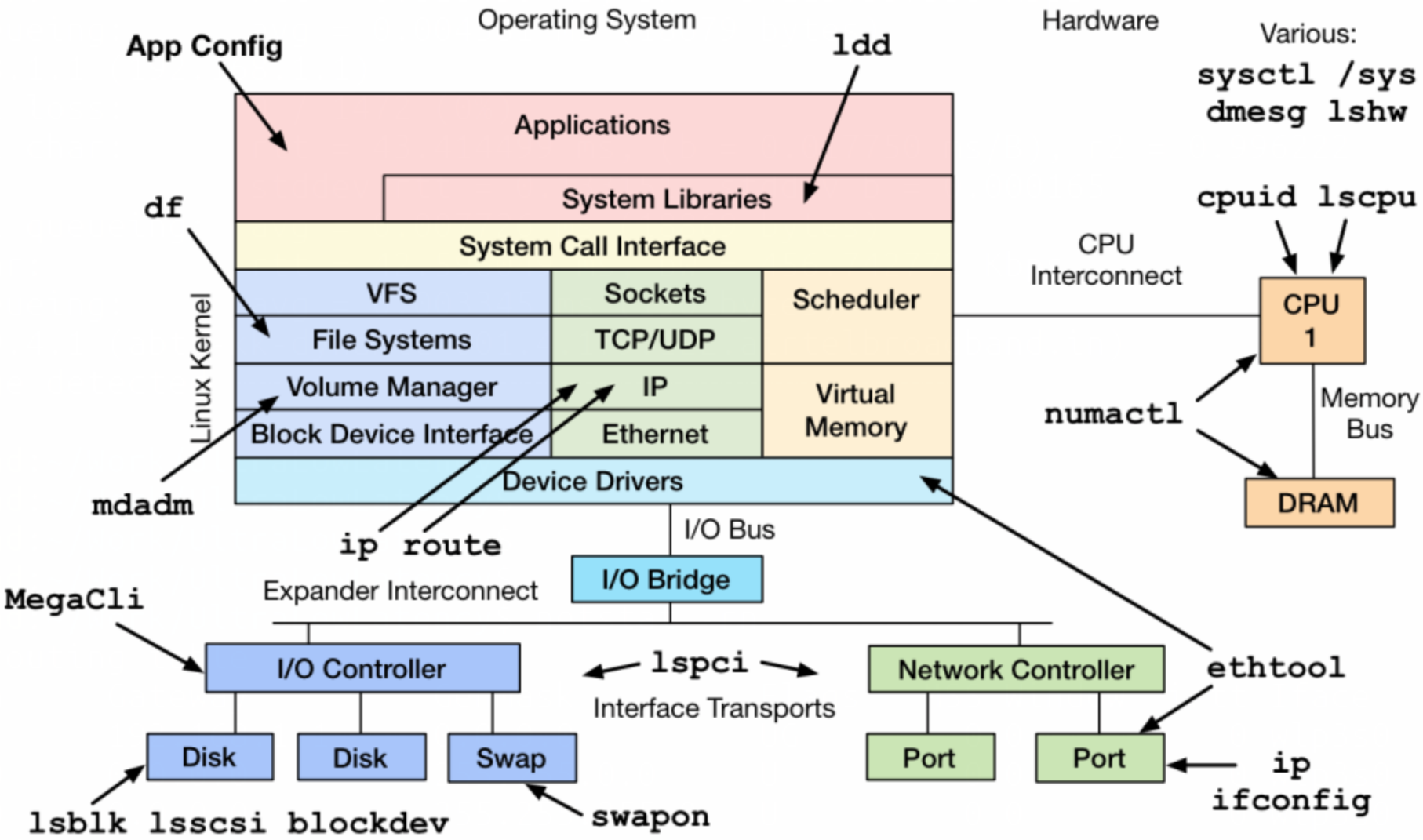
- Use "ip route get" to test a given IP:

```
mohit@nomind:~/Work/UltraLowLatency$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG        0 0          0 wlp3s0
169.254.0.0     0.0.0.0         255.255.0.0     U         0 0          0 wlp3s0
192.168.1.0     0.0.0.0         255.255.255.0   U         0 0          0 wlp3s0
mohit@nomind:~/Work/UltraLowLatency$
mohit@nomind:~/Work/UltraLowLatency$
mohit@nomind:~/Work/UltraLowLatency$ ip route get 54.214.28.210
54.214.28.210 via 192.168.1.1 dev wlp3s0  src 192.168.1.139
    cache
```

etc...

- System messages: `dmesg`
- Network interface config: `ifconfig –a; ip link`
- File system capacity: `df -h`
- Volume config: `mdadm --misc -D /dev/md0`
- Storage device info: `smartctl`
- NUMA config: `numactl -s; numactl -H`
- PCI info: `lspci`
- Installed kernel modules: `lsmod`
- Root crontab config: `crontab –l`
- Services: `service --status-all`
- …

Static Tools

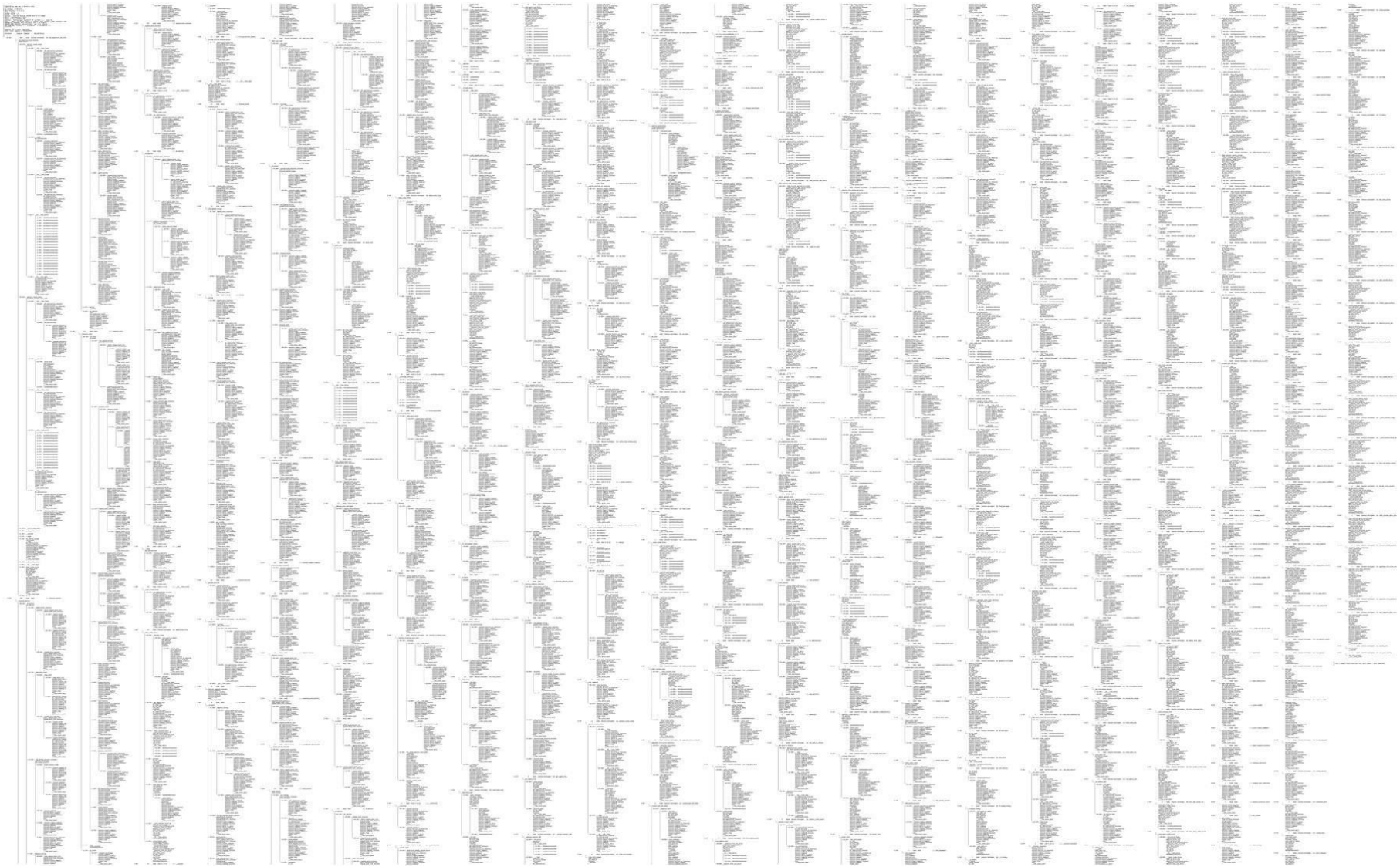


Profiling

Profiling

- Objectives:
  - Profile CPU usage by stack sampling
  - Generate CPU flame graphs
  - Understand gotchas with stacks & symbols

perf_events: Ref to HPC



ex:4:Mysterious CPU consumer...

```
mohit@nomind:~/Work/UltraLowLatency$ $WORK_HOME/TOOLS/GeneralTools/lab004
```

```
top - 05:00:42 up  7:58,  1 user,  load average: 0.85, 0.50, 0.44
Tasks: 276 total,   2 running, 273 sleeping,   0 stopped,   1 zombie
%Cpu(s): 13.3 us,  1.1 sy,  0.0 ni, 85.6 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 16338544 total,  9540352 free,  4283832 used,  2514360 buff/cache
KiB Swap: 17576956 total, 17576956 free,        0 used. 11320392 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 1212 root      20   0  689364 286308  79252 S   5.3  1.8   4:54.73 Xorg
13185 mohit     20   0  628668  31708  26200 S   3.3  0.2   0:00.10 gnome-screensho
 2223 mohit     20   0 2216120 739040  90756 S   2.3  4.5   2:11.45 compiz
 5527 mohit     20   0 1256932 443008 274104 S   1.0  2.7   3:06.77 chrome
 1232 root     -51   0       0      0      0 S   0.7  0.0   1:20.60 irq/40-nvidia
 1986 mohit     20   0  524160  30752  21880 S   0.7  0.2   0:03.09 bamfdaemon
 8024 mohit     20   0 2011788 680552 129156 S   0.7  4.2   0:24.79 wpp
 1018 influxdb  20   0  508172  60020  13828 S   0.3  0.4   0:39.96 influxd
 1035 mysql     20   0 1236388 160696  16732 S   0.3  1.0   0:04.96 mysqld
 1128 root      20   0    4552   1568   1352 S   0.3  0.0   0:26.93 macfanctld
 1141 redis     20   0   40136   6556   2232 S   0.3  0.0   0:05.78 redis-server
 1898 mohit     20   0   44220   4788   2912 S   0.3  0.0   0:03.36 dbus-daemon
 1987 mohit     20   0  433880  12640   7136 S   0.3  0.1   0:06.93 ibus-daemon
 1999 mohit     20   0  274520   6348   5572 S   0.3  0.0   0:00.08 gvfsd
 2116 mohit     20   0  663168  63444  26056 S   0.3  0.4   0:10.28 unity-panel-ser
 2436 mohit     20   0  647328  20144  13796 S   0.3  0.1   0:00.23 unity-scope-hom
 2847 mohit     20   0  731392 295520 101196 S   0.3  1.8   2:53.56 chrome
 3673 mohit     20   0 7006428 913616  40484 S   0.3  5.6   1:38.72 java
 4143 mohit     20   0  668972  42392  28836 S   0.3  0.3   0:03.19 gnome-terminal-
 5003 mohit     20   0  234796 121404  52304 S   0.3  0.7   1:17.81 acroread
 8415 mohit     20   0    4228   1284   1192 S   0.3  0.0   0:00.19 lab004
11281 mohit     20   0   41940   3880   3216 R   0.3  0.0   0:00.04 top
13374 mohit     20   0    7296    712    636 R   0.3  0.0   0:00.01 cksum
    1 root      20   0  185592   6208   3948 S   0.0  0.0   0:02.89 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S   0.0  0.0   0:00.02 ksoftirqd/0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0H
    7 root      20   0       0      0      0 S   0.0  0.0   0:09.15 rcu_sched
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 rcu_bh
```

```
mohit@nomind:~/Work/UltraLowLatency$ mpstat -P ALL 2
Linux 4.4.0-75-generic (nomind)         Wednesday 26 April 2017         _x86_64_        (8 CPU)

05:09:12  IST  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
05:09:14  IST  all   12.20    0.00    0.75    0.06    0.00    0.00    0.00    0.00    0.00   86.98
05:09:14  IST    0   14.43    0.00    1.00    0.00    0.00    0.00    0.00    0.00    0.00   84.58
05:09:14  IST    1    5.50    0.00    1.00    0.00    0.00    0.00    0.00    0.00    0.00   93.50
05:09:14  IST    2   13.57    0.00    1.01    0.00    0.00    0.00    0.00    0.00    0.00   85.43
05:09:14  IST    3   11.62    0.00    1.01    0.51    0.00    0.00    0.00    0.00    0.00   86.87
05:09:14  IST    4    4.50    0.00    0.50    0.00    0.00    0.00    0.00    0.00    0.00   95.00
05:09:14  IST    5    6.50    0.00    0.50    0.00    0.00    0.00    0.00    0.00    0.00   93.00
05:09:14  IST    6    5.91    0.00    0.99    0.00    0.00    0.00    0.00    0.00    0.00   93.10
05:09:14  IST    7   35.50    0.00    1.00    0.00    0.00    0.00    0.00    0.00    0.00   63.50

05:09:14  IST  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
05:09:16  IST  all   12.89    0.00    1.00    0.00    0.00    0.06    0.00    0.00    0.00   86.05
05:09:16  IST    0   28.00    0.00    1.50    0.00    0.00    0.00    0.00    0.00    0.00   70.50
05:09:16  IST    1   23.86    0.00    0.51    0.00    0.00    0.00    0.00    0.00    0.00   75.63
05:09:16  IST    2    8.50    0.00    0.50    0.00    0.00    0.00    0.00    0.00    0.00   91.00
05:09:16  IST    3    4.04    0.00    1.52    0.00    0.00    0.00    0.00    0.00    0.00   94.44
05:09:16  IST    4   11.50    0.00    1.00    0.00    0.00    0.00    0.00    0.00    0.00   87.50
05:09:16  IST    5   14.07    0.00    1.51    0.00    0.00    0.00    0.00    0.00    0.00   84.42
05:09:16  IST    6    8.04    0.00    1.01    0.00    0.00    0.00    0.00    0.00    0.00   90.95
05:09:16  IST    7    4.48    0.00    0.50    0.00    0.00    0.00    0.00    0.00    0.00   95.02
```

- moderate amount of CPU usage.

ex:4:Mysterious CPU consumer...

```
mohit@nomind:~/Work/UltraLowLatency$ $WORK_HOME/TOOLS/GeneralTools/lab004
```

```
mohit@nomind:~/Work/UltraLowLatency$ iostat -x 1
Linux 4.4.0-75-generic (nomind)         Wednesday 26 April 2017         _x86_64_        (8 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           6.27    0.01    1.37    0.05    0.00   92.31

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.13     1.68    2.07    1.56    52.41    54.22    58.79     0.00    1.22    0.72    1.90   0.55   0.20

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          11.68    0.00    1.13    0.00    0.00   87.19

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     0.00    2.00    0.00    12.00     0.00    12.00     0.00    0.00    0.00    0.00   0.00   0.00
```

```
mohit@nomind:~/Work/UltraLowLatency$ sar -n DEV 1
Linux 4.4.0-75-generic (nomind)         Wednesday 26 April 2017         _x86_64_        (8 CPU)

05:12:57  IST     IFACE   rxpck/s   txpck/s    rxkB/s    txkB/s   rxcmp/s   txcmp/s  rxmcst/s   %ifutil
05:12:58  IST    wlp3s0      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
05:12:58  IST        lo      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00

05:12:58  IST     IFACE   rxpck/s   txpck/s    rxkB/s    txkB/s   rxcmp/s   txcmp/s  rxmcst/s   %ifutil
05:12:59  IST    wlp3s0      1.00      1.00      0.06      0.08      0.00      0.00      0.00      0.00
05:12:59  IST        lo      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00

05:12:59  IST     IFACE   rxpck/s   txpck/s    rxkB/s    txkB/s   rxcmp/s   txcmp/s  rxmcst/s   %ifutil
05:13:00  IST    wlp3s0      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
05:13:00  IST        lo      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
```

- but no disc IO or network IO.

ex:4:Mysterious CPU consumer...

```
+   33.58%     0.06%  swapper          [kernel.kallsyms]          [k] cpu_startup_entry
-   33.40%     0.00%  cksum            cksum                      [.] 0xfffffffffc01875
     0x1875
-   33.40%    33.40%  cksum            cksum                      [.] 0x0000000000001875
   + 0.67% 0x16bc260
   + 0.66% 0x16aa260
   + 0.60% 0x1151260
   + 0.59% 0x2157260
   + 0.58% 0x1131260
   + 0.57% 0x1754260
   + 0.55% 0x1718260
   + 0.55% 0x85b260
   + 0.55% 0x1094260
   + 0.55% 0x8b9260
   + 0.55% 0x1451260
   + 0.55% 0x1f3e260
   + 0.54% 0x1ae0260
   + 0.51% 0x17ca260
   + 0.50% 0x1ce6260
   + 0.49% 0x1a6f260
```

ex:4:Mysterious CPU consumer...

```
root@nomind:/home/mohit/Work/Linux/resources/perf-tools# ./execsnoop
Tracing exec()s. Ctrl-C to end.
Instrumenting sys_execve
   PID   PPID ARGS
  6833   6831 cat -v trace_pipe
  6832   6828 gawk -v o=1 -v opt_name=0 -v name= -v opt_duration=0 [...]
  6834   3194 /home/mohit/Work/UltraLowLatency/TOOLS/GeneralTools/lab004
  6835   6834            lab004-6835  [002] d...  4361.466547: execsnoop_sys_execve: (SyS_execve+0x0/0x50)
  6836   6835 cksum lab004.data
  6837   6834                 sh-6837  [006] d...  4361.505298: execsnoop_sys_execve: (SyS_execve+0x0/0x50)
  6838   6837 cksum lab004.data
  6839   6834            lab004-6839  [006] d...  4361.534419: execsnoop_sys_execve: (SyS_execve+0x0/0x50)
  6840   6839 cksum lab004.data
  6841   6834            lab004-6841  [007] d...  4361.563534: execsnoop_sys_execve: (SyS_execve+0x0/0x50)
  6842   6841 cksum lab004.data
  6843   6834            lab004-6843  [007] d...  4361.592215: execsnoop_sys_execve: (SyS_execve+0x0/0x50)
```

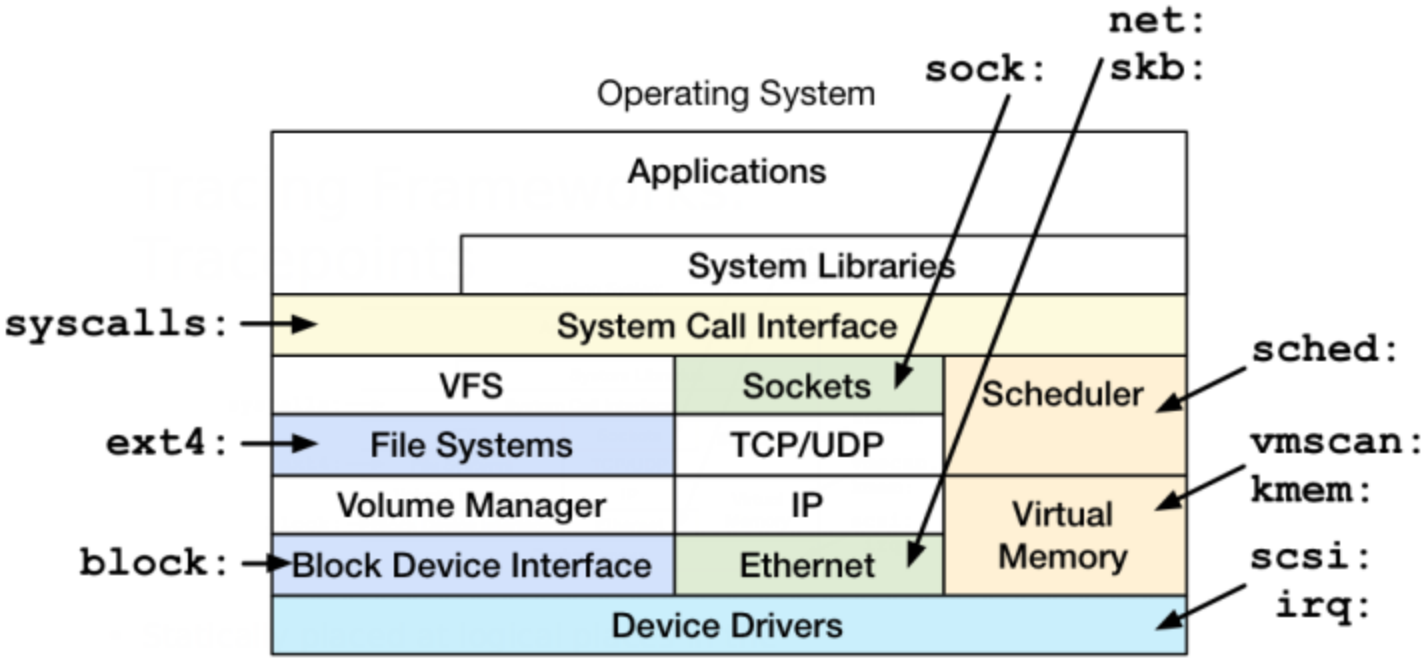
- Top wont catch short lived processes.
  - Perf and Ftrace would.

Tracing

Tracing

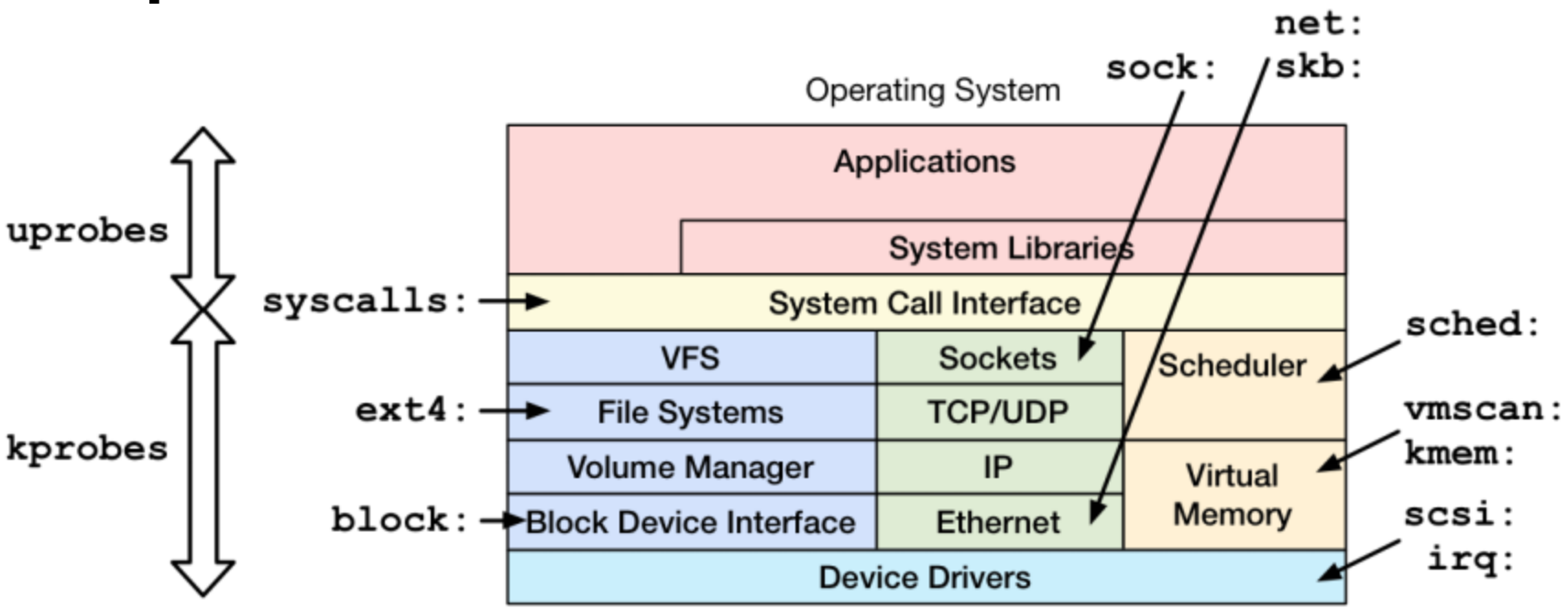
- Objectives:
  - Understand frameworks: tracepoints, kprobes, uprobes
  - Understand mainline tracers: ftrace, perf_events, eBPF
  - Awareness of other tracers: SystemTap, LTTng, ktap, sysdig
  - Awareness of what tracing can accomplish (eg, perf-tools)

Tracing Frameworks: Tracepoints



- Statically placed at logical places in the kernel
- Provides key event details as a “format” string

Tracing Frameworks: + probes



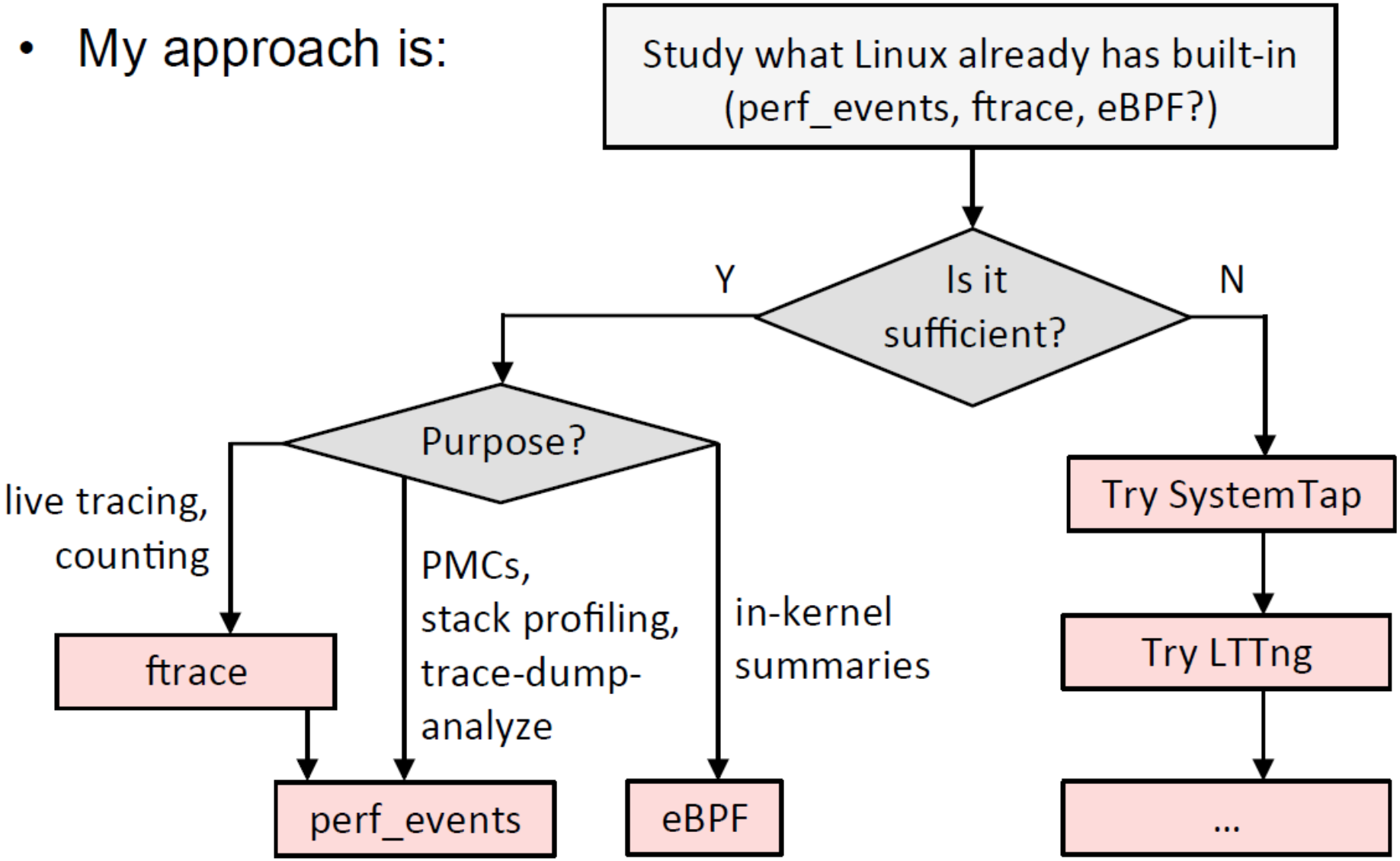
- kprobes: dynamic kernel tracing
  - function calls, returns, line numbers
- uprobes: dynamic user-level tracing

Choosing a Tracer

- Some companies standardize on one tracer
  - eg, SystemTap, LTTng, …

Choosing a Tracer

- My approach is:



ftrace

ftrace

- Added by Steven Rostedt and others since 2.6.27
- Already enabled on our servers (3.2+)
  - CONFIG_FTRACE, CONFIG_FUNCTION_PROFILER, …
  - Use directly via /sys/kernel/debug/tracing:

```
mohit@nomind:~/Work/UltraLowLatency$ sudo ls /sys/kernel/debug/tracing
[sudo] password for mohit:
Sorry, try again.
[sudo] password for mohit:
available_events           events                    per_cpu              set_ftrace_notrace  trace              tracing_on
available_filter_functions free_buffer               printk_formats       set_ftrace_pid      trace_clock        tracing_thresh
available_tracers          function_profile_enabled  README               set_graph_function  trace_marker       uprobe_events
buffer_size_kb             instances                 saved_cmdlines       set_graph_notrace   trace_options      uprobe_profile
buffer_total_size_kb       kprobe_events             saved_cmdlines_size  snapshot            trace_pipe
current_tracer             kprobe_profile            set_event            stack_max_size      trace_stat
dyn_ftrace_total_info      max_graph_depth           set_event_pid        stack_trace         tracing_cpumask
enabled_functions          options                   set_ftrace_filter    stack_trace_filter  tracing_max_latency
```

  - See Linux source: Documentation/trace/ftrace.txt

ftrace Front–Ends

- Steven wrote a front-end: trace-cmd
  - Multi-tool, works well
- "perf-tools" front-ends
  - Both single & multi-purpose, Unix-like
  - Unsupported hacks: see WARNINGs
- perf-tools:
  - single-purpose: iosnoop, iolatency, opensnoop
  - multi-tools: funccount, funcgraph, kprobe

iosnoop

- Block I/O (disk) events with latency:

```
# ./iosnoop –ts
Tracing block I/O. Ctrl-C to end.
STARTs          ENDs            COMM        PID   TYPE DEV     BLOCK      BYTES LATms
5982800.302061 5982800.302679 supervise   1809  W    202,1   17039600   4096   0.62
5982800.302423 5982800.302842 supervise   1809  W    202,1   17039608   4096   0.42
5982800.304962 5982800.305446 supervise   1801  W    202,1   17039616   4096   0.48
5982800.305250 5982800.305676 supervise   1801  W    202,1   17039624   4096   0.43
[…]
```

```
# ./iosnoop –h
USAGE: iosnoop [-hQst] [-d device] [-i iotype] [-p PID] [-n name] [duration]
                 -d device       # device string (eg, "202,1)
                 -i iotype       # match type (eg, '*R*' for all reads)
                 -n name         # process name to match on I/O issue
                 -p PID          # PID to match on I/O issue
                 -Q              # include queueing time in LATms
                 -s              # include start time of I/O (s)
                 -t              # include completion time of I/O (s)
                 -h              # this usage message
                 duration        # duration seconds, and use buffers
[…]
```

iolatency

- Block I/O (disk) latency distributions:

```
# ./iolatency
Tracing block I/O. Output every 1 seconds. Ctrl-C to end.

  >=(ms) .. <(ms)    : I/O       |Distribution                                |
       0 -> 1        : 2104      |########################################|
       1 -> 2        : 280       |######                                  |
       2 -> 4        : 2         |#                                       |
       4 -> 8        : 0         |                                        |
       8 -> 16       : 202       |####                                    |

  >=(ms) .. <(ms)    : I/O       |Distribution                                |
       0 -> 1        : 1144      |########################################|
       1 -> 2        : 267       |#########                               |
       2 -> 4        : 10        |#                                       |
       4 -> 8        : 5         |#                                       |
       8 -> 16       : 248       |#########                               |
      16 -> 32       : 601       |####################                    |
      32 -> 64       : 117       |####                                    |
[…]
```

- Trace open() syscalls showing filenames:

```
# ./opensnoop -t
Tracing open()s. Ctrl-C to end.
TIMEs             COMM          PID      FD FILE
4345768.332626    postgres      23886    0x8 /proc/self/oom_adj
4345768.333923    postgres      23886    0x5 global/pg_filenode.map
4345768.333971    postgres      23886    0x5 global/pg_internal.init
4345768.334813    postgres      23886    0x5 base/16384/PG_VERSION
4345768.334877    postgres      23886    0x5 base/16384/pg_filenode.map
4345768.334891    postgres      23886    0x5 base/16384/pg_internal.init
4345768.335821    postgres      23886    0x5 base/16384/11725
4345768.347911    svstat        24649    0x4 supervise/ok
4345768.347921    svstat        24649    0x4 supervise/status
4345768.350340    stat          24651    0x3 /etc/ld.so.cache
4345768.350372    stat          24651    0x3 /lib/x86_64-linux-gnu/libselinux…
4345768.350460    stat          24651    0x3 /lib/x86_64-linux-gnu/libc.so.6
4345768.350526    stat          24651    0x3 /lib/x86_64-linux-gnu/libdl.so.2
4345768.350981    stat          24651    0x3 /proc/filesystems
4345768.351182    stat          24651    0x3 /etc/nsswitch.conf
[…]
```

tpoint

- Who is creating disk I/O, and of what type?

```
# ./tpoint -H block:block_rq_insert
Tracing block:block_rq_insert. Ctrl-C to end.
# tracer: nop
#
#     TASK-PID    CPU#    TIMESTAMP  FUNCTION
#        | |       |          |         |
 flush-9:0-9318   [013] 1936182.007914: block_rq_insert: 202,16 W 0 () 160186560 + 8 [flush-9:0]
 flush-9:0-9318   [013] 1936182.007939: block_rq_insert: 202,16 W 0 () 280100936 + 8 [flush-9:0]
      java-9469   [014] 1936182.316184: block_rq_insert: 202,1 R 0 () 1319592 + 72 [java]
      java-9469   [000] 1936182.331270: block_rq_insert: 202,1 R 0 () 1125744 + 8 [java]
      java-9469   [000] 1936182.341418: block_rq_insert: 202,1 R 0 () 2699008 + 88 [java]
      java-9469   [000] 1936182.341419: block_rq_insert: 202,1 R 0 () 2699096 + 88 [java]
      java-9469   [000] 1936182.341419: block_rq_insert: 202,1 R 0 () 2699184 + 32 [java]
      java-9469   [000] 1936182.345870: block_rq_insert: 202,1 R 0 () 1320304 + 24 [java]
      java-9469   [000] 1936182.351590: block_rq_insert: 202,1 R 0 () 1716848 + 16 [java]
^C
Ending tracing...
```


- tpoint traces a given tracepoint. -H prints the header.

tpoint -l

```
# ./tpoint -l
block:block_bio_backmerge
block:block_bio_bounce
block:block_bio_complete
block:block_bio_frontmerge
block:block_bio_queue
block:block_bio_remap
block:block_getrq
block:block_plug
block:block_rq_abort
block:block_rq_complete
block:block_rq_insert
block:block_rq_issue
block:block_rq_remap
block:block_rq_requeue
[…]
# ./tpoint –l | wc –l
1257
```

Listing tracepoints

- 1,257 tracepoints for this Linux kernel

funccount

- Count a kernel function call rate:

```
# ./funccount -i 1 'bio_*'
Tracing "bio_*"... Ctrl-C to end.

FUNC                                    COUNT
bio_attempt_back_merge                     26
bio_get_nr_vecs                           361
bio_alloc                                 536
bio_alloc_bioset                          536
bio_endio                                 536
bio_free                                  536
bio_fs_destructor                         536
bio_init                                  536
bio_integrity_enabled                     536
bio_put                                   729
bio_add_page                             1004

[...]
```

Counts are in-kernel, for low overhead

  - -i: set an output interval (seconds), otherwise until Ctrl-C

funcgraph

- Trace a graph of kernel code flow:

```
# ./funcgraph -Htp 5363 vfs_read
Tracing "vfs_read" for PID 5363... Ctrl-C to end.
# tracer: function_graph
#
#     TIME        CPU  DURATION                  FUNCTION CALLS
#      |          |     |   |                     |   |   |   |
4346366.073832 |   0)               |  vfs_read() {
4346366.073834 |   0)               |    rw_verify_area() {
4346366.073834 |   0)               |      security_file_permission() {
4346366.073834 |   0)               |        apparmor_file_permission() {
4346366.073835 |   0)   0.153 us    |          common_file_perm();
4346366.073836 |   0)   0.947 us    |        }
4346366.073836 |   0)   0.066 us    |        __fsnotify_parent();
4346366.073836 |   0)   0.080 us    |        fsnotify();
4346366.073837 |   0)   2.174 us    |      }
4346366.073837 |   0)   2.656 us    |    }
4346366.073837 |   0)               |    tty_read() {
4346366.073837 |   0)   0.060 us    |      tty_paranoia_check();
[…]
```

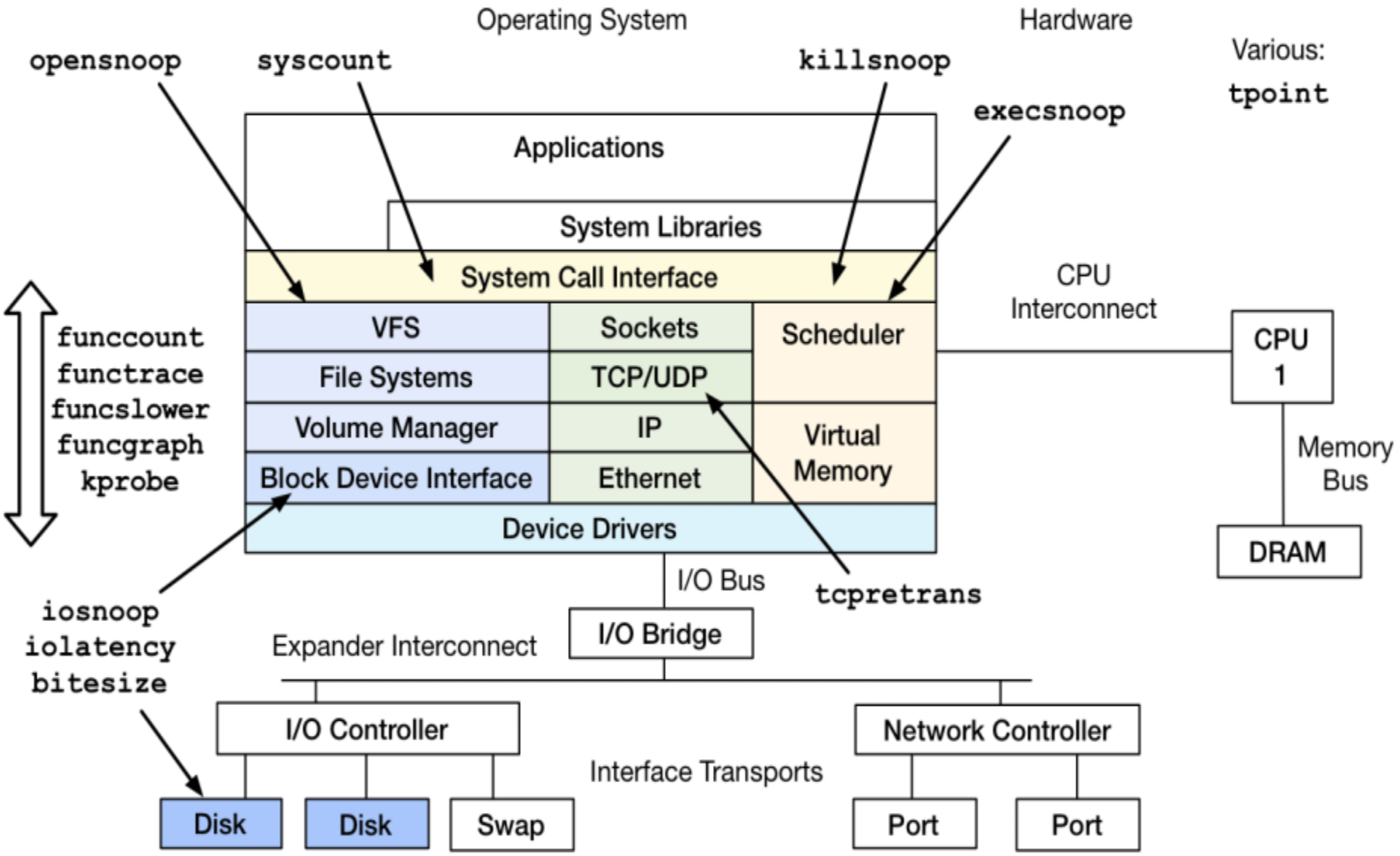
kprobe

- Dynamically trace a kernel function call or return, with variables, and in-kernel filtering:

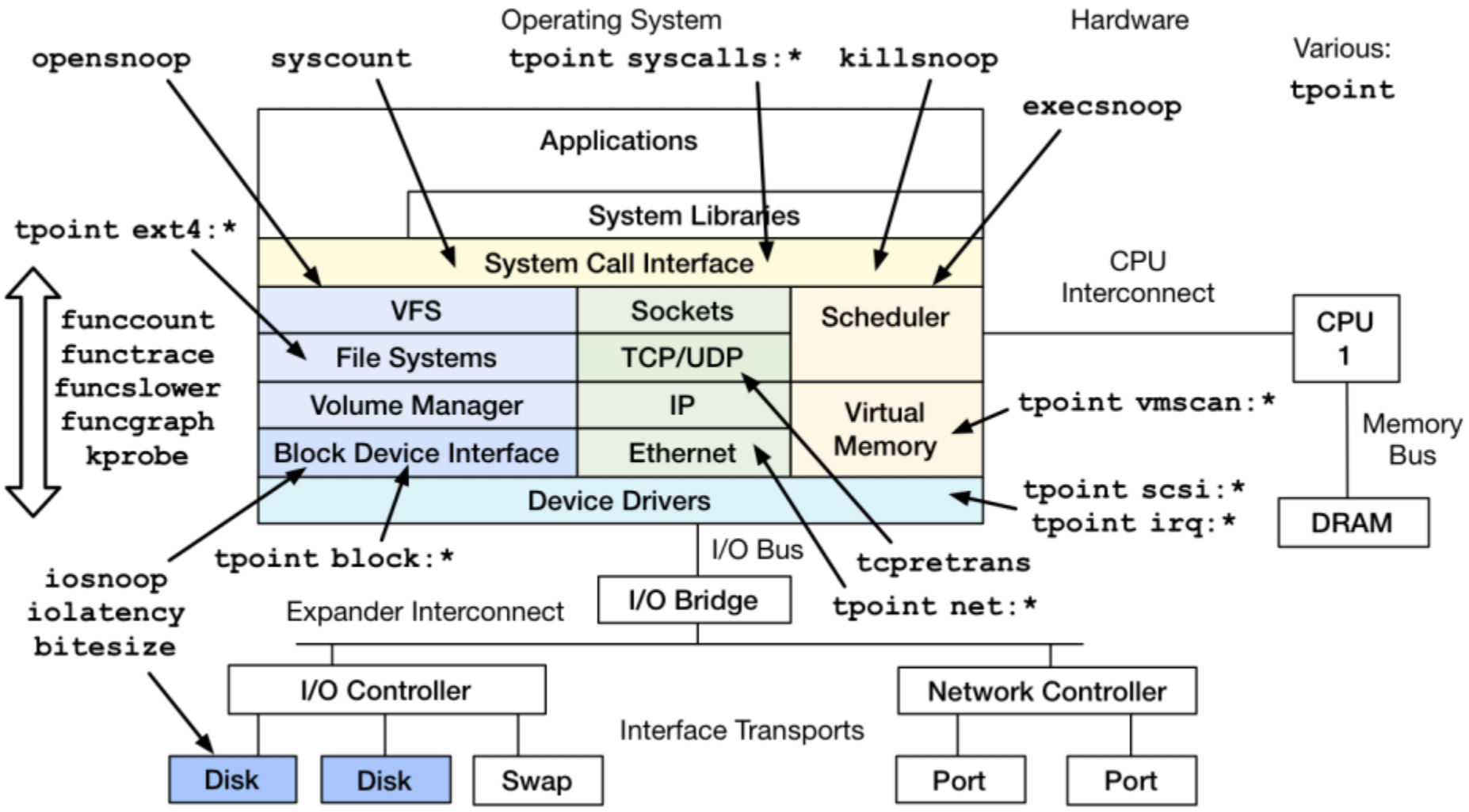
```
# ./kprobe 'p:open do_sys_open filename=+0(%si):string' 'filename ~ "*stat"'
Tracing kprobe myopen. Ctrl-C to end.
        postgres-1172  [000] d... 6594028.787166: open: (do_sys_open
+0x0/0x220) filename="pg_stat_tmp/pgstat.stat"
        postgres-1172  [001] d... 6594028.797410: open: (do_sys_open
+0x0/0x220) filename="pg_stat_tmp/pgstat.stat"
        postgres-1172  [001] d... 6594028.797467: open: (do_sys_open
+0x0/0x220) filename="pg_stat_tmp/pgstat.stat”
^C
Ending tracing...
```

- Add -s for stack traces; -p for PID filter in-kernel.
- Quickly confirm kernel behavior; eg: did a tunable take effect?

perf–tools (so far…)



perf–tools (so far…)



perf_events:ref to HPC

eBPF:ref to eBPF

Other Tracers

"Lightweight SystemTap:ref to Systemtap

ktap

- Was a very promising new Linux tracer:
  - Sampling, static & dynamic tracing
  - Lightweight, simple. Uses bytecode.
  - Suited for embedded devices
- Development suspended while eBPF integrates
- Will it restart?

sysdig

- sysdig: Innovative new tracer. Simple expressions:

```
sysdig fd.type=file and evt.failed=true
sysdig evt.type=open and fd.name contains /etc
sysdig -p"%proc.name %fd.name" "evt.type=accept and proc.name!=httpd"
```

- Replacement for strace? (or “perf trace” will)
- Programmable “chisels”. Eg, one of mine:

```
# sysdig -c fileslower 1
TIME                     PROCESS      TYPE      LAT(ms) FILE
2014-04-13 20:40:43.973  cksum        read            2 /mnt/partial.0.0
2014-04-13 20:40:44.187  cksum        read            1 /mnt/partial.0.0
2014-04-13 20:40:44.689  cksum        read            2 /mnt/partial.0.0
[…]
```

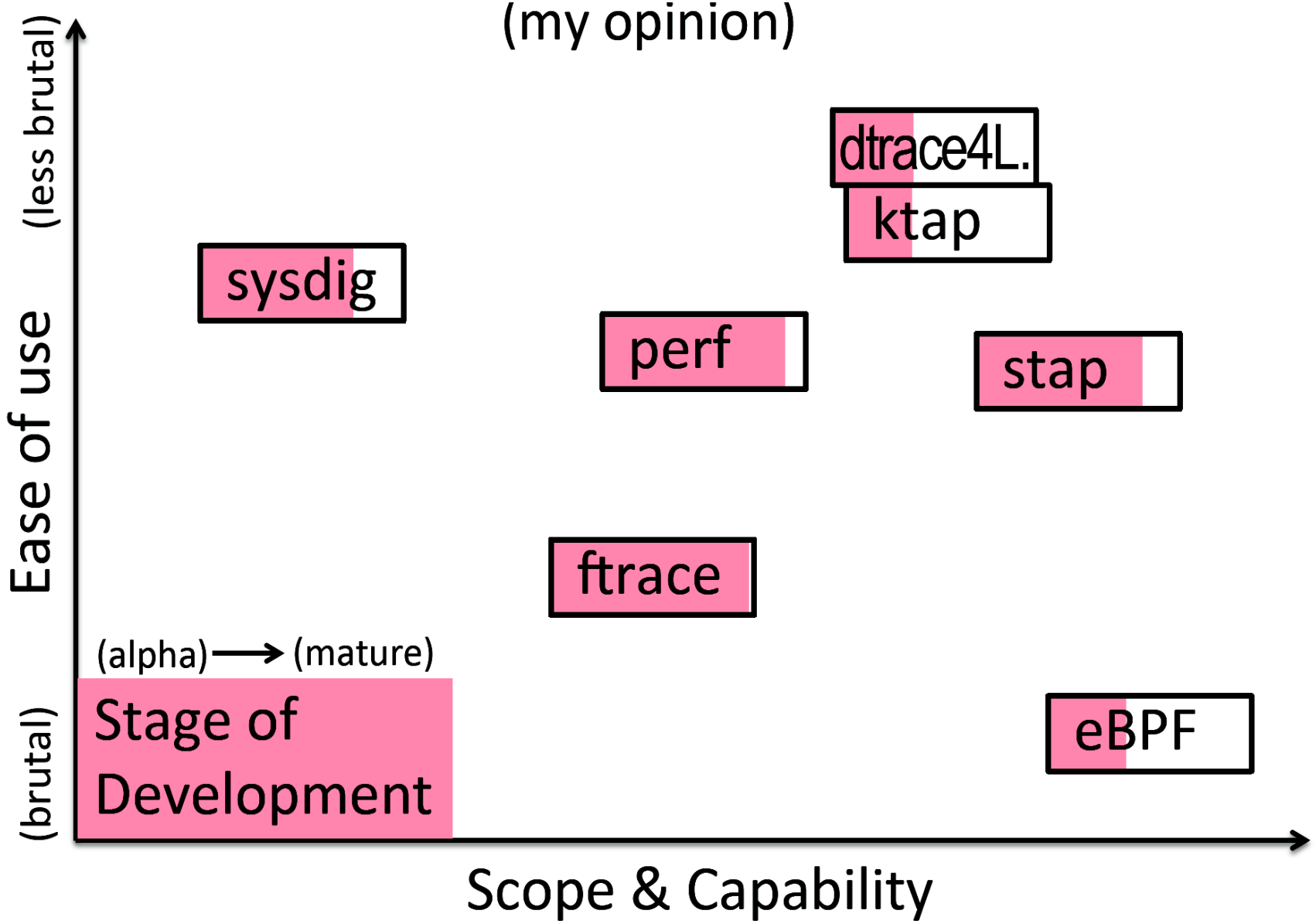
- Currently syscalls and user-level processing only
  - I'm not sure it can be optimized enough for kernel tracing, unless it adopts eBPF for in-kernel processing & summaries

Present & Future

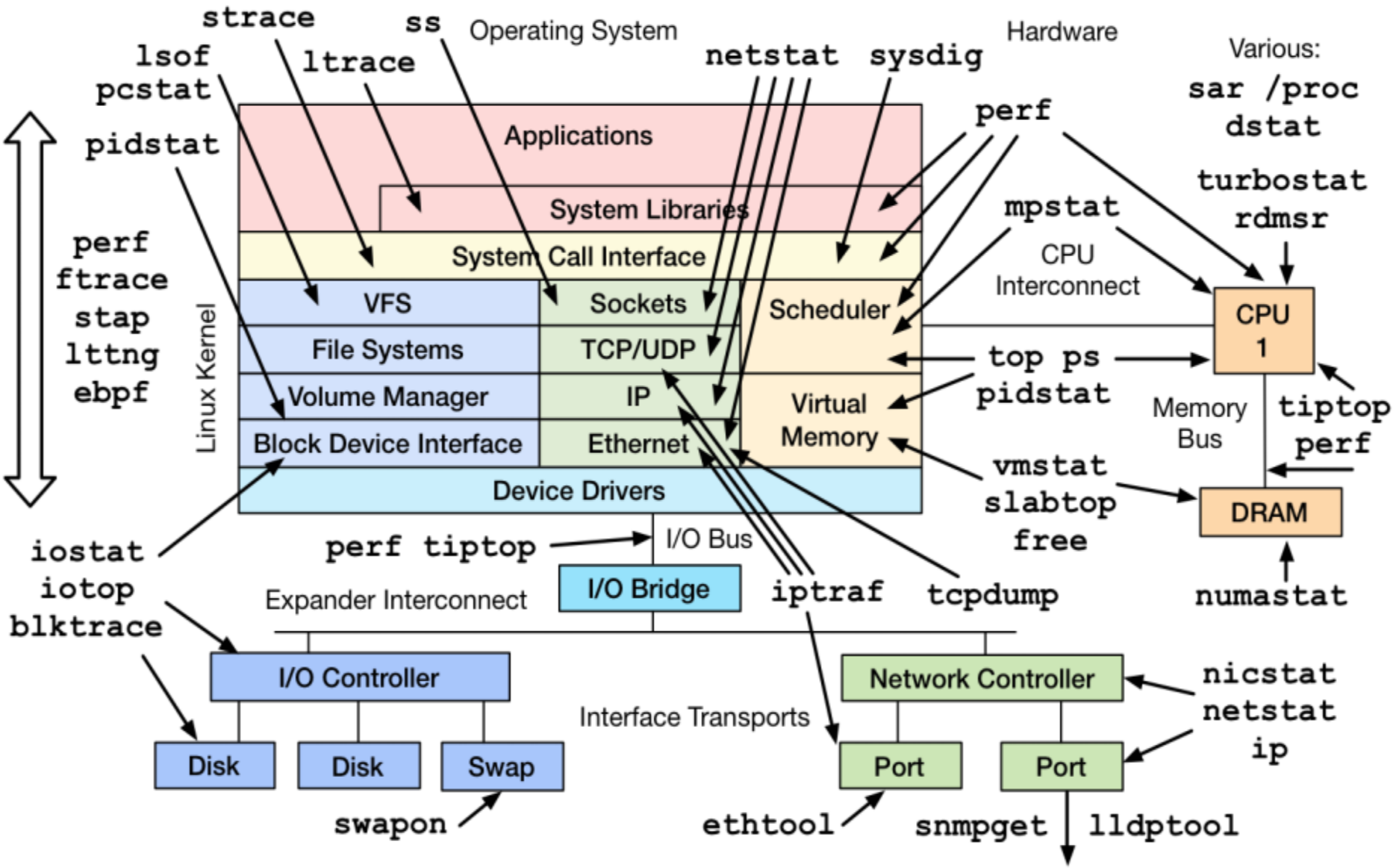
- Present:
  - ftrace & perf_events solving many needs today:
    - PMC profiling, stack profiling, tracepoint & dynamic tracing, …
- Expected Future:
  - eBPF for kernel summaries & advanced programs
  - eBPF perf integration to improve ease of use
- Possible Future:
  - eBPF high level language (ktap?)
  - ftrace/eBPF integration
  - Other tracer eBPF integration (SystemTap, LTTng, sysdig?)
  - One of the other tracers going mainline?

The Tracing Landscape, May 2015

(my opinion)



In Summary...


strace
ss
Operating System
Hardware
lsof
pcstat
ltrace
netstat
sysdig
Various:
sar /proc
dstat
pidstat
perf
Applications
System Libraries
turbostat
rdmsr
mpstat
System Call Interface
CPU
Interconnect
perf
ftrace
stap
lttng
ebpf
Linux Kernel
VFS
Sockets
Scheduler
CPU
1
File Systems
TCP/UDP
top ps
pidstat
Volume Manager
IP
Virtual
Memory
Memory
Bus
tiptop
perf
Block Device Interface
Ethernet
vmstat
slabtop
free
Device Drivers
DRAM
iostat
iotop
blktrace
perf tiptop
I/O Bus
I/O Bridge
iptraf
tcpdump
numastat
Expander Interconnect
I/O Controller
Network Controller
nicstat
netstat
ip
Interface Transports
Disk
Disk
Swap
Port
Port
swapon
ethtool
snmpget
lldptool

Methodologies Summary

- Objectives:
  - Recognize the Streetlight Anti-Method
  - Perform the Workload Characterization Method
  - Perform the USE Method
  - Be aware of other methodologies

Try to start with the questions (methodology), to help guide your use of the tools

Tools Summary

- Objectives:
  - Perform the USE Method for resource utilization
  - Perform Workload Characterization for disks, network
  - Have exposure to various observability tools:
    - Basic: vmstat, iostat, mpstat, ps, top, …
    - Intermediate: tcpdump, netstat, nicstat, pidstat, sar, …
    - Advanced: ss, slaptop, perf_events, …
  - Perform Active Benchmarking
  - Understand tuning risks
  - Perform Static Performance Tuning