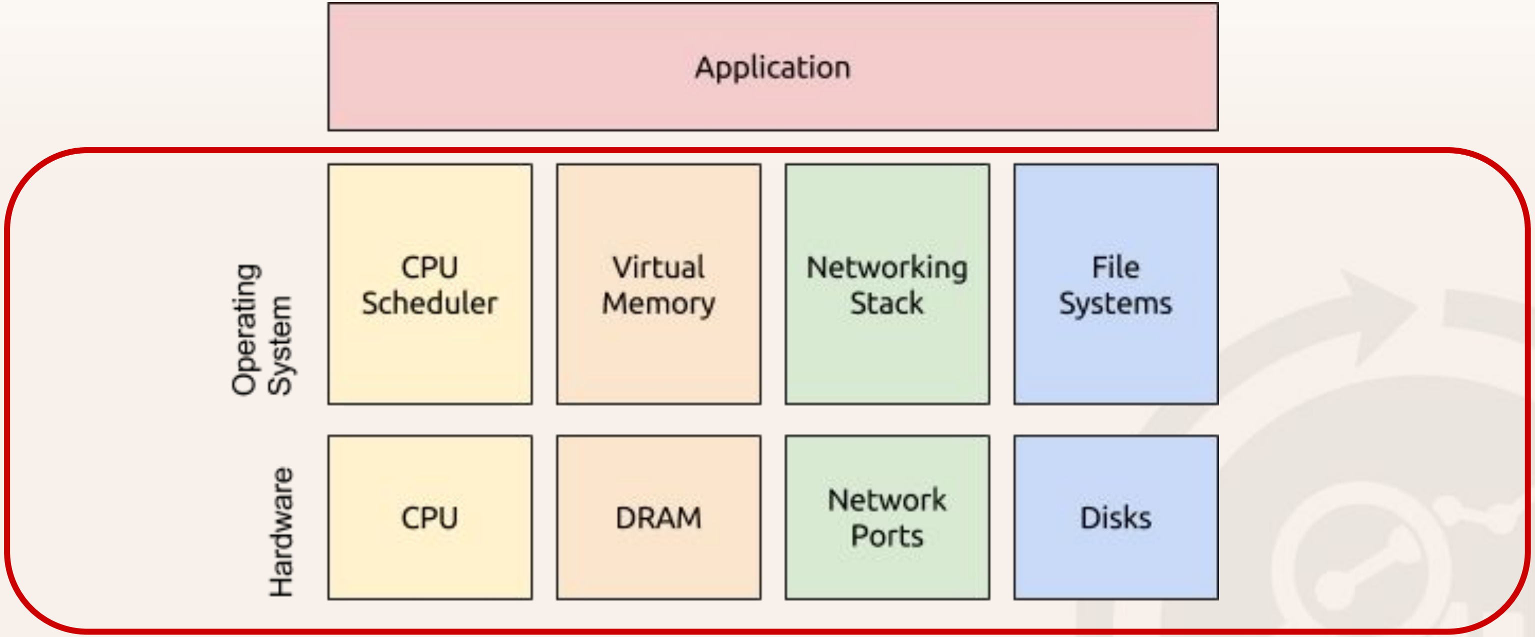
Linux System Monitoring with eBPF

DevOpsDays Zurich, 2018-05-03

Heinrich Hartmann

System Monitoring is about Kernel & Hardware



Best Practice: The USE Method

https://www.circonus.com/2017/08/system-monitoring-with-the-use-dashboard

Utilization

Saturation

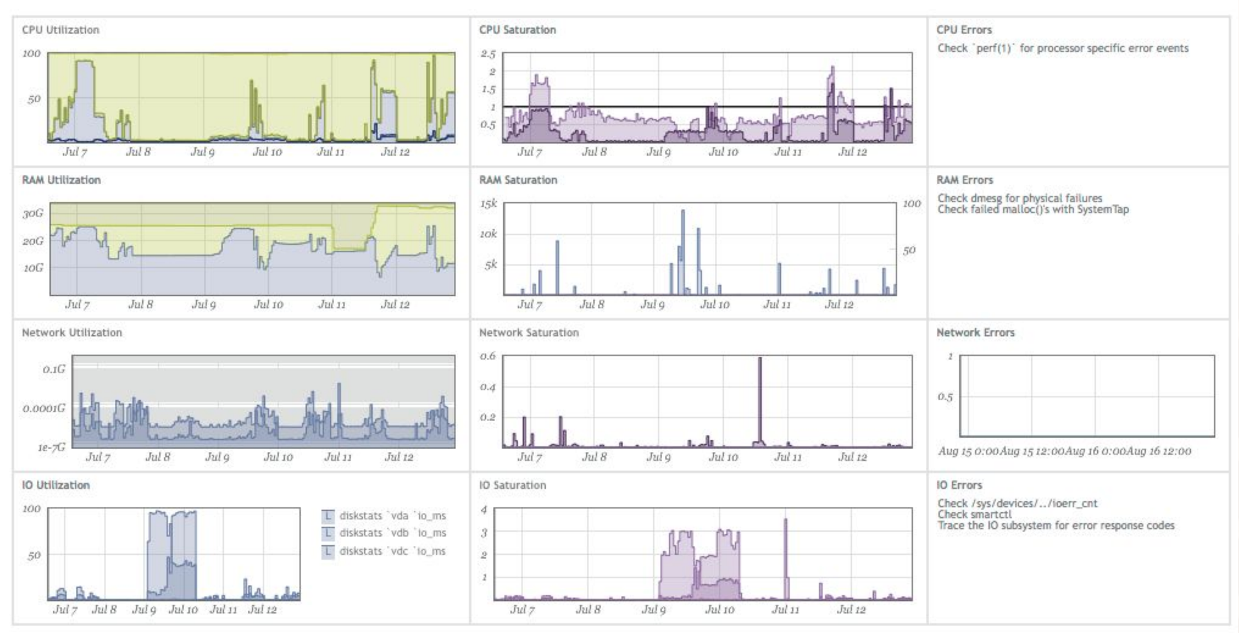
Errors

CPU

Memory

Network

Disks



CPU Errors

Check `perf(1)` for processor specific error events

RAM Errors

Check dmesg for physical failures
Check failed malloc()'s with SystemTap

IO Errors

Check /sys/devices/../ioerr_cnt
Check smartctl
Trace the IO subsystem for error response codes

Best Practice: The USE Method

https://www.circonus.com/2017/08/system-monitoring-with-the-use-dashboard

Utilization

Saturation

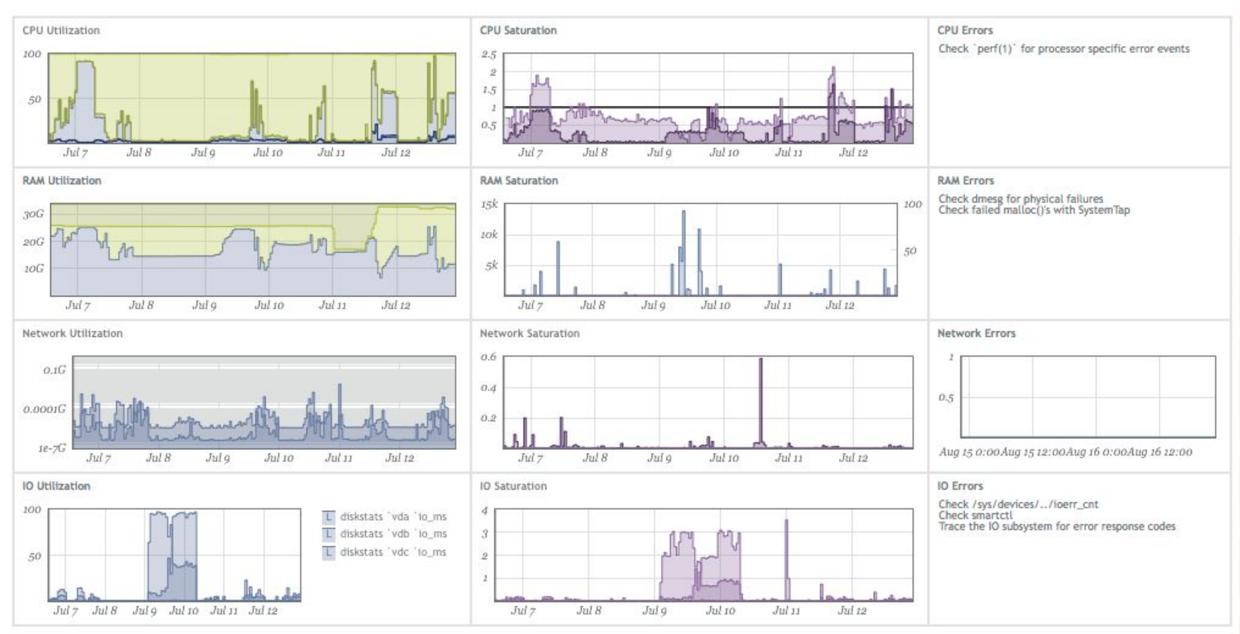
Errors

CPU

Memory

Network

Disks



CPU Utilization

CPU Saturation

CPU Errors

Check `perf(1)` for processor specific error events

RAM Utilization

RAM Saturation

RAM Errors

Check dmesg for physical failures
Check failed malloc()'s with SystemTap

Network Utilization

Network Saturation

Network Errors

IO Utilization

diskstats `vda `io_ms
diskstats `vdb `io_ms
diskstats `vdc `io_ms

IO Saturation

IO Errors

Check /sys/devices/../ioerr_cnt
Check smartctl
Trace the IO subsystem for error response codes

IRONdb POWERED BY CIRCONUS

CIRCONUS

Lot’s of Unknowns remaining

https://www.circonus.com/2017/08/system-monitoring-with-the-use-dashboard

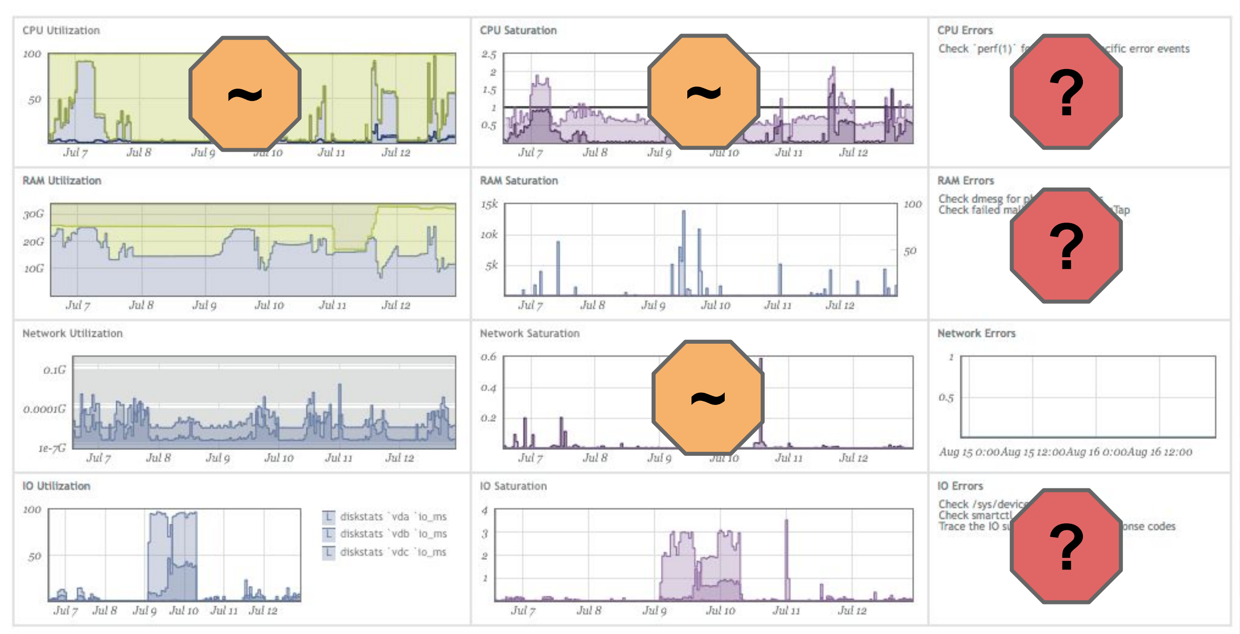
Utilization Saturation Errors

CPU

Memory

Network

Disks



CPU Errors

Check `perf(1)` f… …cific error events

RAM Errors

Check dmesg for p…

Check failed mal… …Tap

IO Errors

Check /sys/devic…

Check smartctl…

Trace the IO su… …onse codes

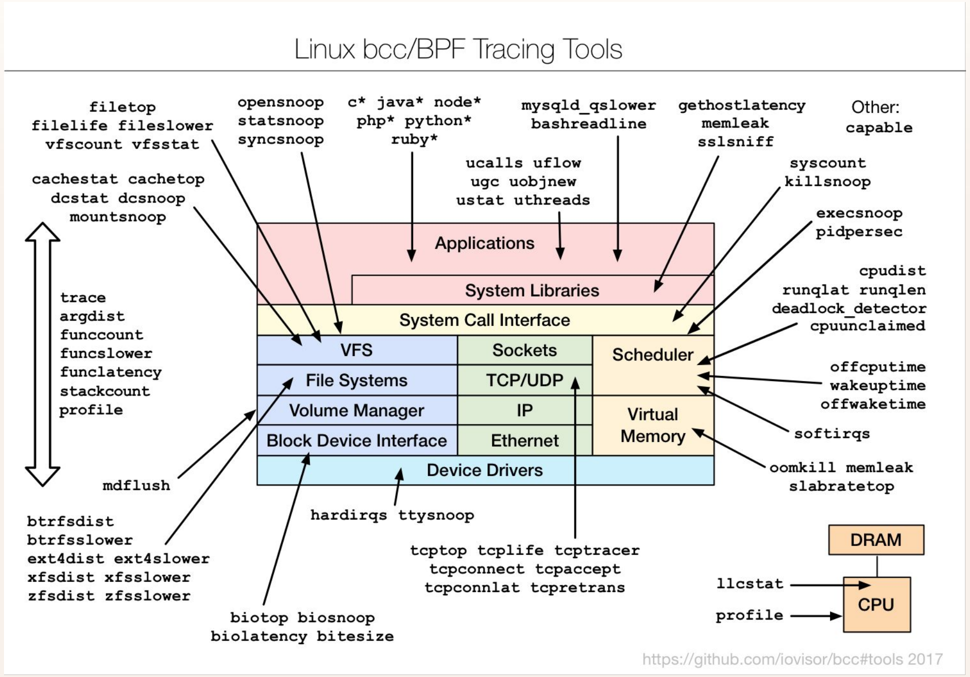
IRONdb POWERED BY CIRCONUS

Heinrich.Hartmann@Circonus.com

CIRCONUS

eBPF allows unparalleled insights

https://github.com/iovisor/bcc



Linux bcc/BPF Tracing Tools

filetop
filelife fileslower
vfscount vfsstat

opensnoop
statsnoop
syncsnoop

c* java* node*
php* python*
ruby*

mysqld_qslower
bashreadline

gethostlatency
memleak
sslsniff

Other:
capable

cachestat cachetop
dcstat dcsnoop
mountsnoop

ucalls uflow
ugc uobjnew
ustat uthreads

syscount
killsnoop

execsnoop
pidpersec

Applications

System Libraries

System Call Interface

VFS | Sockets | Scheduler

File Systems | TCP/UDP

Volume Manager | IP | Virtual Memory

Block Device Interface | Ethernet

Device Drivers

trace
argdist
funccount
funcslower
funclatency
stackcount
profile

cpudist
runqlat runqlen
deadlock_detector
cpuunclaimed

offcputime
wakeuptime
offwaketime

softirqs

oomkill memleak
slabratetop

mdflush

btrfsdist
btrfsslower
ext4dist ext4slower
xfsdist xfsslower
zfsdist zfsslower

hardirqs ttysnoop

tcptop tcplife tcptracer
tcpconnect tcpaccept
tcpconnlat tcpretrans

biotop biosnoop
biolatency bitesize

DRAM

llcstat

profile

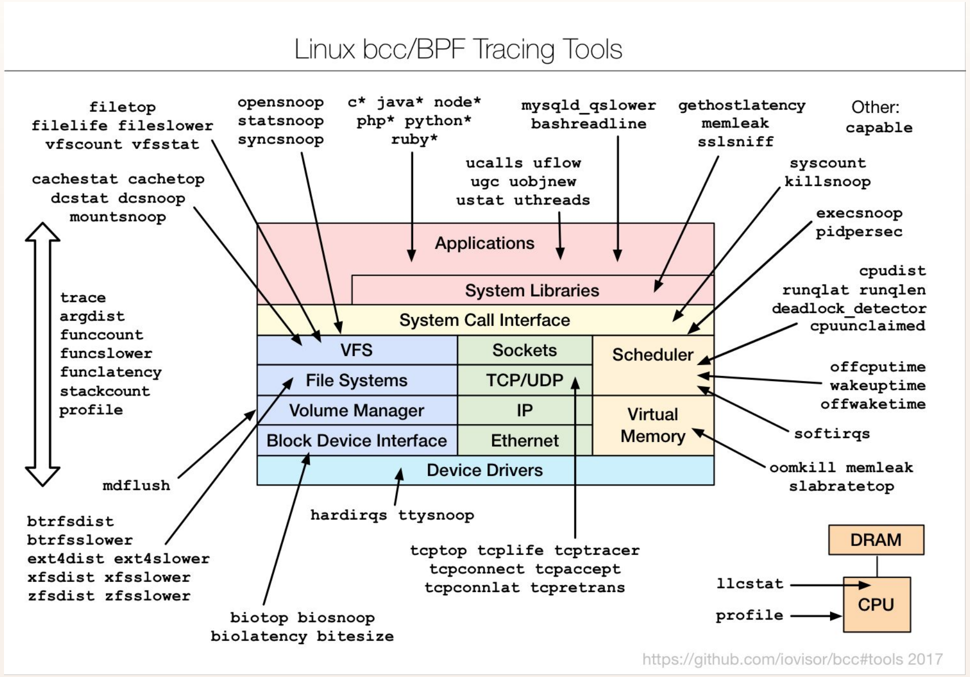
CPU

https://github.com/iovisor/bcc#tools 2017

Credits:
- Brendan Gregg @ Netflix (Sun)
- Sasha Goldshtein @ Sela, Microsoft
- Brenden Blanco @ VMWare
- Linus Torvalds, et. al.

eBPF allows unparalleled insights

https://github.com/iovisor/bcc



Linux bcc/BPF Tracing Tools

filetop
filelife fileslower
vfscount vfsstat

opensnoop
statsnoop
syncsnoop

c* java* node*
php* python*
ruby*

mysqld_qslower
bashreadline

gethostlatency
memleak
sslsniff

Other:
capable

ucalls uflow
ugc uobjnew
ustat uthreads

cachestat cachetop
dcstat dcsnoop
mountsnoop

syscount
killsnoop

execsnoop
pidpersec

cpudist
runqlat runqlen
deadlock_detector
cpuunclaimed

trace
argdist
funccount
funcslower
funclatency
stackcount
profile

Applications

System Libraries

System Call Interface

VFS

Sockets

Scheduler

File Systems

TCP/UDP

Volume Manager

IP

Virtual
Memory

Block Device Interface

Ethernet

Device Drivers

offcputime
wakeuptime
offwaketime

softirqs

oomkill memleak
slabratetop

mdflush

btrfsdist
btrfsslower
ext4dist ext4slower
xfsdist xfsslower
zfsdist zfsslower

hardirqs ttysnoop

tcptop tcplife tcptracer
tcpconnect tcpaccept
tcpconnlat tcpretrans

biotop biosnoop
biolatency bitesize

DRAM

llcstat

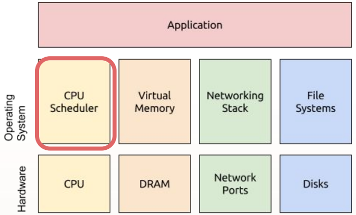
profile

CPU

https://github.com/iovisor/bcc#tools 2017

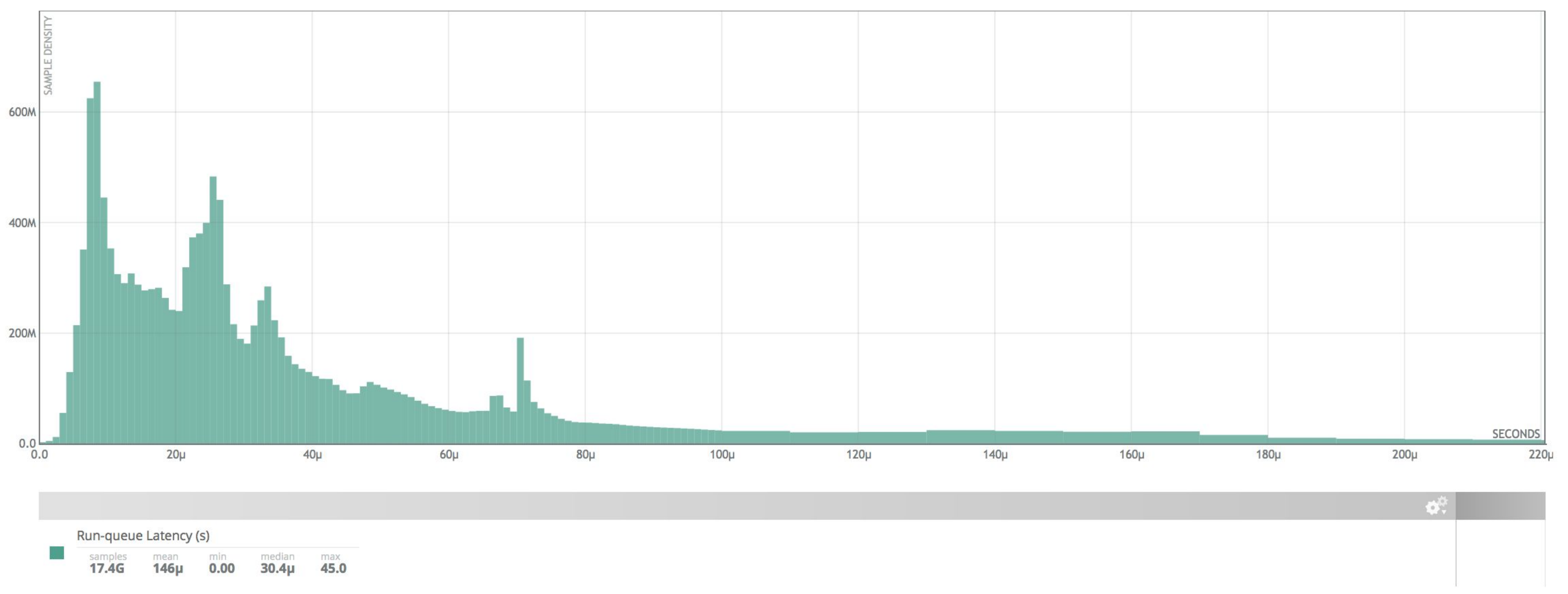
Credits:
- Brendan Gregg @ Netflix (Sun)
- Sasha Goldshtein @ Sela, Microsoft
- Brenden Blanco @ VMWare
- Linus Torvalds, et. al.

CPU: Scheduling Latency


Application

Operating System: CPU Scheduler | Virtual Memory | Networking Stack | File Systems

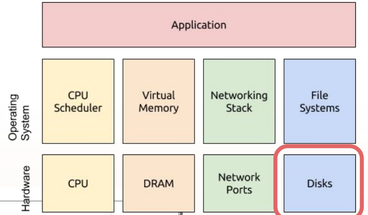
Hardware: CPU | DRAM | Network Ports | Disks



🟩 Run-queue Latency (s)

samples	mean	min	median	max
17.4G	146μ	0.00	30.4μ	45.0

Disk: Block-I/O Latency



Application

Operating System

CPU Scheduler

Virtual Memory

Networking Stack

File Systems

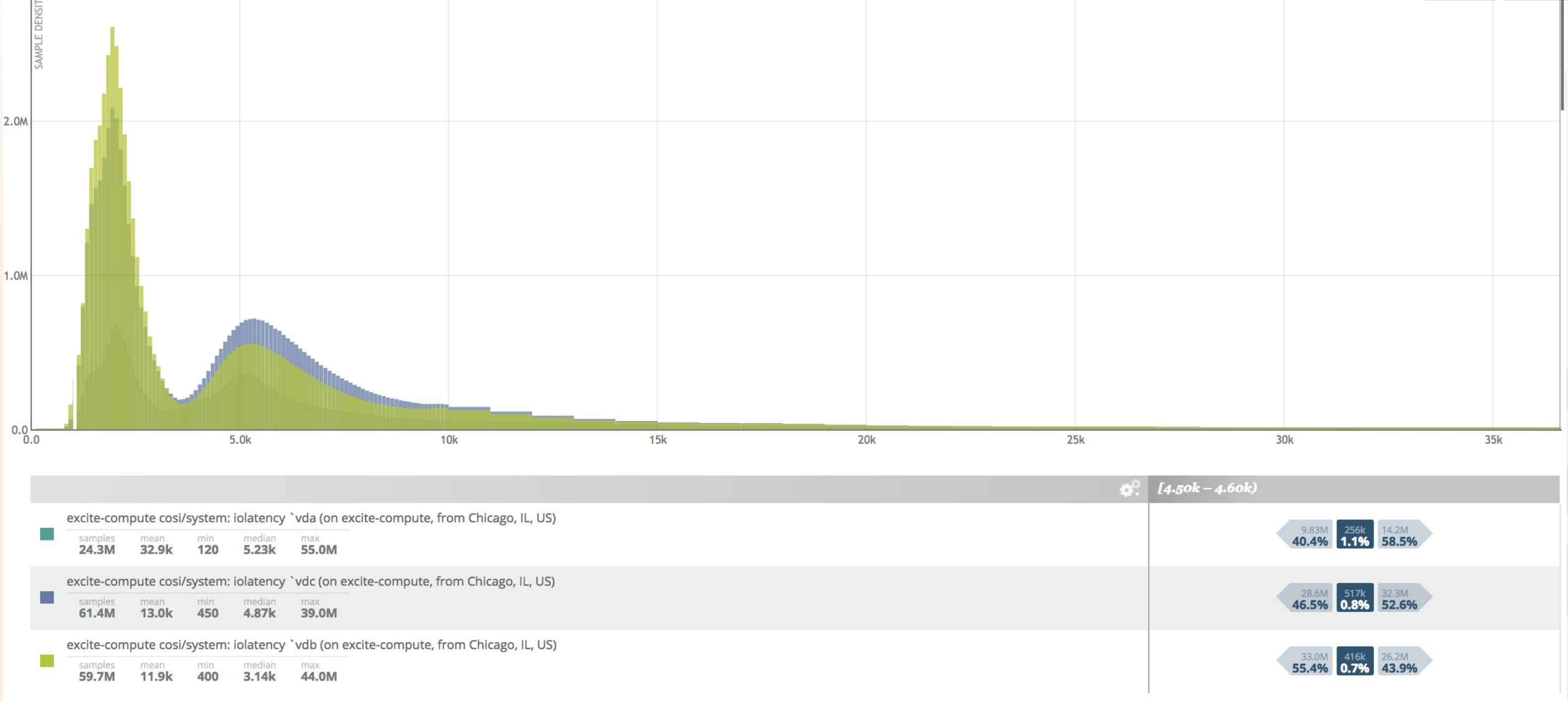
Hardware

CPU

DRAM

Network Ports

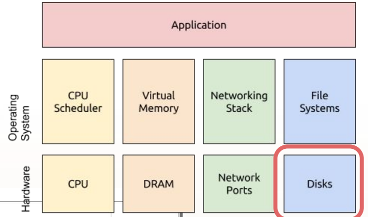
Disks

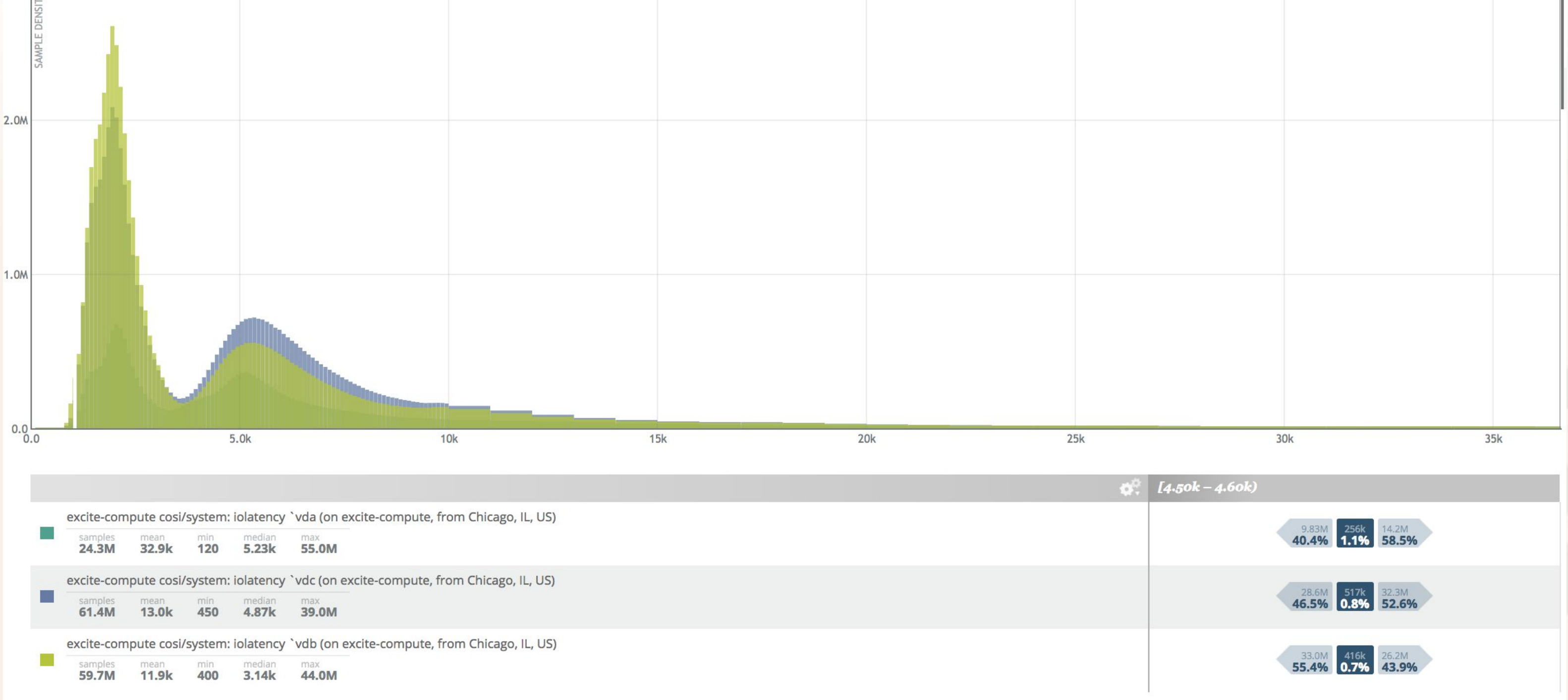


[4.50k – 4.60k)

		samples	mean	min	median	max			
🟩	excite-compute cosi/system: iolatency `vda (on excite-compute, from Chicago, IL, US)	24.3M	32.9k	120	5.23k	55.0M	9.83M 40.4%	256k 1.1%	14.2M 58.5%
🟦	excite-compute cosi/system: iolatency `vdc (on excite-compute, from Chicago, IL, US)	61.4M	13.0k	450	4.87k	39.0M	28.6M 46.5%	517k 0.8%	32.3M 52.6%
🟩	excite-compute cosi/system: iolatency `vdb (on excite-compute, from Chicago, IL, US)	59.7M	11.9k	400	3.14k	44.0M	33.0M 55.4%	416k 0.7%	26.2M 43.9%

Disk: Block-I/O Latency





🟩 excite-compute cosi/system: iolatency `vda (on excite-compute, from Chicago, IL, US)

samples	mean	min	median	max
24.3M	32.9k	120	5.23k	55.0M

9.83M 40.4% | 256k 1.1% | 14.2M 58.5%

🟦 excite-compute cosi/system: iolatency `vdc (on excite-compute, from Chicago, IL, US)

samples	mean	min	median	max
61.4M	13.0k	450	4.87k	39.0M

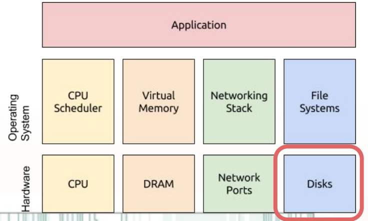
28.6M 46.5% | 517k 0.8% | 32.3M 52.6%

🟩 excite-compute cosi/system: iolatency `vdb (on excite-compute, from Chicago, IL, US)

samples	mean	min	median	max
59.7M	11.9k	400	3.14k	44.0M

33.0M 55.4% | 416k 0.7% | 26.2M 43.9%

Disk: Block-I/O Latency over time



Application

Operating System

CPU Scheduler

Virtual Memory

Networking Stack

File Systems

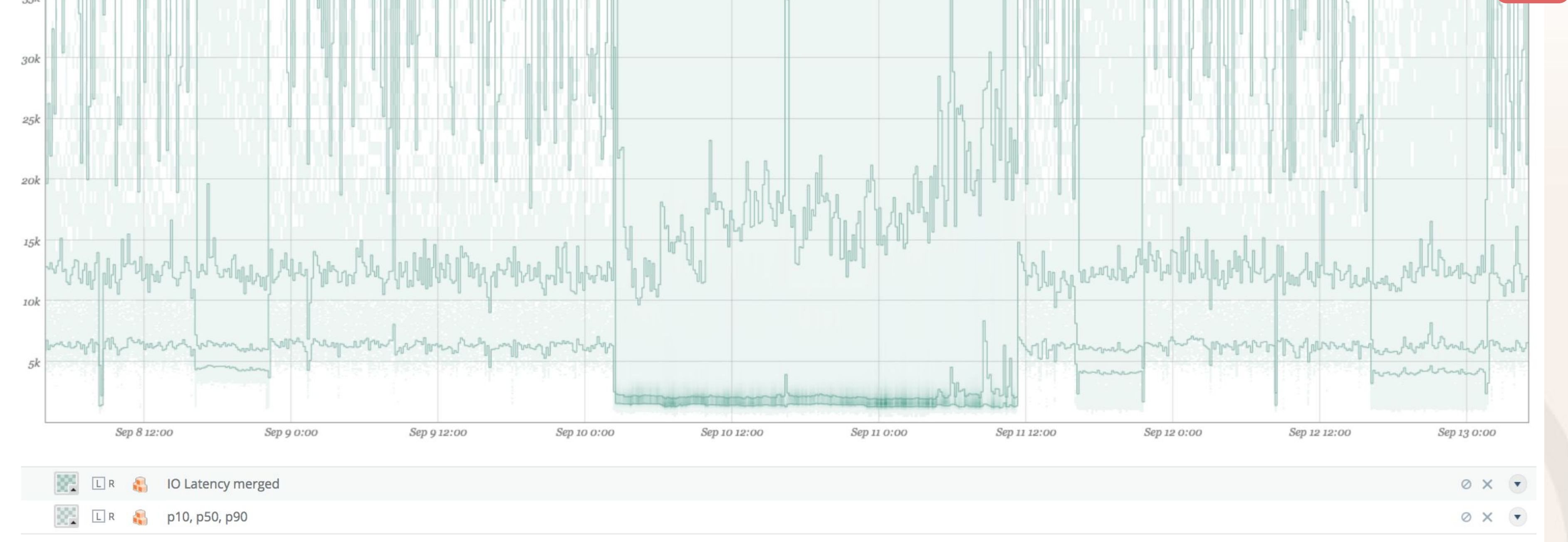
Hardware

CPU

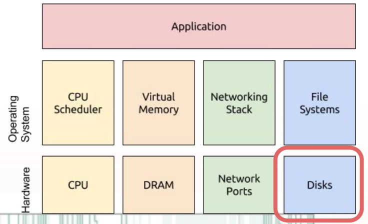
DRAM

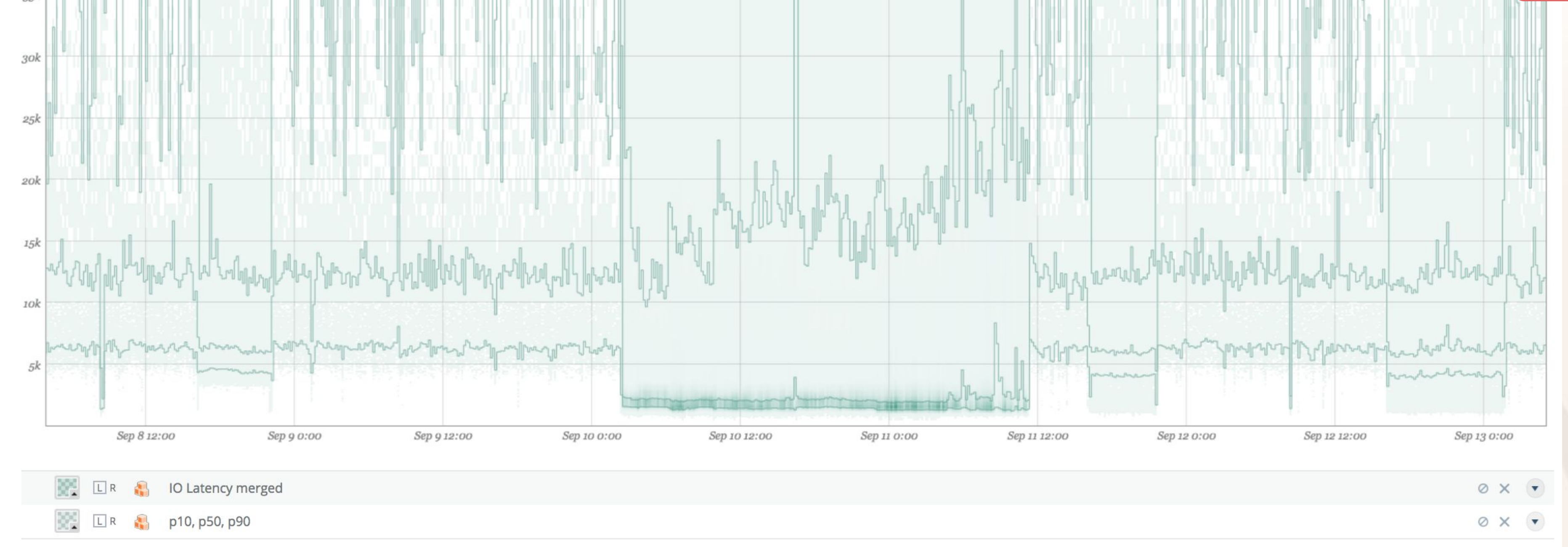
Network Ports

Disks



Disk: Block-I/O Latency over time



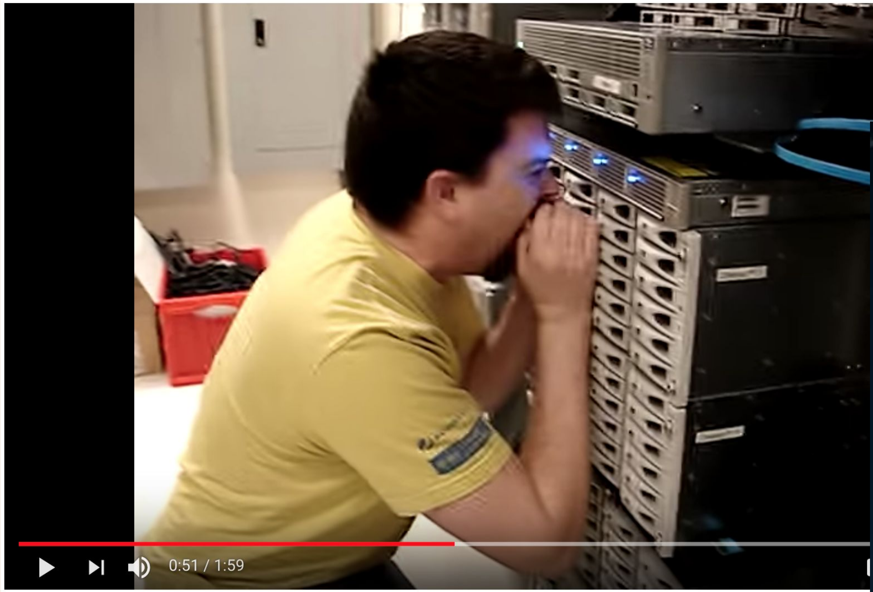


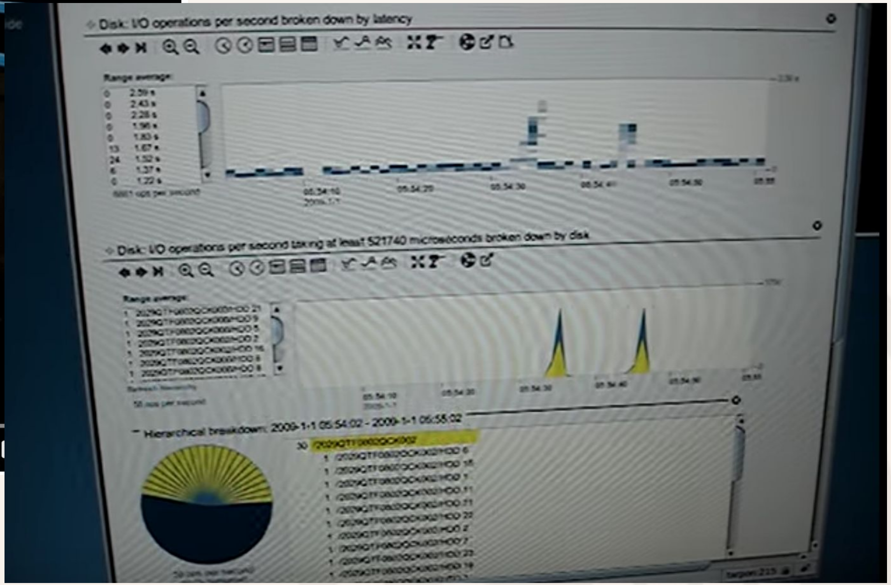
IO Latency merged

p10, p50, p90

Don’t shout in the Datacenter

Brendan Gregg (2008) https://www.youtube.com/watch?v=tDacjrSCeq4





Disk: I/O operations per second broken down by latency

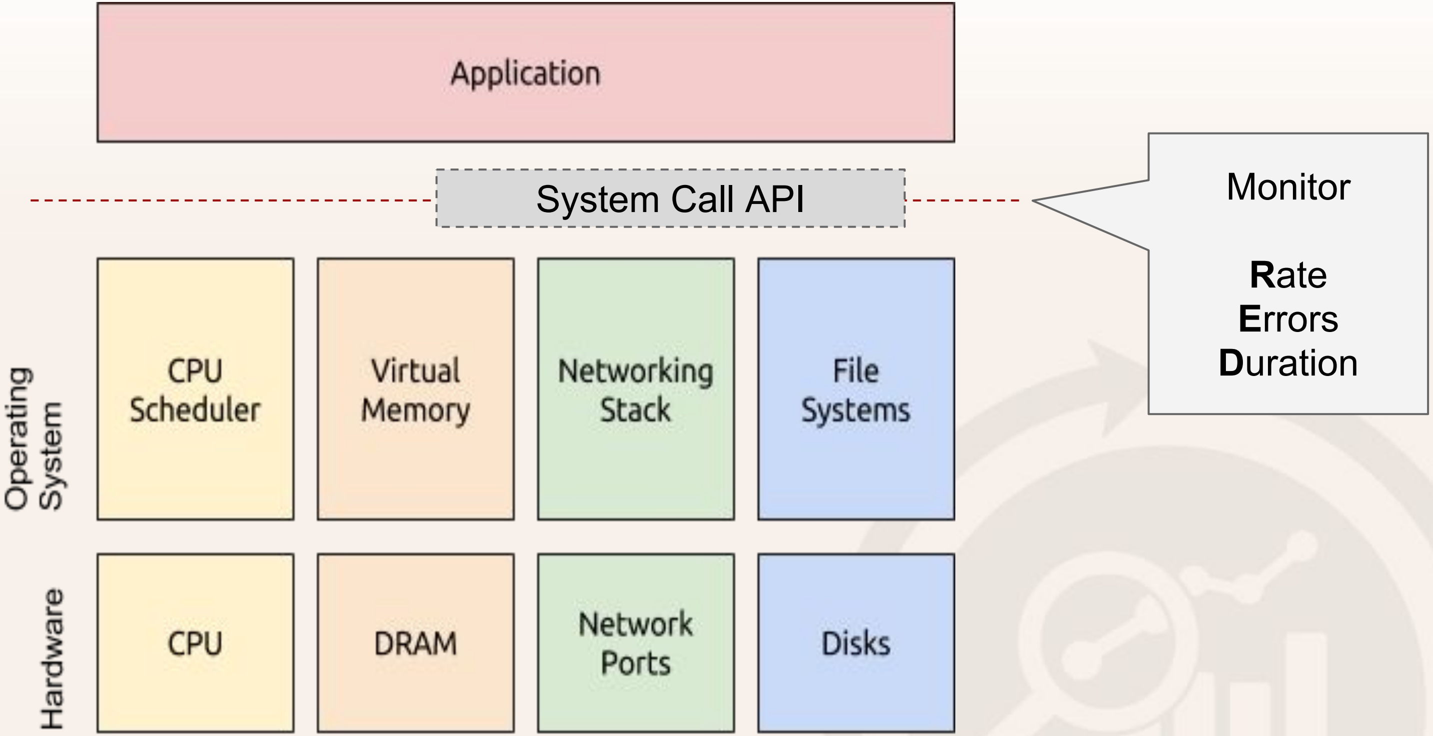
Range average:

Disk: I/O operations per second taking at least 521740 microseconds broken down by disk

Range average:

Hierarchical breakdown: 2009-1-1 05:54:02 - 2009-1-1 05:55:02

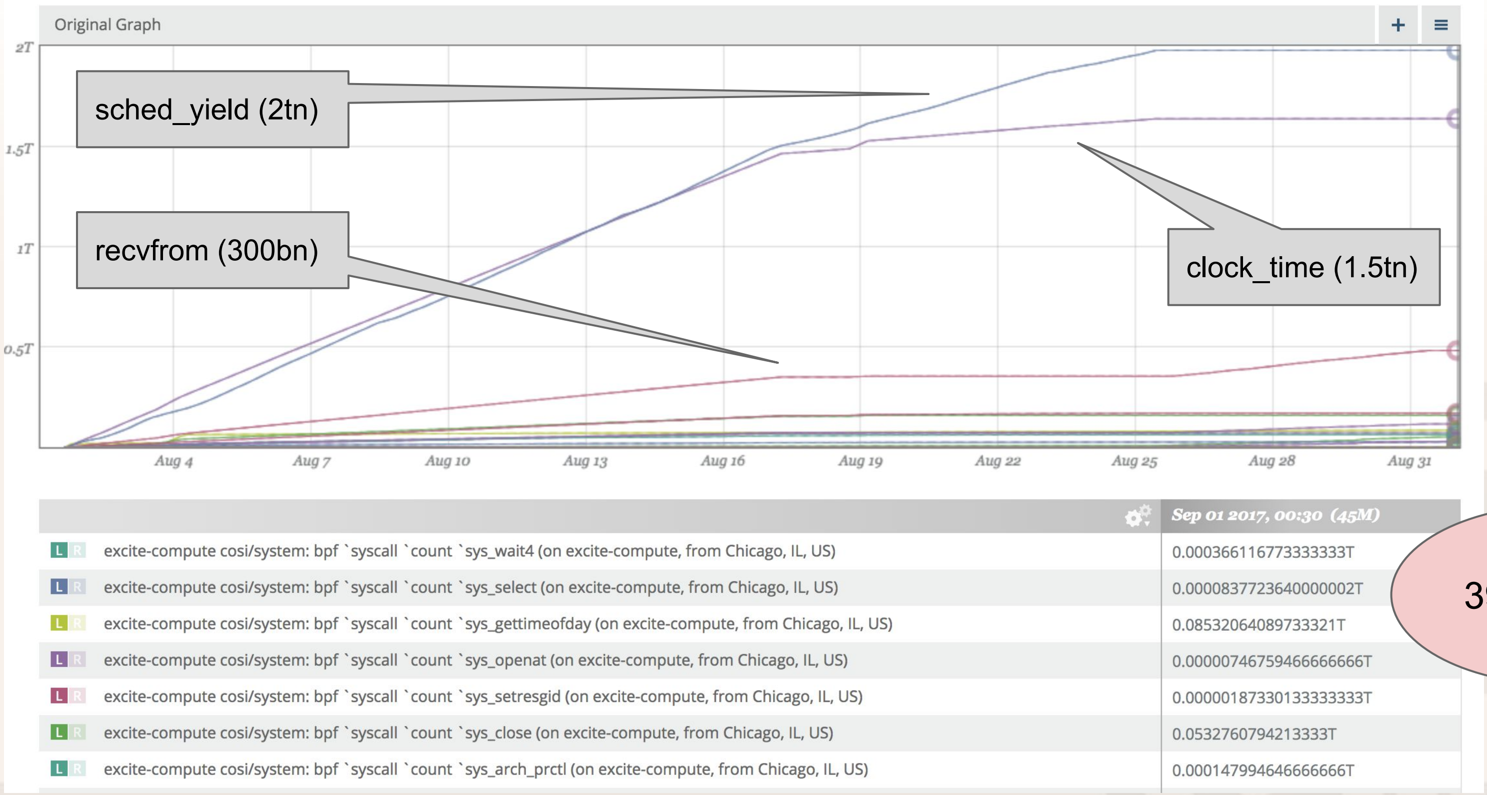
System Calls: The Kernel API



Monitor

Rate
Errors
Duration

Syscalls: Rate / Count



sched_yield (2tn)

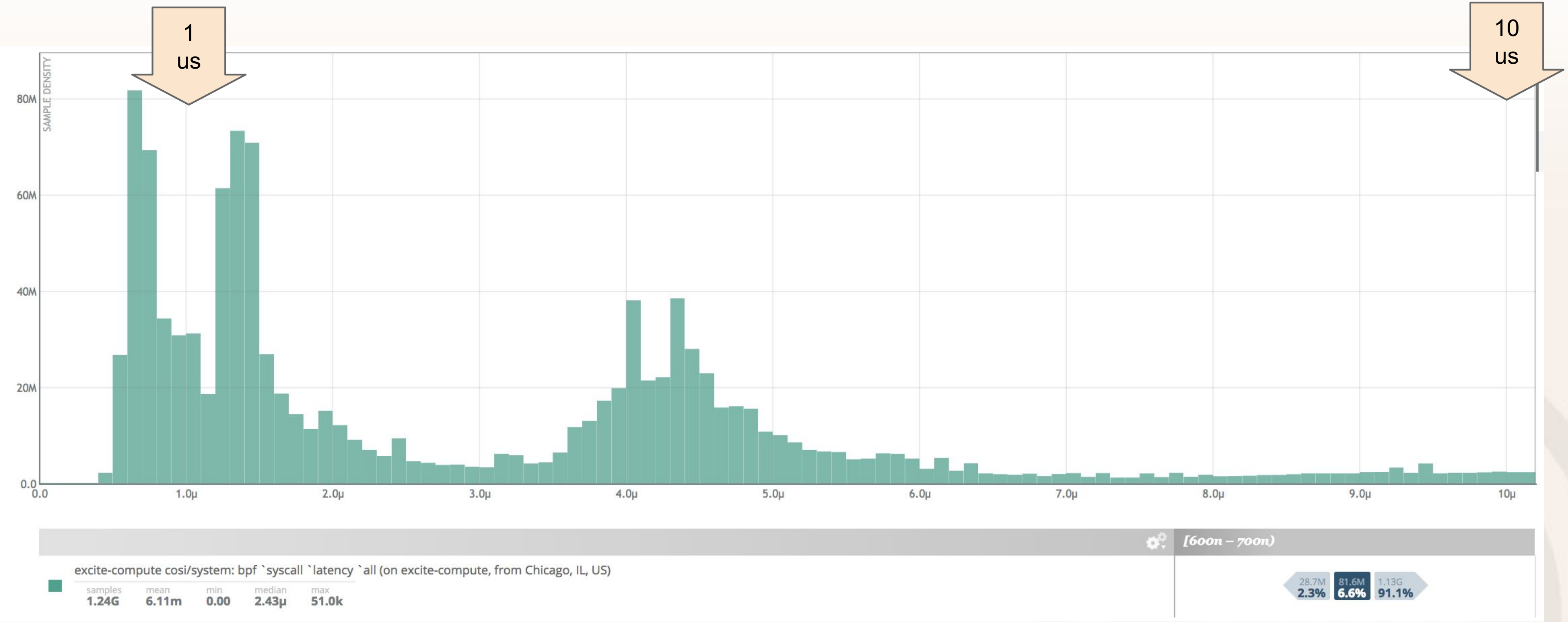
recvfrom (300bn)

clock_time (1.5tn)

	Sep 01 2017, 00:30 (45M)
L R excite-compute cosi/system: bpf `syscall `count `sys_wait4 (on excite-compute, from Chicago, IL, US)	0.000366116773333333T
L R excite-compute cosi/system: bpf `syscall `count `sys_select (on excite-compute, from Chicago, IL, US)	0.0000837723640000002T
L R excite-compute cosi/system: bpf `syscall `count `sys_gettimeofday (on excite-compute, from Chicago, IL, US)	0.0853206408973321T
L R excite-compute cosi/system: bpf `syscall `count `sys_openat (on excite-compute, from Chicago, IL, US)	0.0000074675946666666T
L R excite-compute cosi/system: bpf `syscall `count `sys_setresgid (on excite-compute, from Chicago, IL, US)	0.0000018733013333333T
L R excite-compute cosi/system: bpf `syscall `count `sys_close (on excite-compute, from Chicago, IL, US)	0.0532760794213333T
L R excite-compute cosi/system: bpf `syscall `count `sys_arch_prctl (on excite-compute, from Chicago, IL, US)	0.000147994646666666T

394 Metrics

Syscalls: Duration



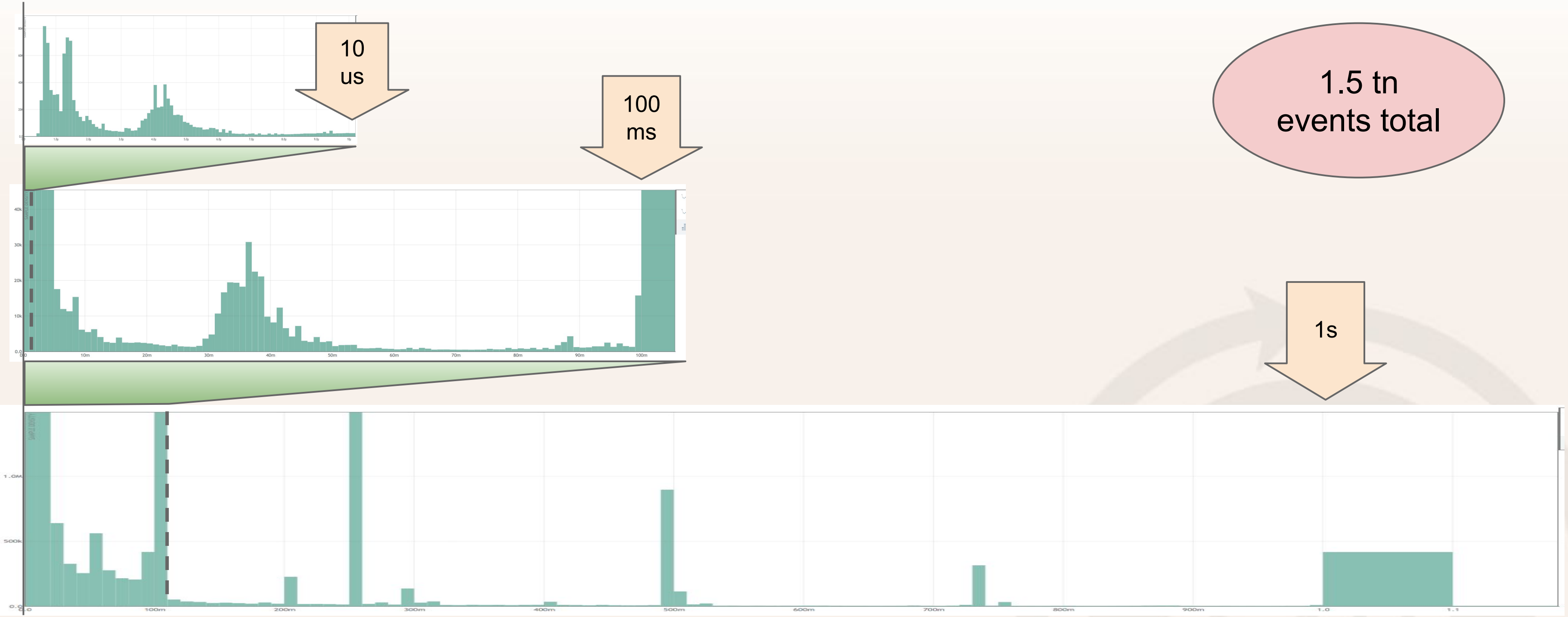
1 us

10 us

excite-compute cosi/system: bpf `syscall `latency `all (on excite-compute, from Chicago, IL, US)

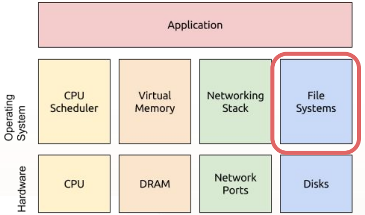
samples	mean	min	median	max
1.24G	6.11m	0.00	2.43µ	51.0k

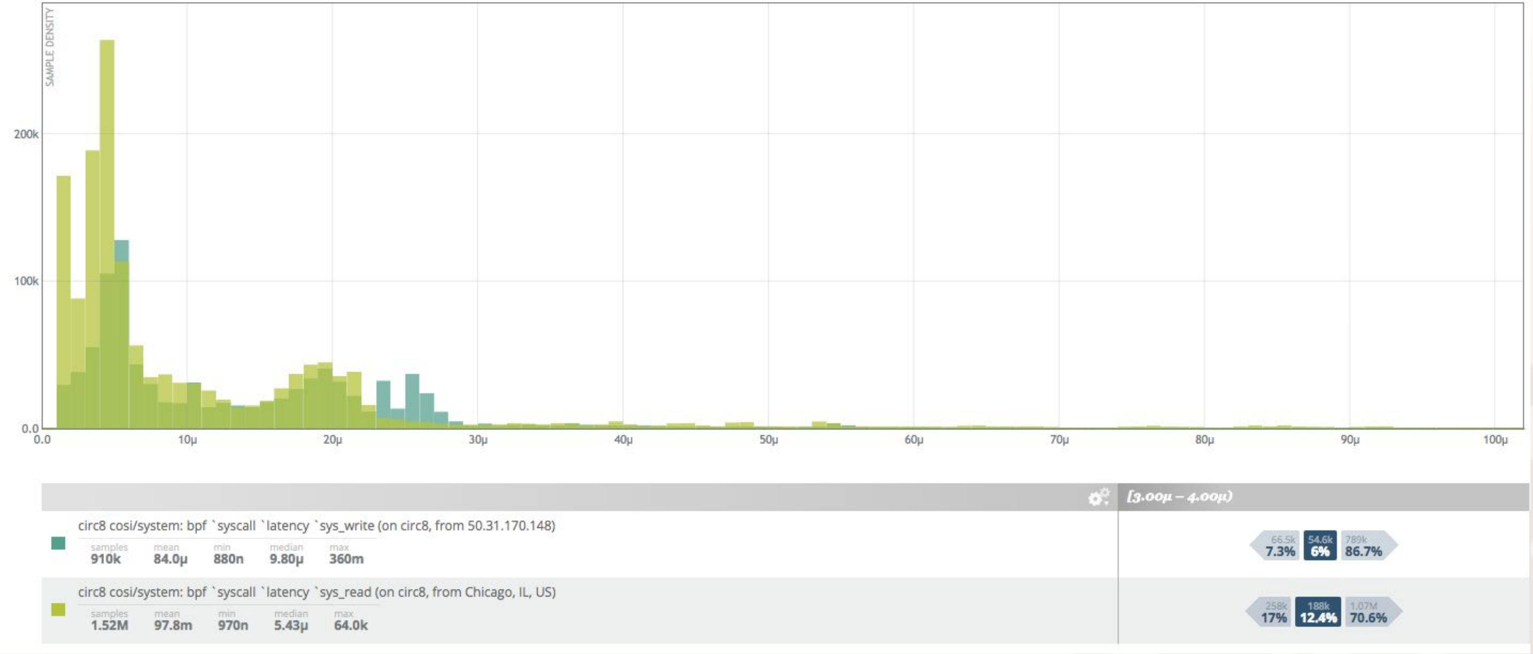
Syscall durations span >8 orders of magnitude



1.5 tn events total

File System: Latency





🟩 circ8 cosi/system: bpf `syscall `latency `sys_write (on circ8, from 50.31.170.148)

samples	mean	min	median	max
910k	84.0µ	880n	9.80µ	360m

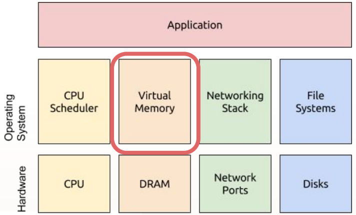
66.5k 7.3% | 54.6k 6% | 789k 86.7%

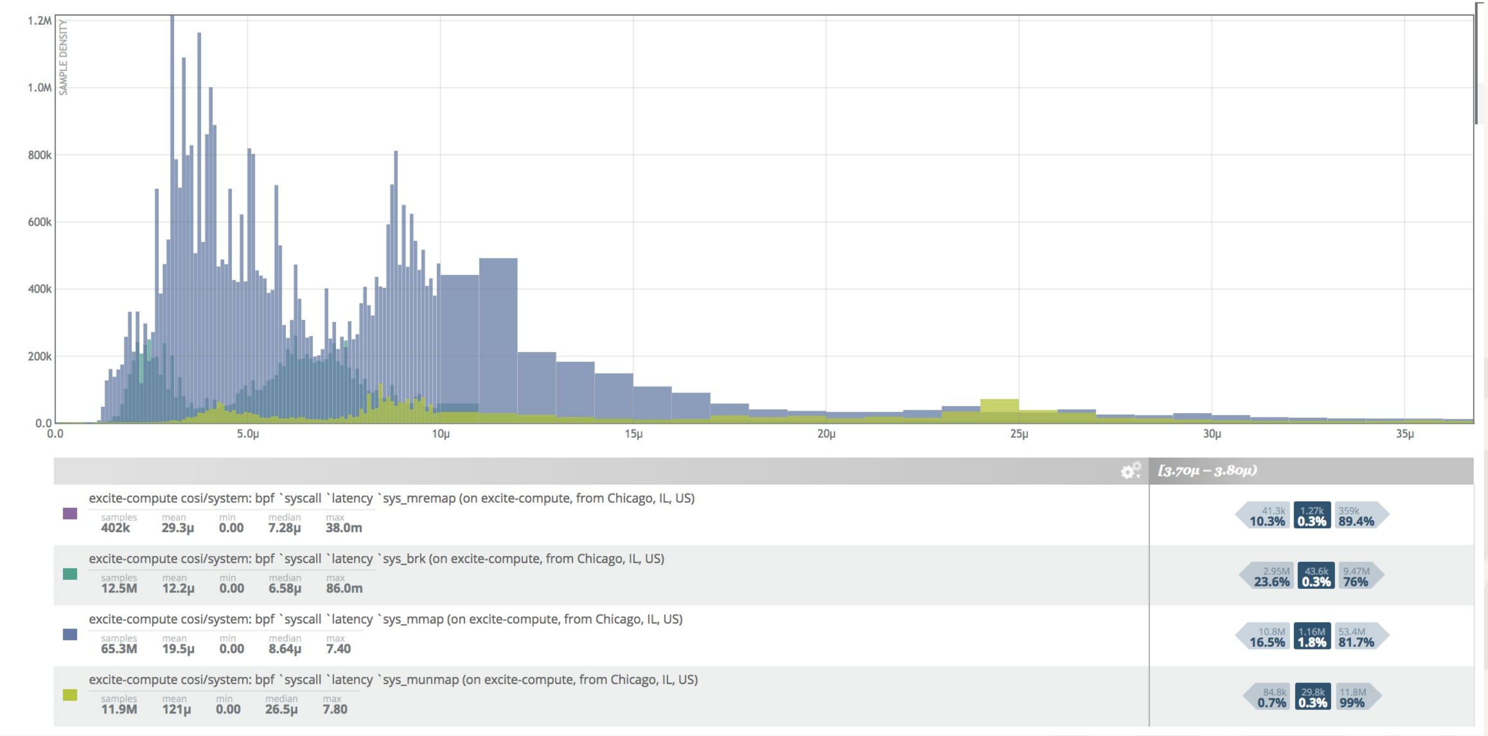
🟨 circ8 cosi/system: bpf `syscall `latency `sys_read (on circ8, from Chicago, IL, US)

samples	mean	min	median	max
1.52M	97.8m	970n	5.43µ	64.0k

258k 17% | 188k 12.4% | 1.07M 70.6%

Memory: Allocation Latency





[3.70µ – 3.80µ)

	samples	mean	min	median	max			
excite-compute cosi/system: bpf `syscall `latency `sys_mremap (on excite-compute, from Chicago, IL, US)	402k	29.3µ	0.00	7.28µ	38.0m	41.3k 10.3%	1.27k 0.3%	359k 89.4%
excite-compute cosi/system: bpf `syscall `latency `sys_brk (on excite-compute, from Chicago, IL, US)	12.5M	12.2µ	0.00	6.58µ	86.0m	2.95M 23.6%	43.6k 0.3%	9.47M 76%
excite-compute cosi/system: bpf `syscall `latency `sys_mmap (on excite-compute, from Chicago, IL, US)	65.3M	19.5µ	0.00	8.64µ	7.40	10.8M 16.5%	1.16M 1.8%	53.4M 81.7%
excite-compute cosi/system: bpf `syscall `latency `sys_munmap (on excite-compute, from Chicago, IL, US)	11.9M	121µ	0.00	26.5µ	7.80	84.8k 0.7%	29.8k 0.3%	11.8M 99%

Further Reading

Slides: @HeinrichHartman / #DevOpsDaysZH

Code: https://github.com/circonus-labs/nad/.../bccbpf

Blog: http://www.circonus.com/2018/05/linux-system-monitoring-with-ebpf/