bpftrace recipes

5 real problems solved

@lathiat
Trent Lloyd

Intent

Performance Analysis
Behaviour Analysis

At runtime
In production

Traditional Tools - Performance Analysis

- Fast subsystem specific performance counters
- Heavy use of averages which hide outliers

- Limited per-process or per-device breakdown
  - iostat
  - iotop
  - mpstat
  - bwm-ng
  - netstat
  - vmstat
  - nfsstat
- Instant snapshot misses data
  - top
  - netstat

iostat

```
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          31.78    0.00    9.90    0.93    0.00   57.39

Device   r/s   rkB/s  r_await  w/s       wkB/s      w_await  %util
nvme0n1  0.00  0.00   0.00     27767.60  568768.80  0.03     76.64
sda      0.00  0.00   0.00     0.00      0.00       0.00     0.00
sdb      0.00  0.00   0.00     0.00      0.00       0.00     0.00
sdc      0.00  0.00   0.00     0.00      0.00       0.00     0.00
sdd      0.00  0.00   0.00     0.00      0.00       0.00     0.00
```

top

```
top - 00:12:14 up 1 day, 19:14,  4 users,  load average: 1.26, 0.80, 0.90
Tasks: 869 total,   3 running, 866 sleeping,   0 stopped,   0 zombie

%Cpu(s):  0.2 us,  6.0 sy,  0.0 ni, 92.7 id,  1.0 wa,  0.0 hi,  0.0 si,  0.0 st

MiB Mem : 128773.1 total,    481.3 free,  31294.7 used,  96997.2 buff/cache
MiB Swap:   8192.0 total,   8191.5 free,      0.5 used.  96423.9 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 775088 root      20   0  218464   7748   2056 R  99.7   0.0   0:25.91 fio
```

top

```
top - 00:12:14 up 1 day, 19:14,  4 users,  load average: 1.26, 0.80, 0.90
Tasks: 869 total,   3 running, 866 sleeping,   0 stopped,   0 zombie

%Cpu0  :  0.0 us,  0.7 sy,  0.0 ni, 98.7 id,  0.3 wa,  0.0 hi,  0.3 si,  0.0 st
%Cpu1  :  0.0 us, 47.9 sy,  0.0 ni, 22.9 id, 29.2 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu2  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu3  :  6.3 us, 93.7 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu4  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu5  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu6  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
...
%Cpu31 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 775088 root      20   0  218464   7748   2056 R  99.7   0.0   0:25.91 fio
```

Traditional Tools - Behaviour Analysis

Transfer data of every event to userspace

- strace
- gdb
- blktrace
- iptraf
- Debug logging level

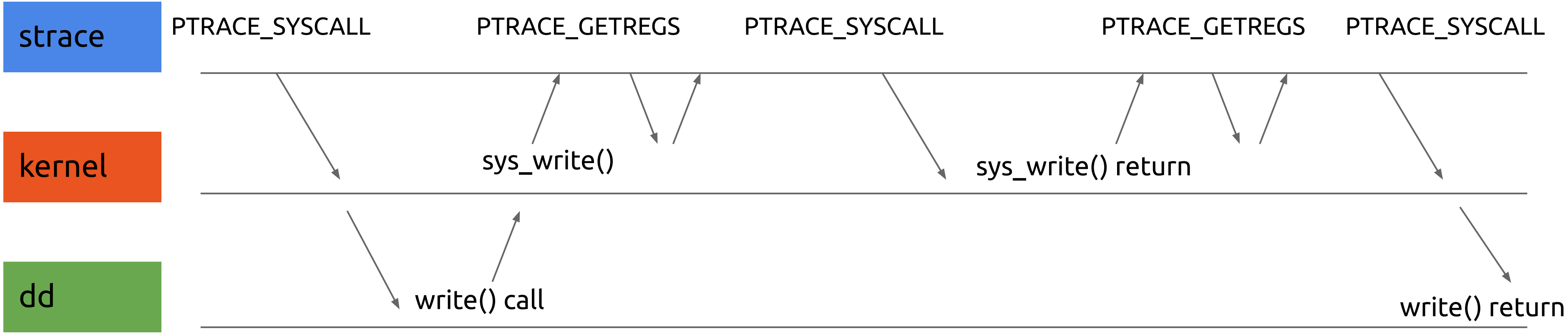
strace performance

```
root@mamar:~# time dd if=/dev/zero of=/dev/null bs=1 count=500k
512000+0 records in
512000+0 records out
512000 bytes (512 kB, 500 KiB) copied, 0.194292 s, 2.6 MB/s
root@mamar:~# strace -eaccept -o x -- dd if=/dev/zero of=/dev/null bs=1
count=500k
512000+0 records in
512000+0 records out
512000 bytes (512 kB, 500 KiB) copied, 18.6821 s, 27.4 kB/s (96x slower)
```

strace output

```
getrandom("\x6e\x03\x2a\xbb\x36\x28\xae\x0f", 8, GRND_NONBLOCK) = 8
brk(0x55d3dc048000)                     = 0x55d3dc048000
openat(AT_FDCWD, "/usr/lib/locale/locale-archive", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/dev/zero", O_RDONLY) = 3
lseek(0, 0, SEEK_CUR)                   = 0
openat(AT_FDCWD, "/dev/null", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 3
dup2(3, 1)                              = 1
close(3)                                = 0
read(0, "\0", 1)                        = 1
write(1, "\0", 1)                       = 1
read(0, "\0", 1)                        = 1
write(1, "\0", 1)                       = 1
read(0, "\0", 1)                        = 1
write(1, "\0", 1)                       = 1
read(0, "\0", 1)                        = 1
write(1, "\0", 1)                       = 1
close(0)                                = 0
close(1)                                = 0
```

strace performance



Custom metrics

Custom grouping

High performance

DNS Lookup Latency

```
# gethostlatency.bt
Attaching 7 probes…
Tracing getaddr/gethost calls... Hit Ctrl-C to end.
TIME      PID    COMM              LATms HOST
05:06:24  441003 python3               0 mamar
05:06:26  441623 http                  3 archive.ubuntu.com
05:06:26  441633 https                82 esm.ubuntu.com
05:06:26  441621 http                238 ddebs.ubuntu.com
```

UDP Connection Life

```
# udplife.bt
Attaching 8 probes...
PID    COMM       LADDR           LPORT RADDR           RPORT   TX_B   RX_B MS
783820 wget       127.0.0.1       0     127.0.0.53      53        78    159 30
1325   systemd-re 10.230.61.29    0     10.230.56.2     53        39    110 34
1325   systemd-re 10.230.61.29    0     10.230.56.2     53        39    226 33
1887   chronyd    10.230.61.29    0     212.71.253.212  123       48    116 15
```

Files Opened

```
# opensnoop.bt
PID    COMM               FD ERR PATH
817503 sudo               -1   2 userdb
817503 sudo               -1   2 /etc/userdb/root.user
817503 sudo               -1   2 /run/userdb/root.user
817503 sudo               -1   2 /run/host/userdb/root.user
817503 sudo               -1   2 /usr/local/lib/userdb/root.user
817503 sudo               -1   2 /usr/lib/userdb/root.user
817503 sudo               -1   2 /lib/userdb/root.user
817503 sudo               13   0 /usr/lib/x86_64-linux-gnu/libnss_systemd.so.2
817503 sudo               13   0 /etc/passwd
817503 sudo               13   0 /etc/shadow
1765   dbus-daemon        -1   2 /run/systemd/users/0
1765   dbus-daemon        25   0 /proc/817503/cmdline
1786   systemd-logind     23   0 /proc/817503/cgroup
1786   systemd-logind     23   0 /proc/1/cgroup
```

I/O Latency Histogram

```
# biolatency.bt
@usecs:
[8, 16)             8609 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@|
[16, 32)            8544 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ |
[32, 64)            3474 |@@@@@@@@@@@@@@@@@@@@                                |
[64, 128)            391 |@@                                                  |
[128, 256)           162 |                                                    |
[256, 512)           148 |                                                    |
[512, 1K)             30 |                                                    |
[1K, 2K)               4 |                                                    |
[2K, 4K)               0 |                                                    |
[256K, 512K)           0 |                                                    |
[512K, 1M)             0 |                                                    |
[1M, 2M)               1 |                                                    |
```

Top System Calls

```
# syscount.bt
Counting syscalls... Hit Ctrl-C to end.

Top 10 syscalls IDs:                   Top 10 processes:
@syscall[13]: 5132                     @process[ps]: 4326
@syscall[16]: 6067                     @process[sshd]: 4784
@syscall[281]: 12470                   @process[dpkg]: 5988
@syscall[9]: 13235                     @process[apt-key]: 6892
@syscall[257]: 13386                   @process[python3]: 7134
@syscall[3]: 14472                     @process[apt]: 12187
@syscall[1]: 19153                     @process[gpgv]: 17214
@syscall[262]: 21109                   @process[jujud]: 19960
@syscall[0]: 24177                     @process[apt-config]: 23521
@syscall[202]: 24431                   @process[landscape-sysin]: 26158
```

Dynamic Tracing

History

ftrace (2008)

Dynamically instrument various points and generate an event

- Static "Tracepoints"
  - Also captured by perf trace
  - ABI Stable
  - Pre-calculates various bits of useful context information
- Dynamic "kprobe"
  - Generate an event on every call to a specific kernel function
  - Need to manually access context information manually from structures etc

```
tracepoint:block:block_bio_{queue,complete}
tracepoint:block:block_rq_{issue,complete}
tracepoint:syscalls:sys_{enter,exit}_*
tracepoint:writeback:writeback_start
tracepoint:filelock:posix_lock_inode
tracepoint:kmem:kmalloc
tracepoint:net:net_dev_{queue,xmit}
tracepoint:scsi:scsi_dispatch_cmd_{done,error,start,timeout}
```

Dynamic Tracing

How?

- Every function entry/exit has a compiled call to the ftrace handler
- At boot time, they are dynamically rewritten with fast NOPs instead
- When a specific tracepoint is enabled, a call to ftrace is written over them

Kernel Ring Buffer

- Kernel handles the trace event
- Writes the trace event information into a ring buffer in memory
- Userspace tool asynchronously consumes the ring buffer
- Advantages
  - Recording happens entirely in kernel space
  - No context switches or process pauses
- Disadvantages
  - Relies on pre-defined information in the kernel static 'tracepoint'
  - Still transferring (less) data to userspace for processing
- Events are lost if the buffer fills up
- Kernel will throttle if events are taking too much time

Performance

```
root@mamar:~# time dd if=/dev/zero of=/dev/null bs=1 count=500k
512000 bytes (512 kB, 500 KiB) copied, 0.194292 s, 2.6 MB/s
root@mamar:~# strace -eaccept -o x -- dd if=/dev/zero of=/dev/null bs=1 count=500k
512000 bytes (512 kB, 500 KiB) copied, 18.6821 s, 27.4 kB/s (96x slower)
```

```
root@mamar:~# perf trace -o /tmp/x1 -- dd if=/dev/zero of=/dev/null bs=1 count=500k
512000+0 records in
512000+0 records out
512000 bytes (512 kB, 500 KiB) copied, 3.18901 s, 161 kB/s
```

```
root@mamar:~# perf trace -e syscalls:sys_exit_write -o /tmp/x1 -- dd if=/dev/zero
of=/dev/null bs=1 count=500k
512000+0 records in
512000+0 records out
512000 bytes (512 kB, 500 KiB) copied, 0.987682 s, 518 kB/s
```

BPF (1992)

Origin: Packet capture (tcpdump) - Efficient packet filtering

User defined programs executed safely in the kernel

```
# tcpdump -ni any port 22
01:23:37.562535 enp68s0f0 Out IP 10.230.61.29.22 > 10.230.65.62.49654: ... length 76
01:23:37.562940 enp68s0f0 In  IP 10.230.65.62.49654 > 10.230.61.29.22: ... length 0

# tcpdump -d port 22
(000) ldh      [12]
(001) jeq      #0x86dd         jt 2     jf 10
(002) ldb      [20]
(003) jeq      #0x84           jt 6     jf 4
(004) jeq      #0x6            jt 6     jf 5
(005) jeq      #0x11           jt 6     jf 23
(006) ldh      [54]
(007) jeq      #0x16           jt 22    jf 8
(008) ldh      [56]
(009) jeq      #0x16           jt 22    jf 23
(010) jeq      #0x800          jt 11    jf 23
```

(e)BPF (2013)

- Expanded word size, storage, registers
- JIT-compiled
- Event-driven from many sources (not just packets)
- Verifier
  - Won't crash
  - Won't take an unbounded amount of time
  - Won't access unsafe memory
- Limited in-kernel helpers to perform various safe tasks

Tracing + BPF

- Attach a BPF program to any tracing event
- Process, summarise or extract user-specific data in-kernel (no context-switch)
- Event outputs are stored in a ring buffer (same as perf)
- We can also store additional data into BPF maps in-memory
- Both the kernel BPF and userspace program can read these
- Only the very small amount of summarised data is sent to userspace

Performance

```
# time dd if=/dev/zero of=/dev/null bs=1 count=500k
512000 bytes (512 kB, 500 KiB) copied, 0.194292 s, 2.6 MB/s
# strace -eaccept -o x -- dd if=/dev/zero of=/dev/null bs=1 count=500k
512000 bytes (512 kB, 500 KiB) copied, 18.6821 s, 27.4 kB/s (96x slower)
# perf trace -e syscalls:sys_exit_write -o /tmp/x1 -- dd if=/dev/zero of=/dev/null bs=1
count=500k
512000 bytes (512 kB, 500 KiB) copied, 0.987682 s, 518 kB/s

# ./writesnoop.bt  -c '/usr/bin/dd if=/dev/zero of=/dev/null bs=1 count=500k'
512000 bytes (512 kB, 500 KiB) copied, 0.333215 s, 1.5 MB/s
@count[dd]: 512003
```

fio performance from biolatency.bt

```
root@mamar:~# fio --ioengine=libaio --filename=test1 -direct --iodepth=16
--name=sequential-write-all --rw=randwrite --bs=32k --size=16G

  write: IOPS=34.1k, BW=1067MiB/s (1119MB/s)(16.0GiB/15358msec); 0 zone resets
  cpu          : usr=5.67%, sys=35.94%, ctx=531823, majf=0, minf=96
  nvme0n1: ios=0/518955, merge=0/4700, ticks=0/226711, in_queue=226711, util=99.42%

  write: IOPS=37.1k, BW=1160MiB/s (1217MB/s)(16.0GiB/14122msec); 0 zone resets
  cpu          : usr=5.30%, sys=34.19%, ctx=530613, majf=0, minf=189
  nvme0n1: ios=0/516958, merge=0/3972, ticks=0/210347, in_queue=210347, util=99.37%
```

seccomp-bpf

```
root@mamar:~# strace -eaccept -o /tmp/x2 -- dd if=/dev/zero of=/dev/null bs=1 count=500k
512000+0 records in
512000+0 records out
512000 bytes (512 kB, 500 KiB) copied, 18.6821 s, 27.4 kB/s
root@mamar:~# strace -o /tmp/x2 -f --seccomp-bpf -ewrite dd if=/dev/zero of=/dev/null
bs=1 count=500k
512000+0 records in
512000+0 records out
512000 bytes (512 kB, 500 KiB) copied, 10.8742 s, 47.1 kB/s
root@mamar:~# perf trace -e syscalls:sys_exit_write -o /tmp/x1 -- time dd if=/dev/zero
of=/dev/null bs=1 count=500k
512000+0 records in
512000+0 records out
512000 bytes (512 kB, 500 KiB) copied, 0.987682 s, 518 kB/s
```

bpftrace language

```
BEGIN {
     print("Starting trace program…")
}

probe_type:probe_name_1
/ comm == "python3" / {
     @start[tid] = nsecs;
}

probe_type:probe_name_3
/ args->ret > 0 && comm == "python3" && @start[tid] / {
     $latms = (@start[tid] - nsecs) / 1000;
     @time[pid] = sum($latms);
     delete(@start[tid]);
}

interval:s:30 {
     print(@time);
}

END {
     print(@time);
}
```

Predicate (condition) for process name

@Global Map, indexed by tid (Thread ID)

$Local variable (latency in ms)

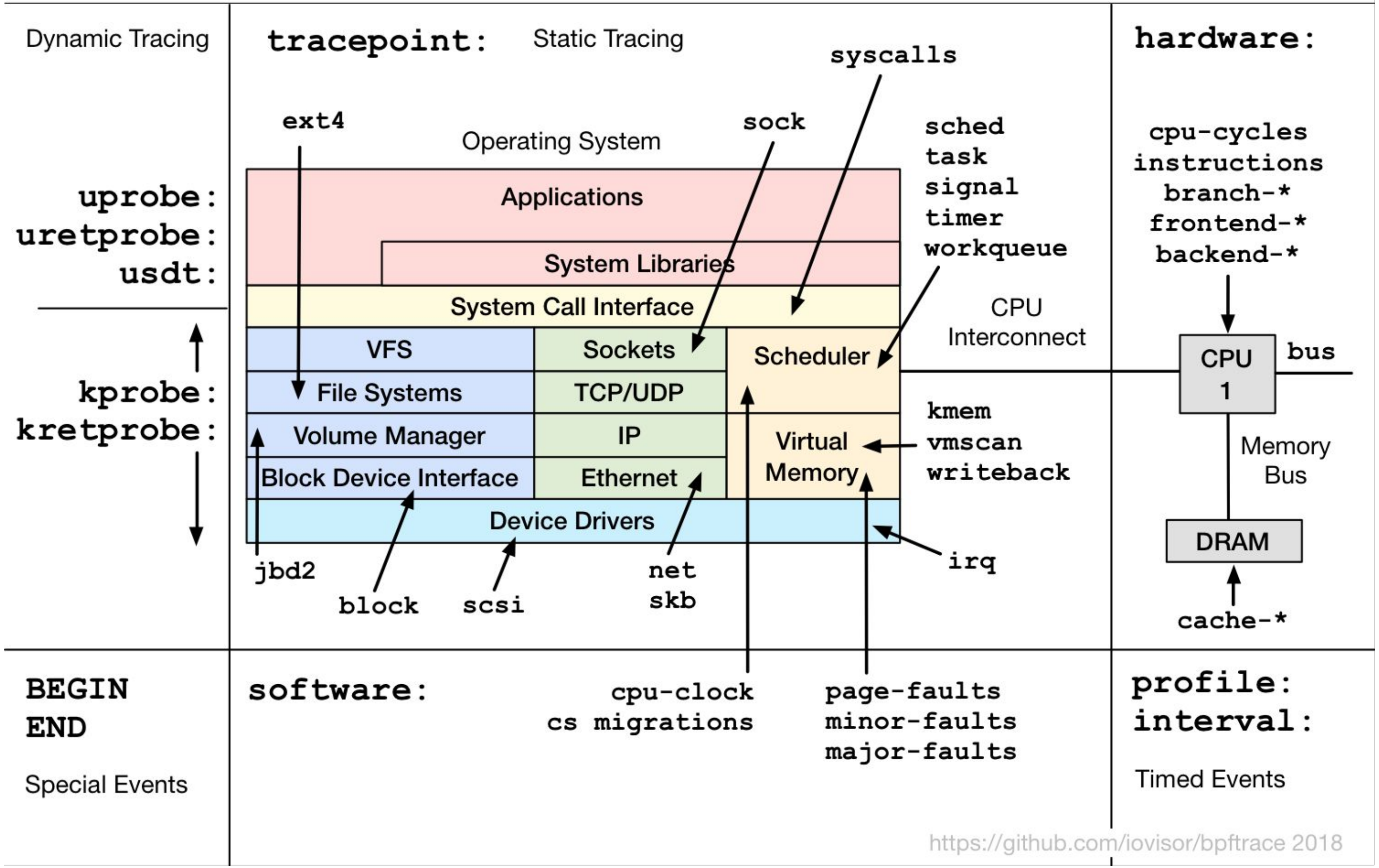
@Global Map, indeed by pid (Process ID

Print entire map every 30 seconds

Print entire map on exit

```
# Files opened by process
bpftrace -e 'tracepoint:syscalls:sys_enter_open { printf("%s %s\n", comm, str(args->filename)); }'

# Syscall count by program
bpftrace -e 'tracepoint:raw_syscalls:sys_enter { @[comm] = count(); }'

# Read bytes by process:
bpftrace -e 'tracepoint:syscalls:sys_exit_read /args->ret/ { @[comm] = sum(args->ret); }'

# Read size distribution by process:
bpftrace -e 'tracepoint:syscalls:sys_exit_read { @[comm] = hist(args->ret); }'

# Show per-second syscall rates:
bpftrace -e 'tracepoint:raw_syscalls:sys_enter { @ = count(); } interval:s:1 { print(@); clear(@); }'

# Trace disk size by process
bpftrace -e 'tracepoint:block:block_rq_issue { printf("%d %s %d\n", pid, comm, args->bytes); }'

# Count LLC cache misses by process name and PID (uses PMCs):
bpftrace -e 'hardware:cache-misses:1000000 { @[comm, pid] = count(); }'

# Profile user-level stacks at 99 Hertz, for PID 189:
bpftrace -e 'profile:hz:99 /pid == 189/ { @[ustack] = count(); };
```

bpftrace Probe Types



https://github.com/iovisor/bpftrace 2018

5 recipes

#1 Start Python Profiler on all new processes

```
bpftrace -e '
tracepoint:syscalls:sys_exit_execve
/ args->ret == 0 && comm == "python3" /
{
printf("%d\n", pid);
}'|xargs -n1 -P32 -I{} austin -C -p {} --output=austin-{}.txt
```

#2 Latency of request by thread name

```
# biosnoop.bt

TIME(ms)     COMM             PID     DISK     OFFSET        LEN    LAT(ms)
2051         journal-offline  346     vda      239030272     4096       131
2214         jbd2/vda1-8      273     vda      1201582080    4096       157
3851         bstore_kv_sync   1146    vdb      0             0           29
4322         bstore_kv_sync   1146    vdb      0             0           50
5676         bstore_kv_sync   1146    vdb      0             0           12
5925         bstore_kv_sync   1146    vdb      0             0          148
6323         bstore_mempool   1146    vdb      835715072     32768        0
6323         bstore_mempool   1146    vdb      47939584      32768        0
8142         jbd2/vda1-8      273     vda      1200656384    24576        0
8144         jbd2/vda1-8      273     vda      1200680960    4096         2
10692        jujud            670     vda      2844270592    32768       10
```

#3 I/O latency correlated with program

```
tracepoint:syscalls:sys_enter_pwritev {
  @start[tid] = nsecs;
}
tracepoint:syscalls:sys_exit_pwritev / @start[tid] / {
    @times[comm] = hist(nsecs - @start[tid]);
    delete(@start[tid]);
}
interval:s:30 { print(@times) }

@times[tp_fstore_op] (nsecs):
[256K, 512K) 353
[512K, 1M)   112
[1M,   2M)   18
```

#4 Murder mystery…

```
root@mamar:~# bpftrace -e '
tracepoint:signal:signal_generate
  /args->sig == 15 / {
    printf("%s (%d) sent signal %d to PID %d\n",
           comm, pid, args->sig, args->pid);
}'

Attaching 1 probe…
killall (157967) sent signal 15 to PID 157966
```

#5 SSL Snoop

```
# BPFTRACE_STRLEN=200 ./sslintercept.bt
=====
> pid=594548 comm=openssl retval=15
GET / HTTP/1.0
=====
> pid=594548 comm=openssl retval=1

=====
< pid=594548 comm=openssl retval=103
HTTP/1.1 301 Moved Permanently
Content-Length: 0
Location: https://github.com/
connection: close
```

```
#!/usr/bin/env bpftrace
uprobe:libssl:SSL_read, uprobe:libssl:SSL_write
{ @buf[tid] = arg1; }

uretprobe:libssl:SSL_read {
  if (retval > 0) {
    printf("=====\n< pid=%-6d comm=%s retval=%d\n%s\n",
           pid, comm, retval, str(@buf[tid], retval));
  }
  delete(@buf[tid]);
}
uretprobe:libssl:SSL_write {
  if (retval > 0) {
    printf("=====\n< pid=%-6d comm=%s retval=%d\n%s\n",
           pid, comm, retval, str(@buf[tid], retval));
  }
  delete(@buf[tid]);
}
```

https://github.com/gojue/ecapture

#6 wildcard kprobe with userspace stack

```
# bpftrace -e 'tracepoint:sched:sched_switch { @[kstack] = count(); }'
@[
__schedule+697
__schedule+697
schedule+50
schedule_timeout+365
xfsaild+274
kthread+248
ret_from_fork+53
]: 73
@[
__schedule+697
__schedule+697
schedule_idle+40
do_idle+356
cpu_startup_entry+111
start_secondary+423
secondary_startup_64+165
]: 305
```

I/O latency correlated with stack

```
TODO: need to try probe io_schedule for the latency here

tracepoint:syscalls:sys_enter_pwritev
  @start[tid] = nsecs;
}
tracepoint:syscalls:sys_exit_pwritev
/ @start[tid] /
{
    $latms = (nsecs - @start[tid]) / 1000;
    if ($latms > 100) { @times[ustack] = hist($latms) }
    delete(@start[tid]);
}
interval:s:30 { print(@times) }
```

```
@times[tp_fstore_op] (nsecs):
```

Many different methods…

System administrator friendly…
(sometimes)

Limitations

Other tools

perf

Anything not needing in kernel summarising - if you want to dump all events

systemtap

dtrace

Flamegraph

Find the CPU reason for the bottleneck

BPF Ecosystem

- New kind of fast, safe, in-kernel, event-driven software
- Use cases accelerating rapidly
  - In-kernel Load Balancers
  - Custom CPU Schedulers
  - Network/Firewall Processing
  - Security and Auditing
  - Continuous Profiling

Resources

Books

- BPF Performance Tools (Brendan Gregg)
- Systems Performance 2nd edition (Brendan Gregg)

https://github.com/iovisor/bpftrace

YouTube "bpftrace"

Questions

lathiat.net/talks
twitter.com/lathiat
@lathiat@fosstodon.org
linkedin.com/in/lathiat

trent.lloyd@canonical.com

https://fosstodon.org/@lathiat