Using eBPF for Linux Performance Analyses

In 25 Minutes…

Peter Zaitsev, CEO Percona

February 3rd, 2019

FOSDEM, Monitoring and Observability
Brussels, Belgium

PERCONA

Question

Who is familiar with eBPF ?

About Myself

Performance Geek turned CEO, who kept his passion

PERCONA

About the Presentation

eBPF Overview

Practical examples of eBPF based tools

PERCONA

eBPF History and Linux Support

eBPF - Extended Berkeley Packet Filter

Berkeley Packet Filter - Originated in 1992 as efficient virtual machine for Packet Filtering

Extended Berkeley Packet Filter – Extended Version found in Linux

General Event Processing Framework

JIT Compiler for high efficiency

eBPF in Linux

Has been in Linux Kernel since 2014

Actively being improved

Integrated in “perf” tooling system

Improvements in recent Kernels

bpf2bpf function calls	4.16	`cc8b0b92a169`
BPF used for monitoring socket RX/TX data	4.17	`4f738adba30a`
BPF attached to raw tracepoints	4.17	
BPF attached to `bind()` system call	4.17	
BPF Type Format (BTF)	4.18	
AF_XDP	4.18	
bpfilter	4.18	
End.BPF action for seg6local LWT	4.18	
BPF attached to LIRC devices	4.18	

Netdevice references	4.14	`546ac1ffb70d`
Socket references (array)	4.14	`174a79ff9515`
CPU references	4.15	`6710e1126934`
AF_XDP socket (XSK) references	4.18	`fbfc504a24f5`
Socket references (hashmap)	4.18	`81110384441a`
cgroup storage	4.19	`de9cbbaadba5`
reuseport sockarray	4.19	`5dc4c4b7d4e8`
precpu cgroup storage	4.20	`b741f1630346`
queue	4.20	`f1a2e44a3aec`
stack	4.20	`f1a2e44a3aec`

eBPF Programs

Linux Kernel can load programs in custom byte code

Programs verified before load to prevent misuse

LLVM Clang can compile to eBPF byte code

This compilation is kernel-dependent

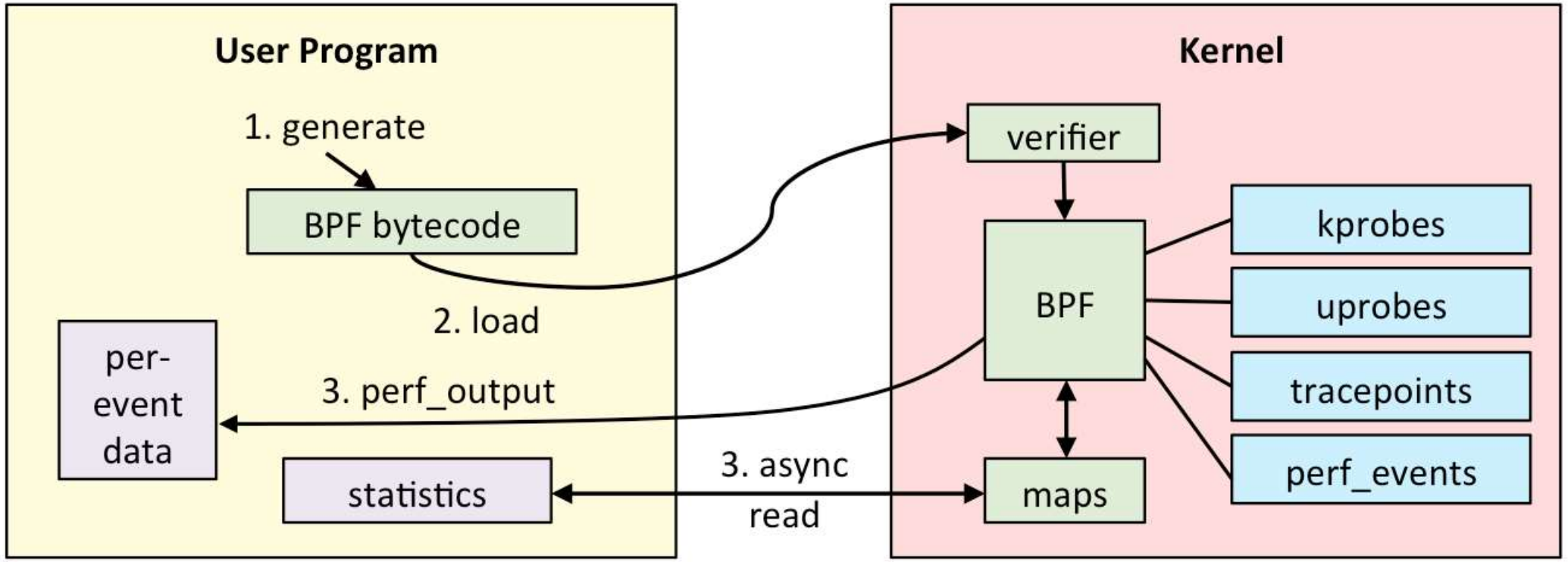
Few will need to write eBPF programs Directly

eBPF Code Example

```
$ sudo tcpdump -p -ni eth0 -d "ip and udp"
(000) ldh      [12]
(001) jeq      #0x800           jt 2    jf 5
(002) ldb      [23]
(003) jeq      #0x11            jt 4    jf 5
(004) ret      #65535
(005) ret      #0
```


PERCONA

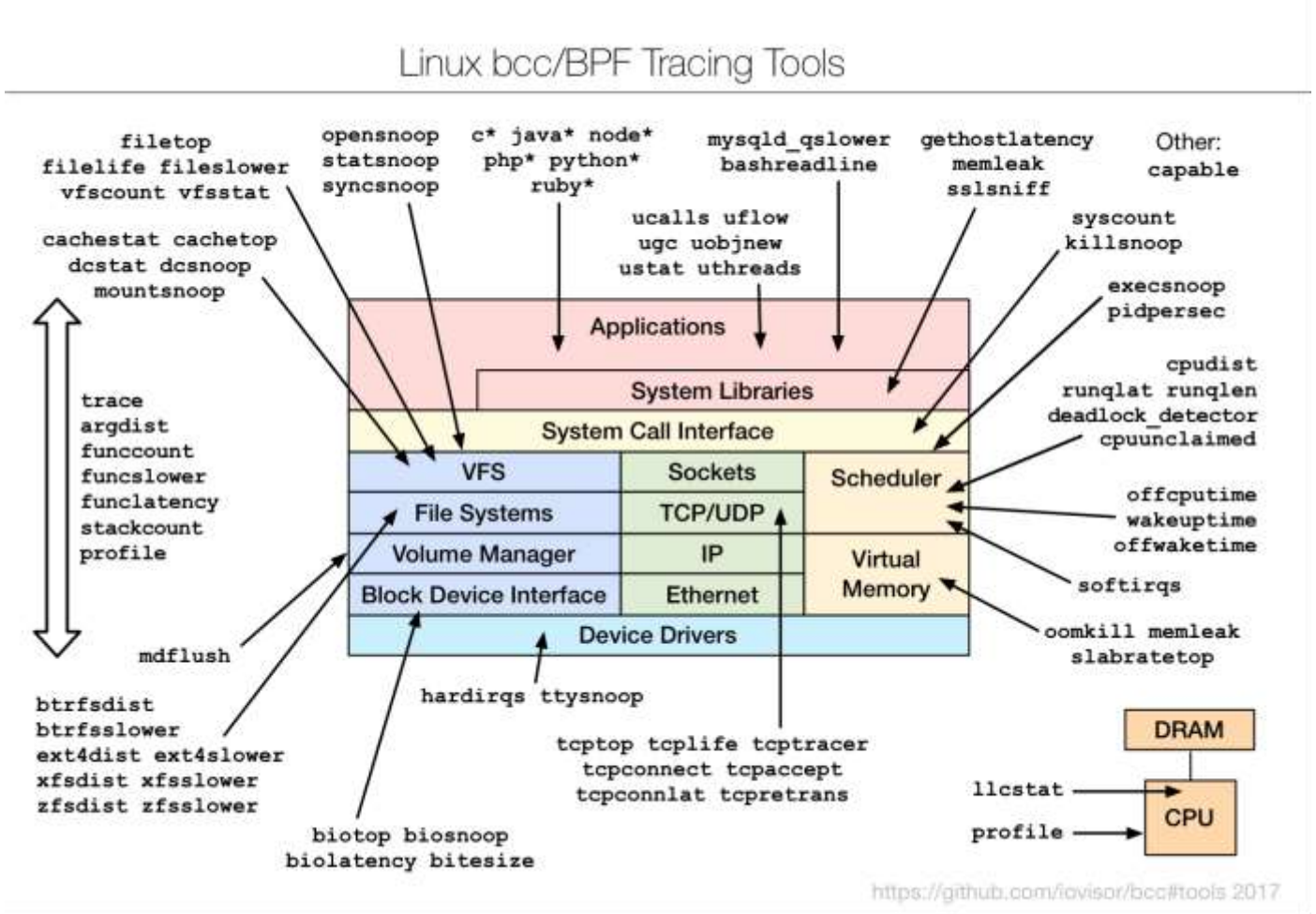
eBPF User Space vs Kernel



Source: http://www.brendangregg.com/ebpf.html

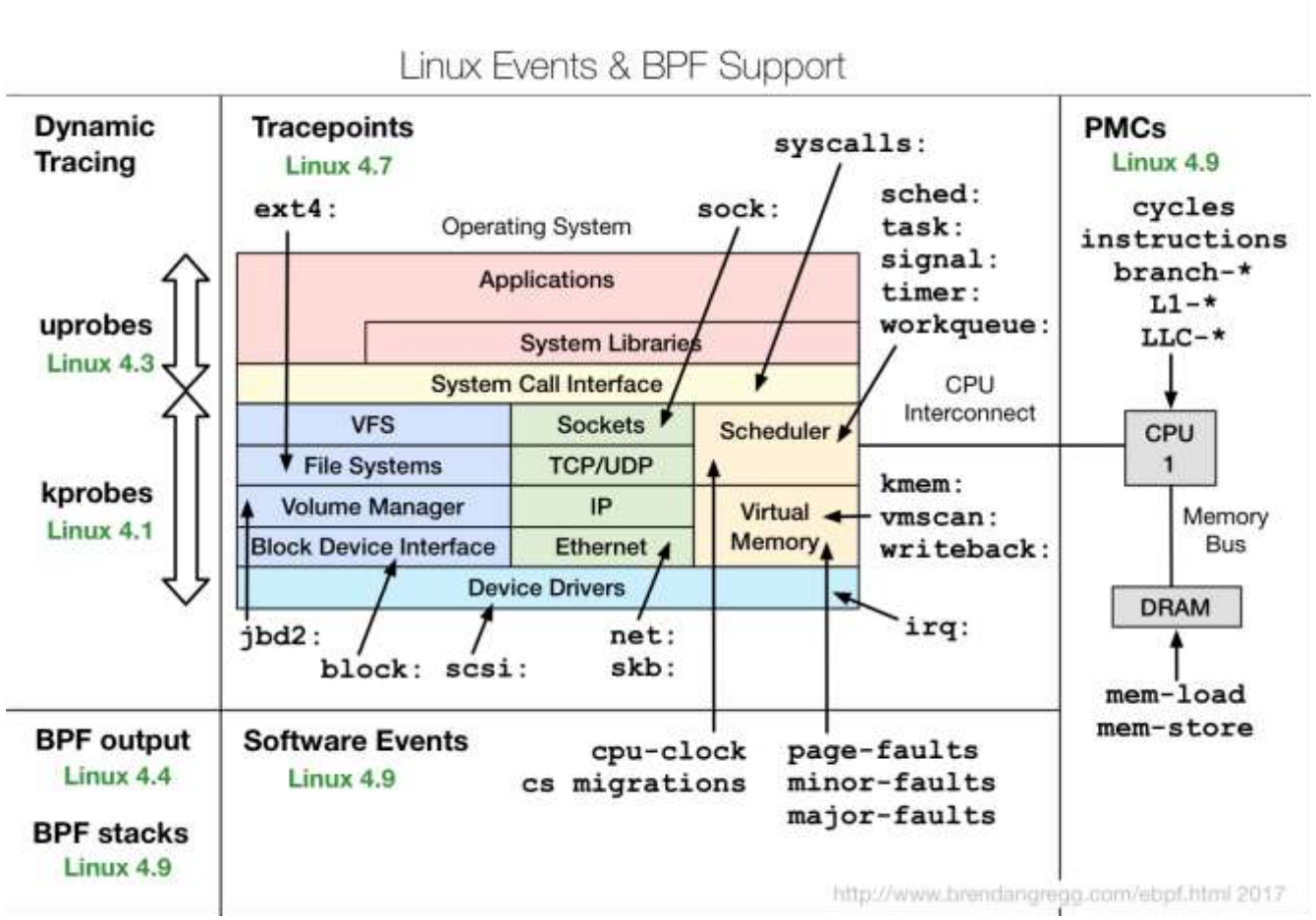
PERCONA

eBPF in Linux Summary



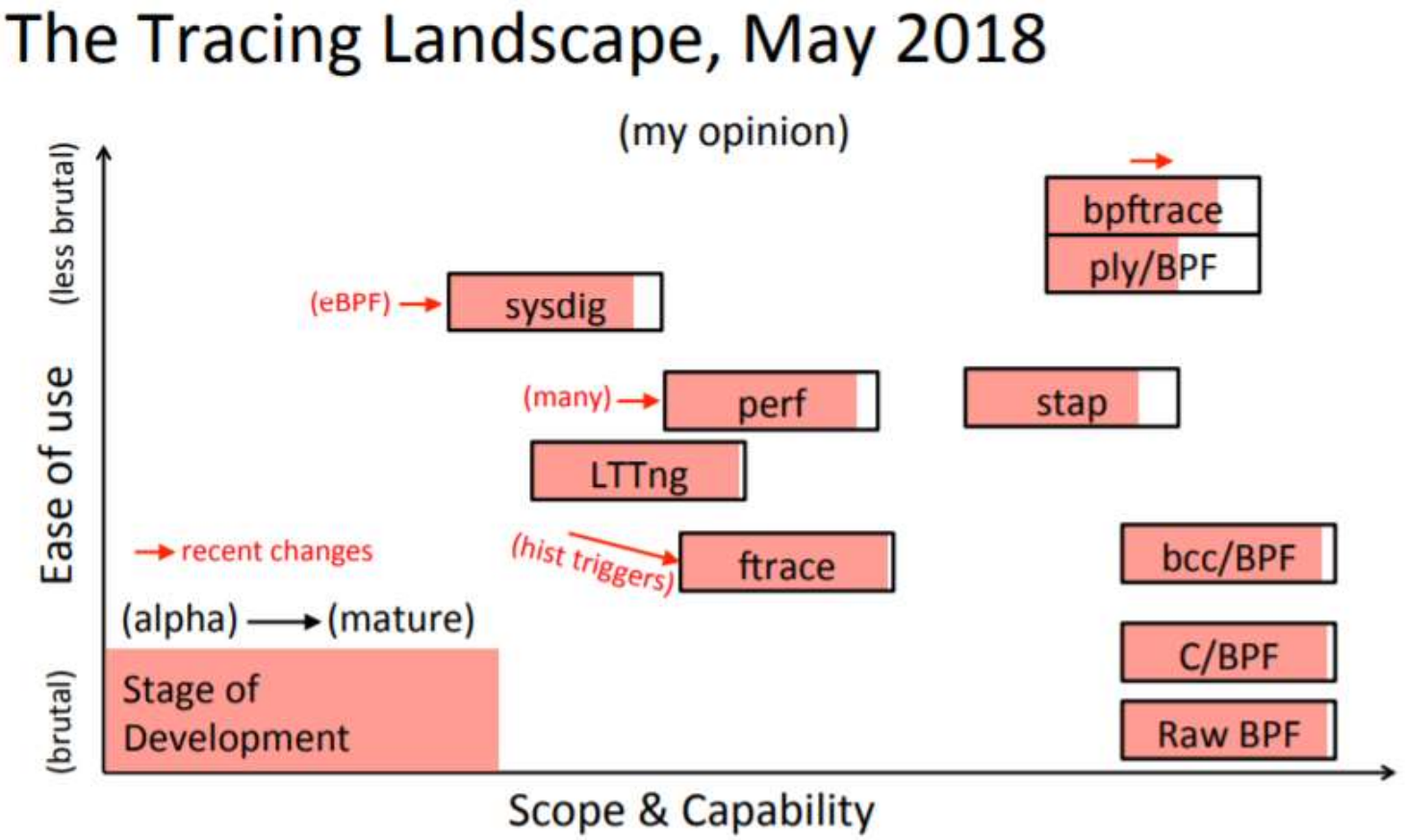
PERCONA

eBPF features in different kernel versions



PERCONA

Tracing Landscape per Brendan Gregg



PERCONA

Things to note

Not all the tools available in the single package

- perf-tools-unstable
- lovisor/bcc
- lovisor/ply
- …

PERCONA

Iovisor bcc installation

Stable and Signed Packages

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 4052245BD4284CDD
echo "deb https://repo.iovisor.org/apt/xenial xenial main" | sudo tee /etc/apt/sources.list.d/iovisor.list
sudo apt-get update
sudo apt-get install bcc-tools libbcc-examples linux-headers-$(uname -r)
```

(replace `xenial` with `artful` or `bionic` as appropriate)

Detailed Instructions: https://github.com/iovisor/bcc/blob/master/INSTALL.md

Note: Tools installed to */usr/share/bcc/tools* not added to PATH automatically

Tools in Action

Profile: Better “poor man’s profiler”

```
root@localhost:/usr/share/bcc/tools# ./profile
Sampling at 49 Hertz of all threads by user + kernel stack... Hit Ctrl-C to end.
^C
    rec_get_nth_field_offs_old(unsigned char const*, unsigned long, unsigned long*)
    [unknown]
    [unknown]
    [unknown]
    ibuf_merge_in_background(bool)
    srv_master_thread()
    std::thread::_State_impl<std::thread::_Invoker<std::tuple<Runnable, void (*)()> > >::_M_run()
    [unknown]
    -                mysqld (704)
        1
```


PERCONA

Biolatency: Block Device Latency

```
root@localhost:/usr/share/bcc/tools# ./biolatency
Tracing block device I/O... Hit Ctrl-C to end.
^C
     usecs               : count     distribution
         0 -> 1          : 0        |                                        |
         2 -> 3          : 0        |                                        |
         4 -> 7          : 0        |                                        |
         8 -> 15         : 0        |                                        |
        16 -> 31         : 0        |                                        |
        32 -> 63         : 0        |                                        |
        64 -> 127        : 1433     |*******                                 |
       128 -> 255        : 7739     |****************************************|
       256 -> 511        : 6213     |********************************        |
       512 -> 1023       : 3602     |******************                      |
      1024 -> 2047       : 4373     |**********************                  |
      2048 -> 4095       : 4662     |************************                |
      4096 -> 8191       : 5487     |****************************            |
      8192 -> 16383      : 5126     |**************************              |
     16384 -> 32767      : 1270     |******                                  |
     32768 -> 65535      : 87       |                                        |
     65536 -> 131071     : 3        |                                        |
    131072 -> 262143     : 0        |                                        |
    262144 -> 524287     : 0        |                                        |
    524288 -> 1048575    : 1        |                                        |
   1048576 -> 2097151    : 0        |                                        |
   2097152 -> 4194303    : 1        |                                        |
```


PERCONA

Biosnoop: Block Device IO Tracing

```
root@localhost:/usr/share/bcc/tools# ./biosnoop
TIME(s)        COMM           PID    DISK    T  SECTOR    BYTES   LAT(ms)
0.000000000    sshd           3542   sdb     R  548736    16384      1.80
0.000023000    mysqld         3874   sda     R  10954816  16384      0.73
0.002172000    sshd           3542   sdb     R  489232    4096       2.06
0.002194000    biosnoop       4782   sda     R  37749504  4096       1.39
0.002199000    biosnoop       4782   sda     R  37749512  4096       1.28
0.002203000    biosnoop       4782   sda     R  37749520  4096       1.19
0.002206000    biosnoop       4782   sda     R  37749528  4096       1.10
0.002209000    biosnoop       4782   sda     R  37749536  4096       1.01
0.002212000    biosnoop       4782   sda     R  37749544  4096       0.92
0.002215000    biosnoop       4782   sda     R  37749552  4096       0.79
0.002218000    biosnoop       4782   sda     R  37749560  4096       0.68
0.002221000    biosnoop       4782   sda     R  37749576  4096       0.59
0.002223000    biosnoop       4782   sda     R  37749584  4096       0.50
0.002226000    biosnoop       4782   sda     R  37749592  4096       0.42
0.004248000    biosnoop       4782   sda     R  37749600  4096       2.36
0.004311000    biosnoop       4782   sda     R  37749608  4096       2.08
0.004317000    sshd           3542   sdb     R  489120    4096       1.94
0.004331000    biosnoop       4782   sda     R  37749616  4096       1.86
0.004335000    biosnoop       4782   sda     R  37749624  4096       1.78
0.004339000    biosnoop       4782   sda     R  37749632  4096       1.71
0.004344000    biosnoop       4782   sda     R  37749640  4096       1.64
0.004349000    biosnoop       4782   sda     R  37749648  4096       1.57
```

Ext4dist: Filesystem Latency per Operation

```
operation = write
     usecs               : count     distribution
         0 -> 1          : 8       |*                                       |
         2 -> 3          : 10      |**                                      |
         4 -> 7          : 6       |*                                       |
         8 -> 15         : 18      |***                                     |
        16 -> 31         : 182     |****************************************|
        32 -> 63         : 52      |***********                             |
        64 -> 127        : 9       |*                                       |
       128 -> 255        : 0       |                                        |
       256 -> 511        : 1       |                                        |
       512 -> 1023       : 4       |                                        |
      1024 -> 2047       : 2       |                                        |
      2048 -> 4095       : 3       |                                        |
      4096 -> 8191       : 1       |                                        |
      8192 -> 16383      : 5       |*                                       |
     16384 -> 32767      : 2       |                                        |
```

```
operation = fsync
     usecs               : count     distribution
         0 -> 1          : 0       |                                        |
         2 -> 3          : 0       |                                        |
         4 -> 7          : 0       |                                        |
         8 -> 15         : 0       |                                        |
        16 -> 31         : 0       |                                        |
        32 -> 63         : 0       |                                        |
        64 -> 127        : 0       |                                        |
       128 -> 255        : 1       |*                                       |
       256 -> 511        : 7       |***********                             |
       512 -> 1023       : 17      |****************************            |
      1024 -> 2047       : 15      |************************                |
      2048 -> 4095       : 13      |********************                    |
      4096 -> 8191       : 19      |*******************************         |
      8192 -> 16383      : 10      |****************                        |
     16384 -> 32767      : 25      |****************************************|
     32768 -> 65535      : 10      |****************                        |
     65536 -> 131071     : 3       |****                                    |
```

```
root@localhost:/usr/share/bcc/tools# ./ext4dist 10 1
Tracing ext4 operation latency... Hit Ctrl-C to end.

16:34:38:

operation = read
     usecs               : count     distribution
         0 -> 1          : 0       |                                        |
         2 -> 3          : 0       |                                        |
         4 -> 7          : 4       |******                                  |
         8 -> 15         : 13      |*********************                   |
        16 -> 31         : 1       |*                                       |
        32 -> 63         : 1       |*                                       |
        64 -> 127        : 1       |*                                       |
       128 -> 255        : 4       |******                                  |
       256 -> 511        : 22      |************************************    |
       512 -> 1023       : 21      |***********************************     |
      1024 -> 2047       : 23      |****************************************|
      2048 -> 4095       : 21      |***********************************     |
      4096 -> 8191       : 9       |***************                         |
      8192 -> 16383      : 11      |******************                      |
     16384 -> 32767      : 5       |********                                |
```


PERCONA

Ext4slower: Trace Slow (or all) IO

```
root@localhost:/usr/share/bcc/tools# ./ext4slower
Tracing ext4 operations slower than 10 ms
TIME     COMM           PID    T BYTES   OFF_KB   LAT(ms) FILENAME
16:42:54 mysqld         704    R 16384   0          10.85 history3.ibd
16:42:54 mysqld         704    R 16384   32         14.40 history3.ibd
16:42:54 mysqld         704    W 512     24084      12.81 ib_logfile0
16:42:54 ext4slower     4786   R 2261    0          10.36 ascii.pyc
16:42:54 mysqld         704    S 0       0          37.52 ib_logfile0
16:42:54 mysqld         704    S 0       0          38.35 ib_logfile0
16:42:54 mysqld         704    S 0       0          50.25 binlog.000022
16:42:54 mysqld         704    S 0       0          56.56 ib_logfile0
16:42:54 mysqld         704    R 16384   2368       13.62 customer3.ibd
16:42:54 mysqld         704    R 16384   112        11.67 customer3.ibd
16:42:54 mysqld         704    S 0       0          33.10 ib_logfile0
16:42:54 mysqld         704    R 16384   224        12.37 customer3.ibd
16:42:54 mysqld         704    R 16384   1200       13.41 customer3.ibd
16:42:54 mysqld         704    R 16384   2992       20.99 undo_001
```

Cachestat: File System Cache Performance

```
root@localhost:/usr/share/bcc/tools# ./cachestat 60
    TOTAL   MISSES      HITS  DIRTIES   BUFFERS_MB  CACHED_MB
     6297     6297         0     5584           19         96
    84281    25255     59026     8511           18         79
    26635    26635         0     8309            8         89
     7615     7615         0     7877           11         95
    22538    13479      9059     8076           15         71
   294862    67600    227262    24382            7        139
    42239    18158     24081     8960           13        142
     5697     5697         0     7242           18        183
```


PERCONA

Runqlat: CPU RunQueue Latency

```
root@localhost:/usr/share/bcc/tools# ./runqlat 10 1
Tracing run queue latency... Hit Ctrl-C to end.

     usecs               : count     distribution
         0 -> 1          : 13       |                                        |
         2 -> 3          : 285      |**                                      |
         4 -> 7          : 2564     |**********                              |
         8 -> 15         : 4827     |********************                    |
        16 -> 31         : 4817     |********************                    |
        32 -> 63         : 2141     |*********                               |
        64 -> 127        : 1086     |****                                    |
       128 -> 255        : 709      |***                                     |
       256 -> 511        : 588      |**                                      |
       512 -> 1023       : 426      |*                                       |
      1024 -> 2047       : 192      |                                        |
      2048 -> 4095       : 95       |                                        |
      4096 -> 8191       : 41       |                                        |
      8192 -> 16383      : 3        |                                        |
```


PERCONA

Execsnoop: Trace Program Starts

```
root@localhost:/usr/share/bcc/tools# ./execsnoop
PCOMM            PID    PPID   RET ARGS
configure        10943  4902     0 ./configure
print            10946  10945    0 /usr/bin/print -r -- \\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
which            10947  10946    0 /usr/bin/which file
file             10948  10946    0 /usr/bin/file -b --mime-type -e tokens -L -z -r
which            10949  10946    0 /usr/bin/which file
file             10950  10946    0 /usr/bin/file -b --mime-type -e tokens -L -z \\\\\\\\\
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
sh               10963  10943    0 /bin/sh
sh               10969  10943    0 /bin/sh
bash             10972  10943    0 /bin/bash
bash             10979  10943    0 /bin/bash
expr             10983  10982    0 /usr/bin/expr 16 + 1
```

Opensnoop: Trace Opened Files

```
root@localhost:/usr/share/bcc/tools# ./opensnoop
PID    COMM               FD ERR PATH
12678  sshd                4   0 /etc/login.defs
12678  sshd                4   0 /etc/passwd
12678  sshd                4   0 /etc/shadow
12678  sshd                4   0 /var/run/utmp
12674  opensnoop          -1   2 /usr/lib/python2.7/encodings/ascii.x86_64-linux-gnu.so
12674  opensnoop          -1   2 /usr/lib/python2.7/encodings/ascii.so
12674  opensnoop          -1   2 /usr/lib/python2.7/encodings/asciimodule.so
12674  opensnoop          11   0 /usr/lib/python2.7/encodings/ascii.py
12674  opensnoop          12   0 /usr/lib/python2.7/encodings/ascii.pyc
1      systemd            18   0 /proc/470/cgroup
1      systemd            18   0 /proc/505/cgroup
505    systemd-resolve    11   0 /run/systemd/netif/links/2
505    systemd-resolve    11   0 /run/systemd/netif/links/2
505    systemd-resolve    11   0 /run/systemd/netif/links/2
505    systemd-resolve    11   0 /run/systemd/netif/links/2
505    systemd-resolve    11   0 /run/systemd/netif/links/2
12673  apt-check           3   0 /etc/apt/sources.list
12673  apt-check           4   0 /tmp/fileutl.message.QxEmbs
```


PERCONA

Tcpconnect: Trace TCP Connections

```
root@localhost:/usr/share/bcc/tools# ./tcpconnect
PID    COMM         IP SADDR            DADDR            DPORT
13525  http         6  2600:3c02::f03c:91ff:fe45:f375 2001:67c:1560:8001::14 80
13526  http         6  2600:3c02::f03c:91ff:fe45:f375 2600:3c02:1::42e4:3f76 80
13524  https        4  66.228.57.247    104.199.116.191  443
13523  http         4  66.228.57.247    74.121.199.234   80
13933  mysql        4  127.0.0.1        127.0.0.1        3306
```


PERCONA

Tcpretrans: TCP Retransmits Details

```
root@localhost:/usr/share/bcc/tools# ./tcpretrans
Tracing retransmits ... Hit Ctrl-C to end
TIME     PID    IP LADDR:LPORT          T> RADDR:RPORT          STATE
19:13:51 1154   4  66.228.57.247:22     R> 62.80.122.52:54871   ESTABLISHED
19:14:42 7      4  66.228.57.247:22     R> 62.80.122.52:54474   ESTABLISHED
19:15:10 1154   4  66.228.57.247:22     R> 62.80.122.52:54474   ESTABLISHED
```


PERCONA

Gethostlatency: DNS Lookup Latency

```
root@localhost:/usr/share/bcc/tools# ./gethostlatency
TIME      PID    COMM                         LATms HOST
19:37:56  14419  http                          3.20 security.ubuntu.com
19:37:56  14418  https                         2.57 repo.iovisor.org
19:37:56  14420  http                         10.87 mirrors.linode.com
19:37:56  14417  http                         31.44 repo.percona.com
19:38:21  14825  ping                         81.89 www.google.com
19:38:28  14826  ping                          0.04 8.8.8.8
19:38:43  14827  mtr                           0.05 8.8.8.8
```


PERCONA

All tools available with BCC:

```
root@localhost:/usr/share/bcc/tools# ls
argdist       capable             doc          hardirqs     mdflush         oomkill      pythonflow   runqslower   tcpconnlat  vfscount
bashreadline  cobjnew             execsnoop    inject       memleak         opensnoop    pythongc     slabratetop  tcpdrop     vfsstat
biolatency    cpudist             ext4dist     javacalls    mountsnoop      perlcalls    pythonstat   softirqs     tcplife     wakeuptime
biosnoop      cpuunclaimed        ext4slower   javaflow     mysqld_qslower  perlflow     reset-trace  solisten     tcpretrans  xfsdist
biotop        criticalstat        filelife     javagc       nfsdist         perlstat     rubycalls    sslsniff     tcpstates   xfsslower
bitesize      dbslower            fileslower   javaobjnew   nfsslower       phpcalls     rubyflow     stackcount   tcpsubnet   zfsdist
bpflist       dbstat              filetop      javastat     nodegc          phpflow      rubygc       statsnoop    tcptop      zfsslower
btrfsdist     dcsnoop             funccount    javathreads  nodestat        phpstat      rubyobjnew   syncsnoop    tcptracer
btrfsslower   dcstat              funclatency  killsnoop    offcputime      pidpersec    rubystat     syscount     tplist
cachestat     deadlock_detector   funcslower   lib          offwaketime     profile      runqlat      tcpaccept    trace
cachetop      deadlock_detector.c gethostlatency llcstat    old             pythoncalls  runqlen      tcpconnect   ttysnoop
```


PERCONA

Ply: Simple Scripting for eBPF

Is not available as packages yet!

```
git clone
https://github.com/iovisor/ply
cd ply
./autogen.sh
./configure
make
```

More Details: https://wkz.github.io/ply/

Ply Example

```
root@localhost:~/ply/ply/src# ./ply -c 'kretprobe:SyS_read{ @.quantize(retval()); }'
1 probe active
^Cde-activating probes

@:

                < 0          4 |▏                                                  |
                  0        151 |▏                                                  |
                  1         20 |▏                                                  |
        [    2,    3]         13 |▏                                                  |
        [    4,    7]         16 |▏                                                  |
        [    8,   15]         28 |▏                                                  |
        [   16,   31]          5 |▏                                                  |
        [   32,   63]          9 |▏                                                  |
        [   64,  127]         22 |▏                                                  |
        [  128,  255]          5 |▏                                                  |
        [  256,  511]        136 |▏                                                  |
        [  512,   1k)         51 |▏                                                  |
        [   1k,   2k)         13 |▏                                                  |
        [   2k,   4k)          3 |▏                                                  |
        [   4k,   8k)          3 |▏                                                  |
        [   8k,  16k)          0 |                                                   |
        [  16k,  32k)          1 |▏                                                  |
```


PERCONA

Ply Language

```
#!/usr/bin/env ply

kprobe:SyS_execve {
        @exec[tid()] = mem(arg(0), "128s");

        i = 0;
        unroll(16) {
                argi = mem(arg(1) + i * sizeof("p"), "p");
                if (!argi)
                        return;

                @argv[tid(), i] = mem(argi, "128s");

                i = i + 1;
        }
}
```

Lets Get Some History !

eBPF and Trending

Command Line Frontends are great for interactive troubleshooting of current problems

Not very helpful if you want to analyze the past

How do we get the data into Monitoring System ?

Meet Cloudflare's eBPF Exporter

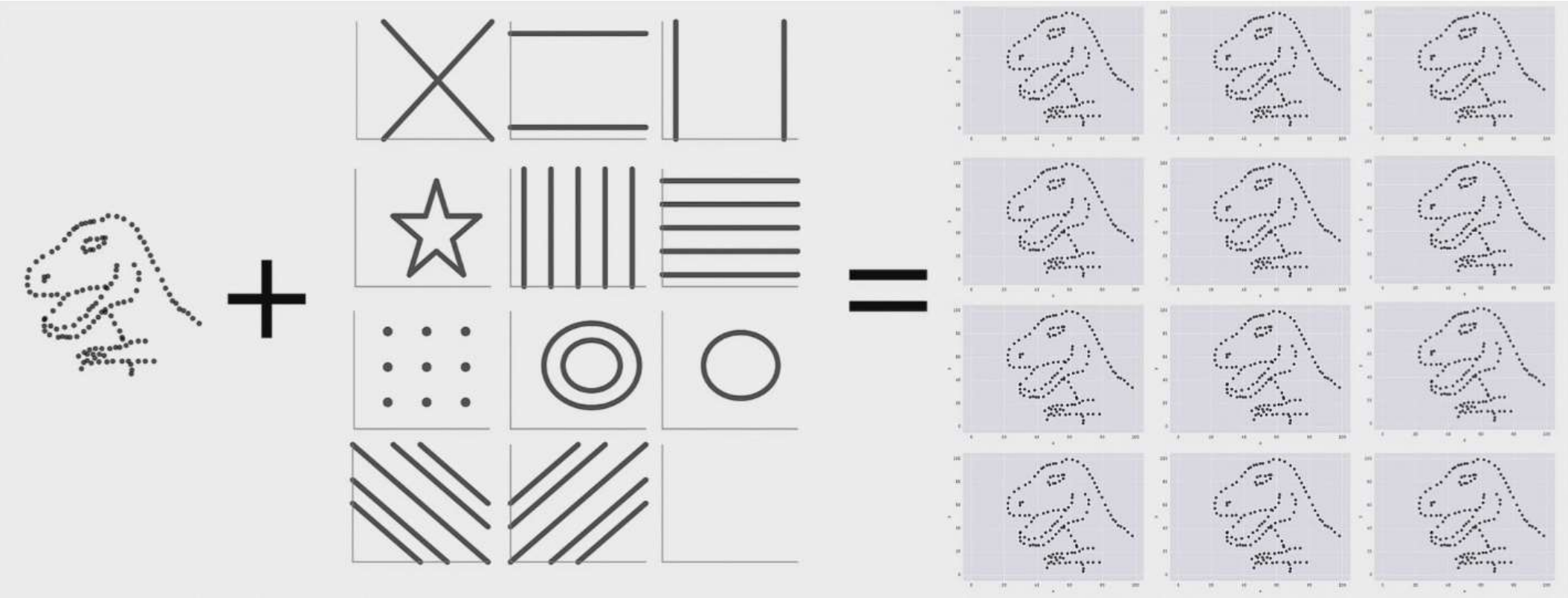
Get results of your eBPF Probes to Prometheus

Can plot Prometheus data using Grafana and other Tools

Use for Alerting with Prometheus Alert Manager

More Info: https://blog.cloudflare.com/introducing-ebpf_exporter/

In Search of the Response Time Histograms



Autodesk Research: https://www.autodeskresearch.com/publications/samestats

PERCONA

eBPF Exporter Architecture

Uses BCC Frontend (Library)

Can be used to take BCC Programs output and expose them in Prometheus Format

Mind Performance Overhead of eBPF Probes

eBPF Overhead

eBPF Programs can be run million+ times per second per core

Case	ns/op	overhead ns/op	ops/s	overhead percent
no probe	316	0	3,164,556	0%
simple	424	108	2,358,490	34%
complex	647	331	1,545,595	105%

More Details: https://github.com/cloudflare/ebpf_exporter/tree/master/benchmark

Installing eBPF Exporter

- **Install Exporter (Binaries Available)**
  - https://github.com/cloudflare/ebpf_exporter/releases
- **Provide Instrumentation Configuration**
  - https://github.com/cloudflare/ebpf_exporter/tree/master/examples

Where should we visualize it ?

- Percona Monitoring and Management (PMM)
- 100% Free and Open Source
- Purpose Built for Open Source Databases
- Based on Grafana and Prometheus
- Easy to Install and Use

PERCONA

Monitoring and Management

Adding eBPF Exporter to PMM

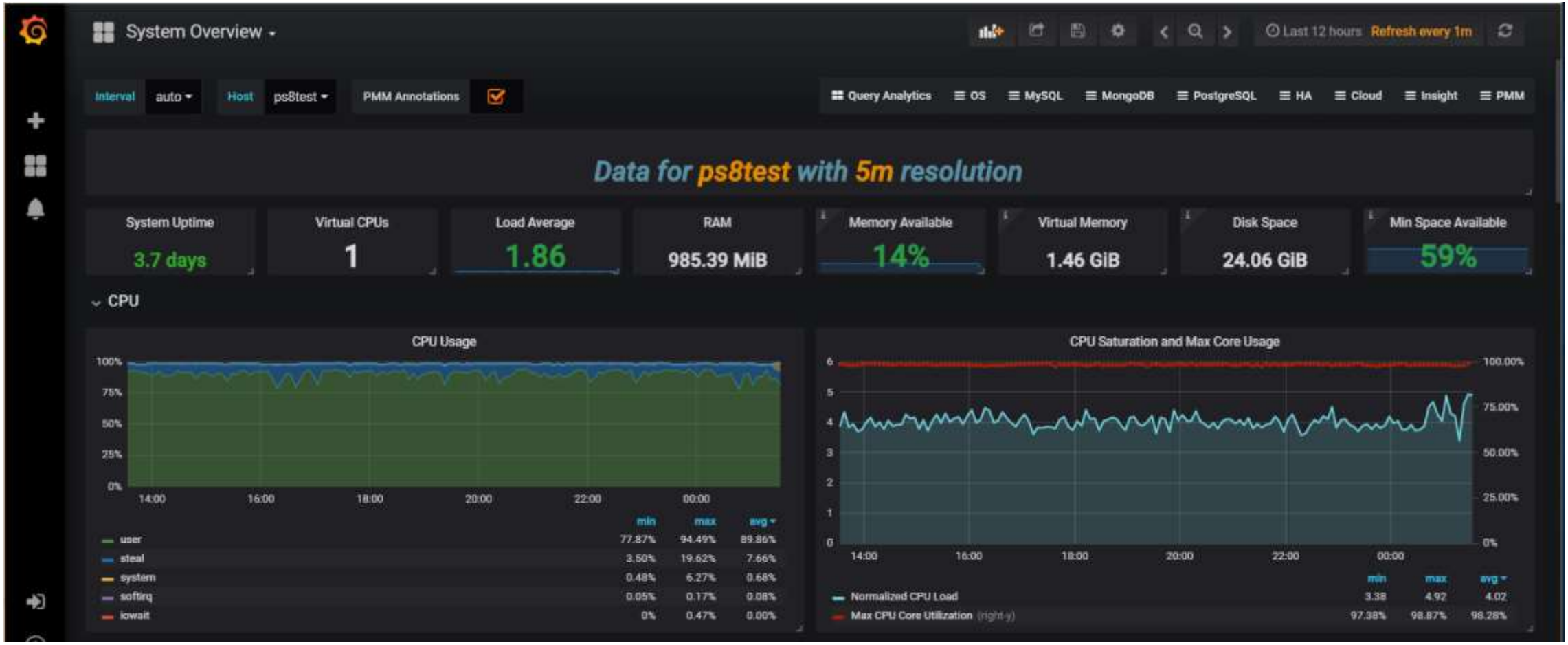
```
pmm-admin add external:service ebpf --service-port=9435
```


PERCONA

Device IO Response Histogram

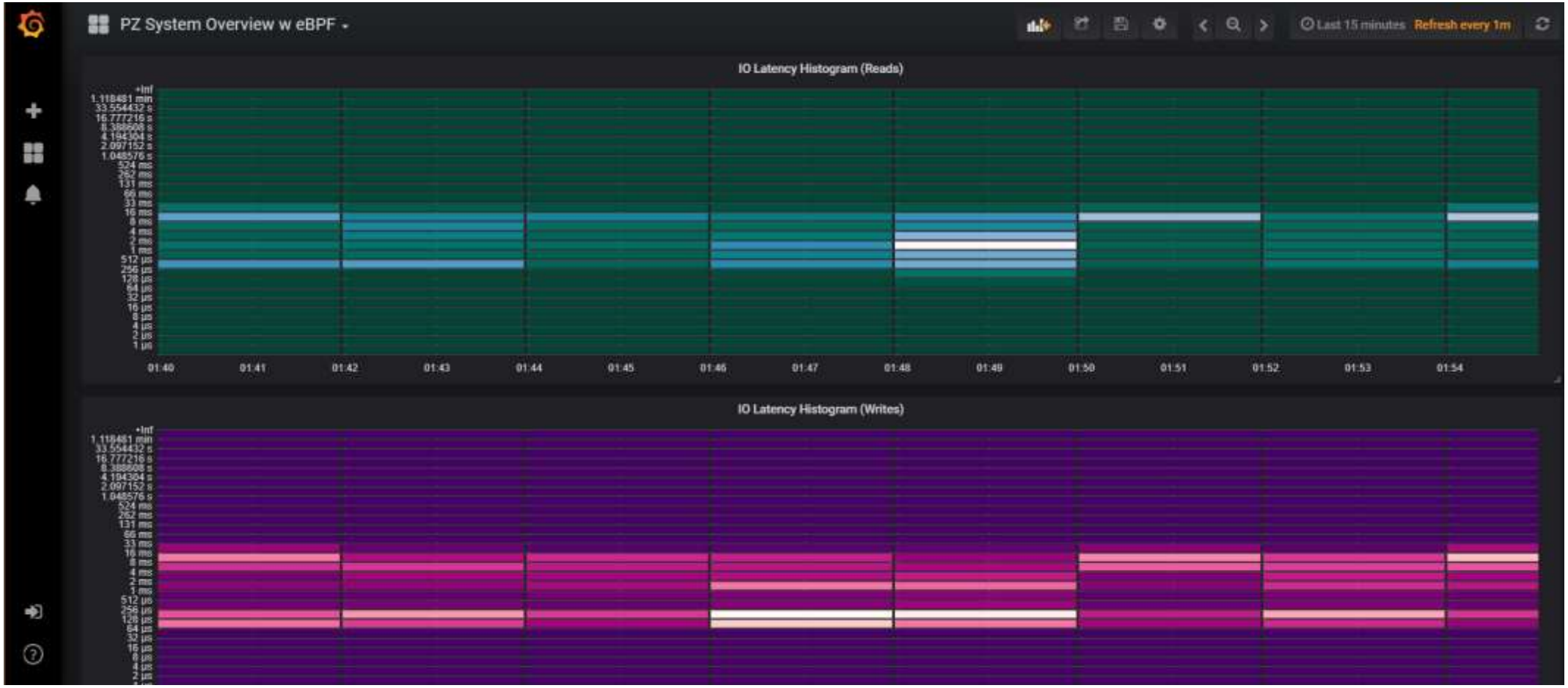
```
# HELP ebpf_exporter_bio_latency_seconds Block IO latency histogram
# TYPE ebpf_exporter_bio_latency_seconds histogram
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="1e-06"} 0
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="2e-06"} 0
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="4e-06"} 0
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="8e-06"} 0
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="1.6e-05"} 0
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="3.2e-05"} 0
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="6.4e-05"} 0
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.000128"} 86
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.000256"} 478
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.000512"} 3333
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.001024"} 4573
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.002048"} 6558
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.004096"} 8061
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.008192"} 9785
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.016384"} 12629
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.032768"} 13146
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.065536"} 13197
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.131072"} 13200
ebpf_exporter_bio_latency_seconds_bucket{device="sda",operation="read",le="0.262144"} 13200
```

PMM System Overview Dashboard



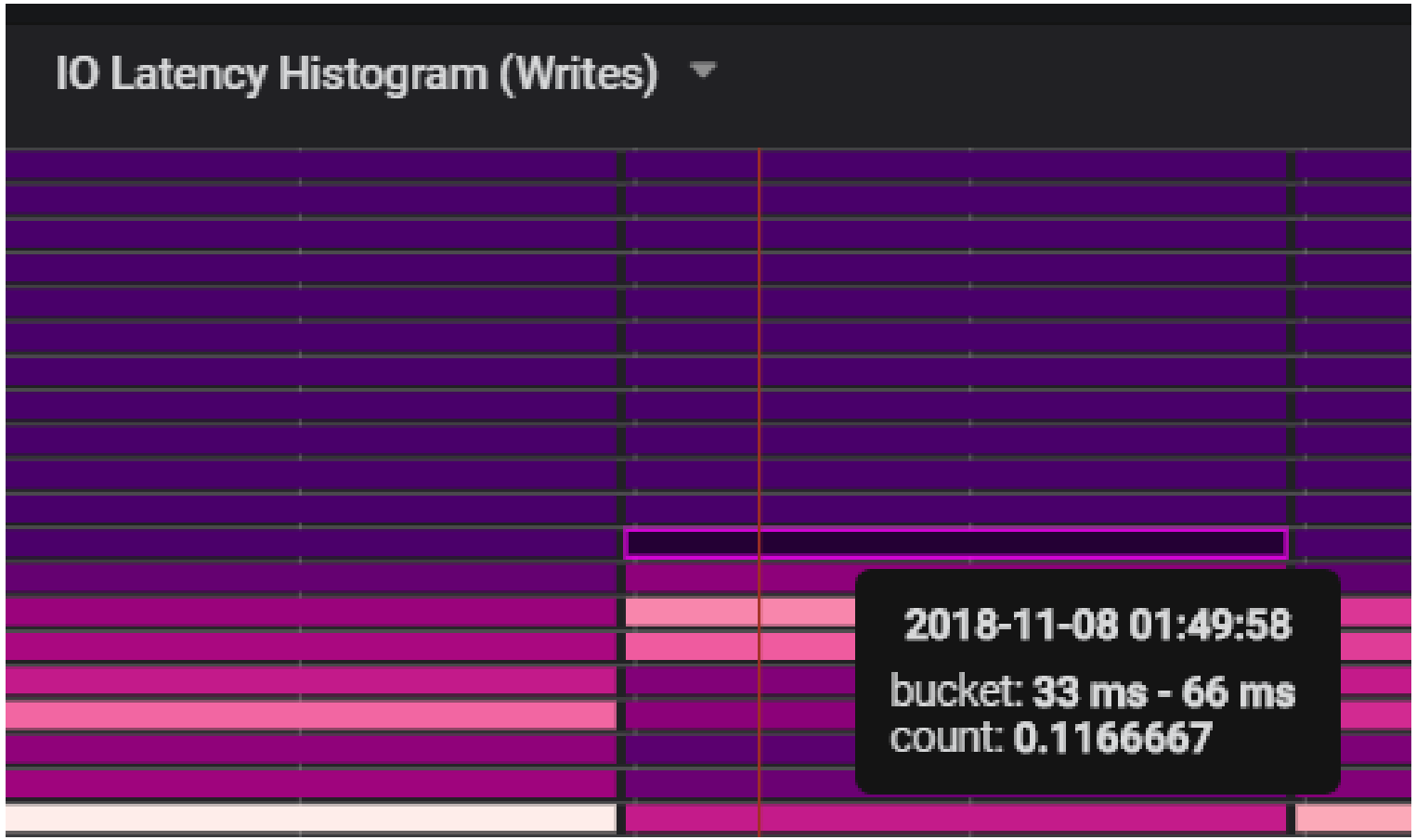
PERCONA

Adding Latency Histograms



PERCONA

Can see outliers



PERCONA

eBPF Future

Bpftrace – Dtrace Replacement

08 Oct 2018

The private bpftrace repository has just been made public, which is big news for DTrace fans. Created by Alastair Robertson, bpftrace is an open source high-level tracing front-end that lets you analyze systems in custom ways. It's shaping up to be a DTrace version 2.0: more capable, and built from the ground up for the modern era of the eBPF virtual machine. eBPF (extended Berkeley Packet Filter) is in the Linux kernel and is the new hotness in systems engineering. It is being developed for BSD, too, where BPF originated.

Screenshot: tracing read latency for PID 181:

```
# bpftrace -e 'kprobe:vfs_read /pid == 30153/ { @start[tid] = nsecs; }
kretprobe:vfs_read /@start[tid]/ { @ns = hist(nsecs - @start[tid]); delete(@start[tid]); }'
Attaching 2 probes...
^C

@ns:
[256, 512)         10900 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@                      |
[512, 1k)          18291 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@|
[1k, 2k)            4998 |@@@@@@@@@@@@@@                                      |
[2k, 4k)              57 |                                                    |
[4k, 8k)             117 |                                                    |
[8k, 16k)             48 |                                                    |
[16k, 32k)           109 |                                                    |
[32k, 64k)             3 |                                                    |
```

See More: http://www.brendangregg.com/blog/2018-10-08/dtrace-for-linux-2018.html

Further Reading List

https://github.com/zoidbergwill/awesome-ebpf

https://slideplayer.com/slide/12710510/

http://www.brendangregg.com/ebpf.html

http://vger.kernel.org/netconf2018_files/BrendanGregg_netconf2018.pdf

http://www.brendangregg.com/Slides/Velocity2017_BPF_superpowers.pdf

https://lwn.net/Articles/740157/

Thank You!