THE MAGICAL THINGS YOU CAN DO WITH LINUX'S KERNEL TRACING

eBPF

WILLIAM STEWART

@zoidbergwill everywhere

SRE/DevOps Team Lead

superbalist.com

Find me on zatech's Slack: zatech.co.za

DISCLAIMER !

	OSI	TCP/IP
7	Application	Applications (FTP, SMTP, HTTP, etc.)
6	Presentation	
5	Session	
4	Transport	TCP (host-to-host)
3	Network	IP
2	Data link	Network access (usually Ethernet)
1	Physical	

ORIGINALLY JUST FOR FILTERING PACKETS

BERKLEY PACKET FILTERS

by Julia Evans

WHAT ARE THOSE COMPUTERS SAYING TO EACH OTHER?

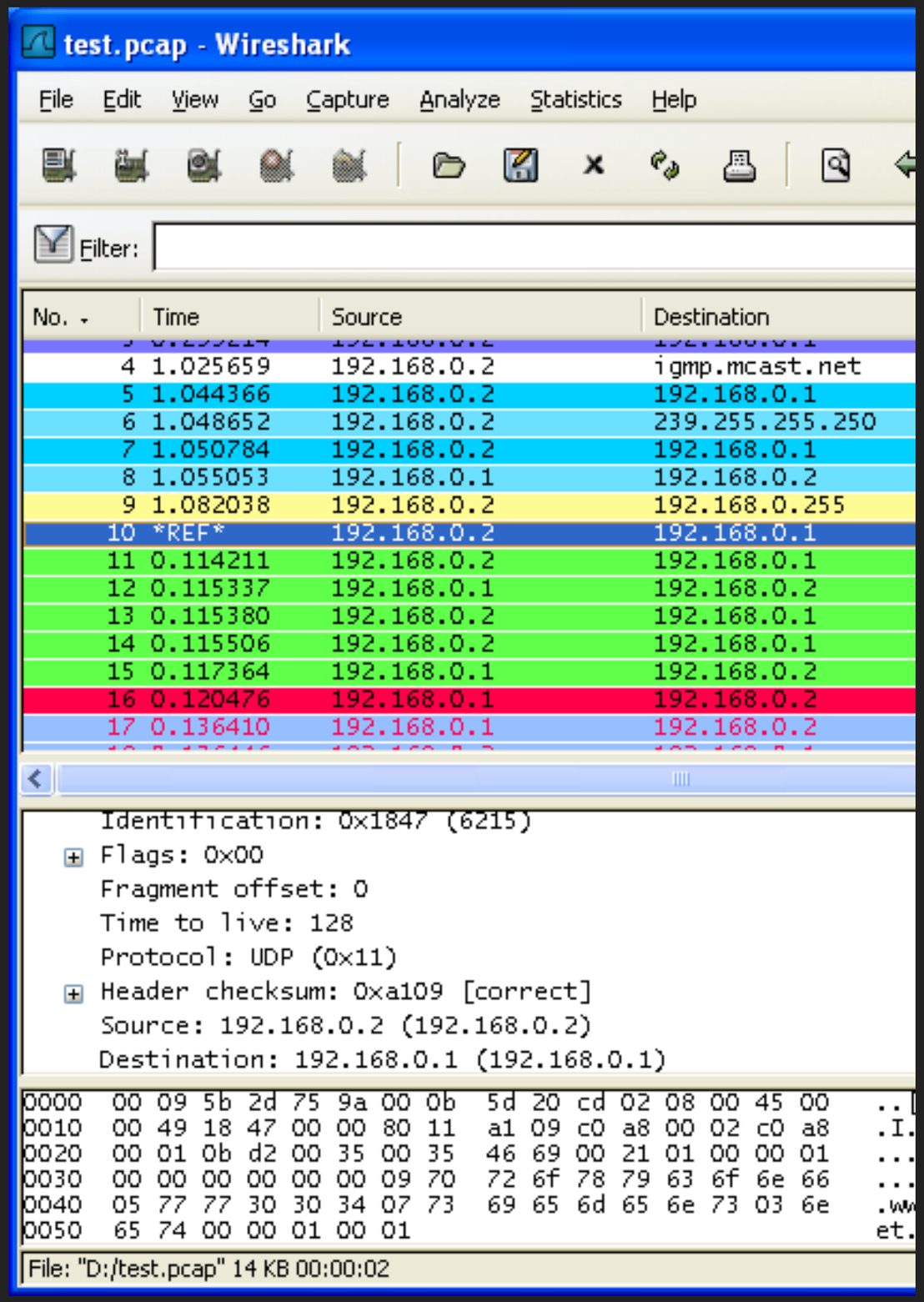
WITH TCPDUMP, YOU CAN FIND OUT!

Source: https://jvns.ca/zines/

TCPDUMP

```
tcpdump -i any port 80 or 443
```

- One of the first tools that used BPF
- Limited
- Layer 3 and 4 filtering at kernel level, but working with packets has to be in userspace



TRACING, NETWORKING, AND SECURITY

WHAT NOW?

extended Berkley Packet Filters and the BPF Compiler Collection

- Linux 4.1+, building on features from Linux 3.15
- BCC Kernel Version Features
- iovisor project
- Bindings for lua, Python, and Go.

extended Berkley Packet Filters and the BPF Compiler Collection

- ftrace
- perf
- kprobes and kretprobes
- tracepoints
- uprobes and uretprobes

Linux Tracing is Magic!

ftrace

perf_events

eBPF

SystemTap

LTTng

ktap

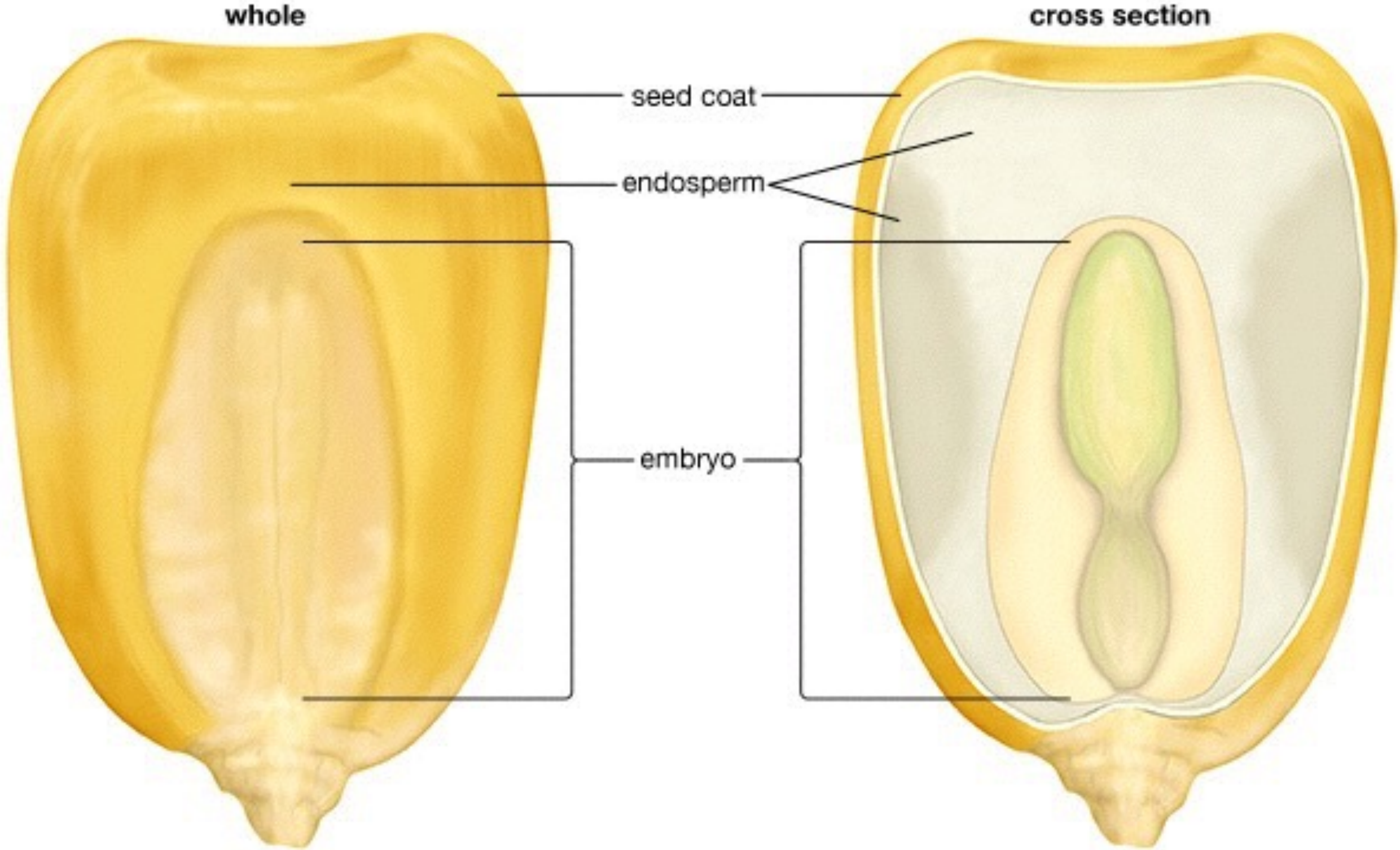
dtrace4linux

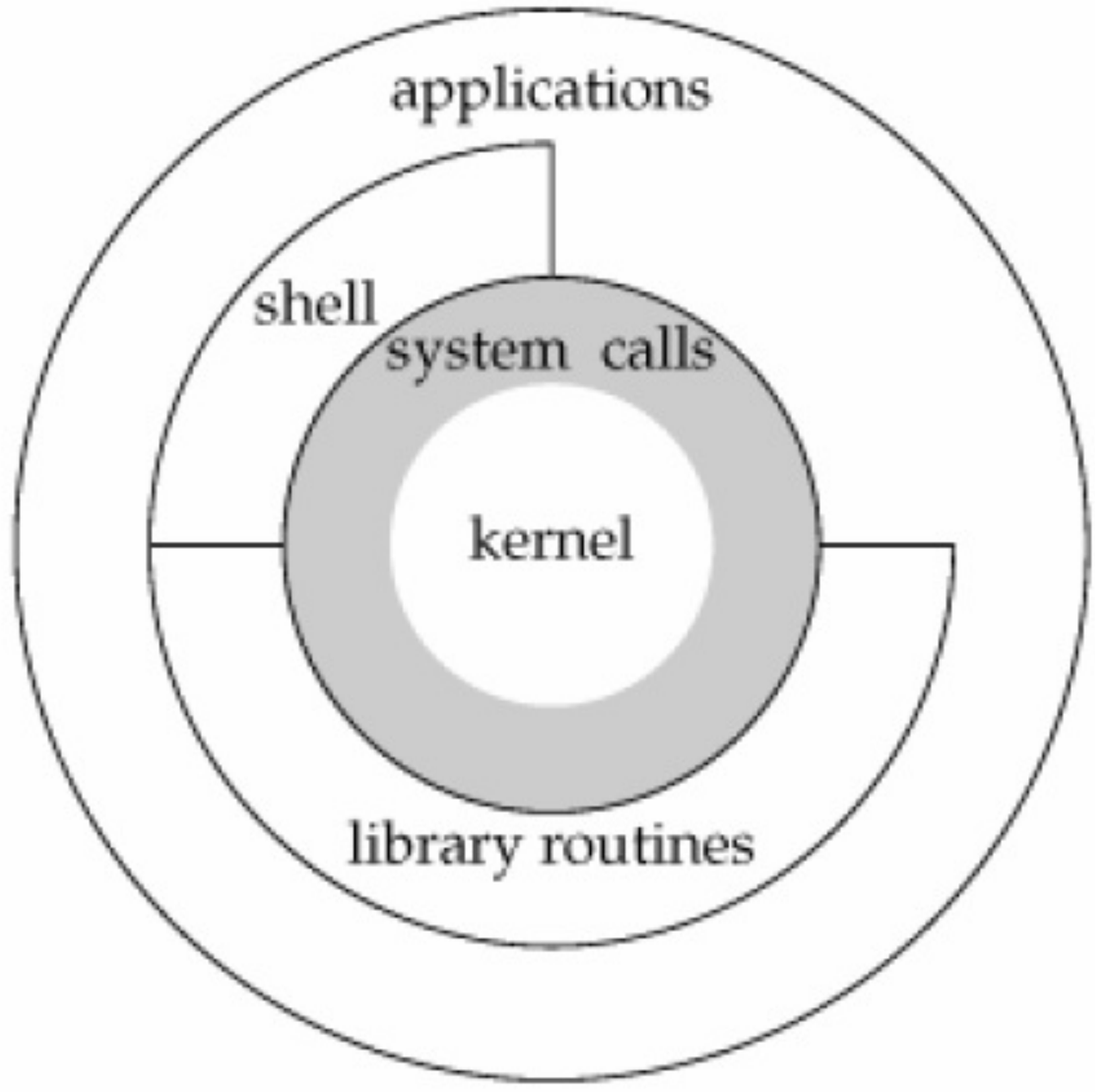
OEL DTrace

sysdig

- (Thanks Deirdré Straughan & General Zoi's Pony Creator)

Corn kernel



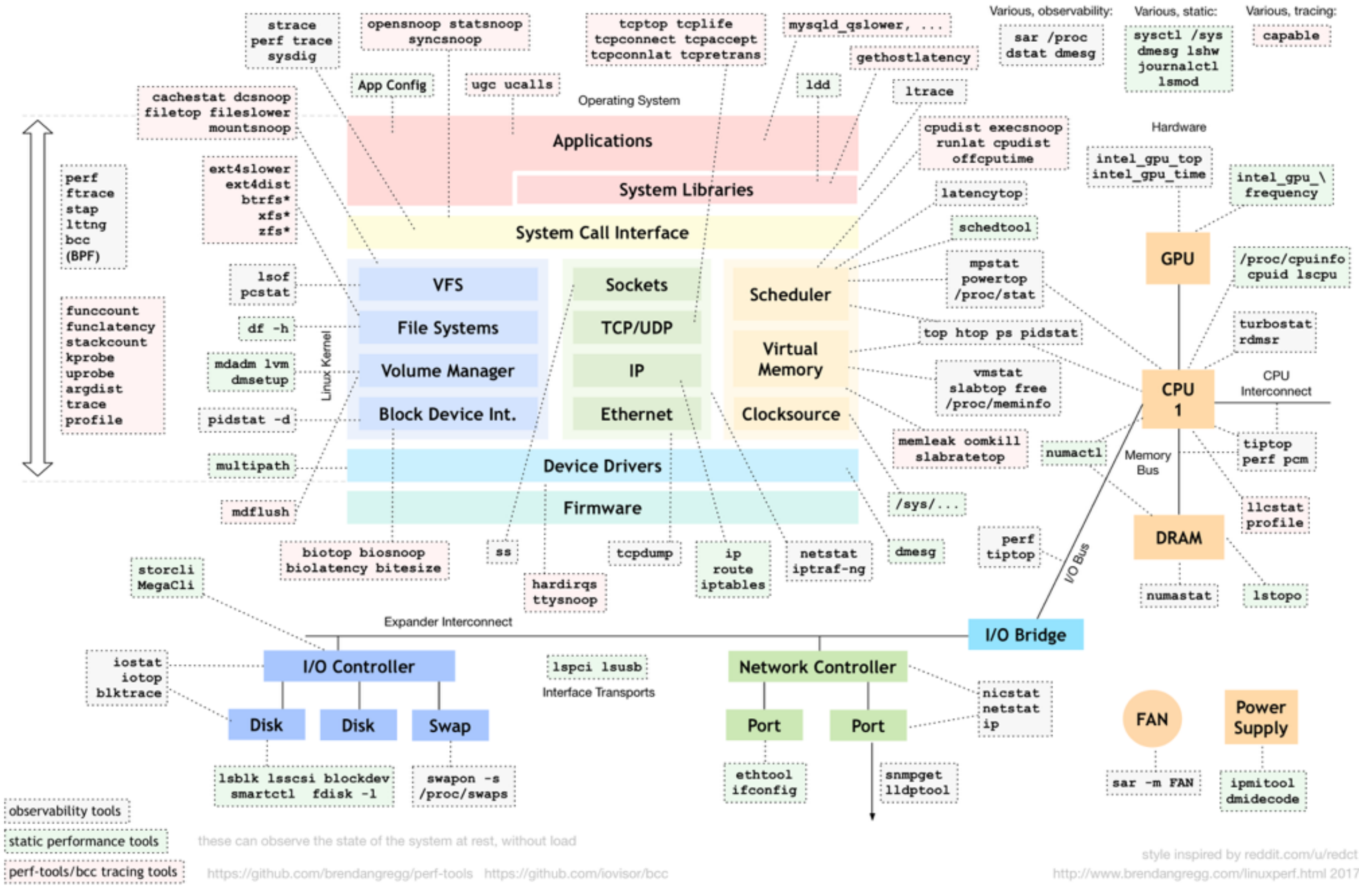

applications
shell
system calls
kernel
library routines

PERFORMANCE MONITORING AND TRACING

SPYING ON YOUR SYSTEM

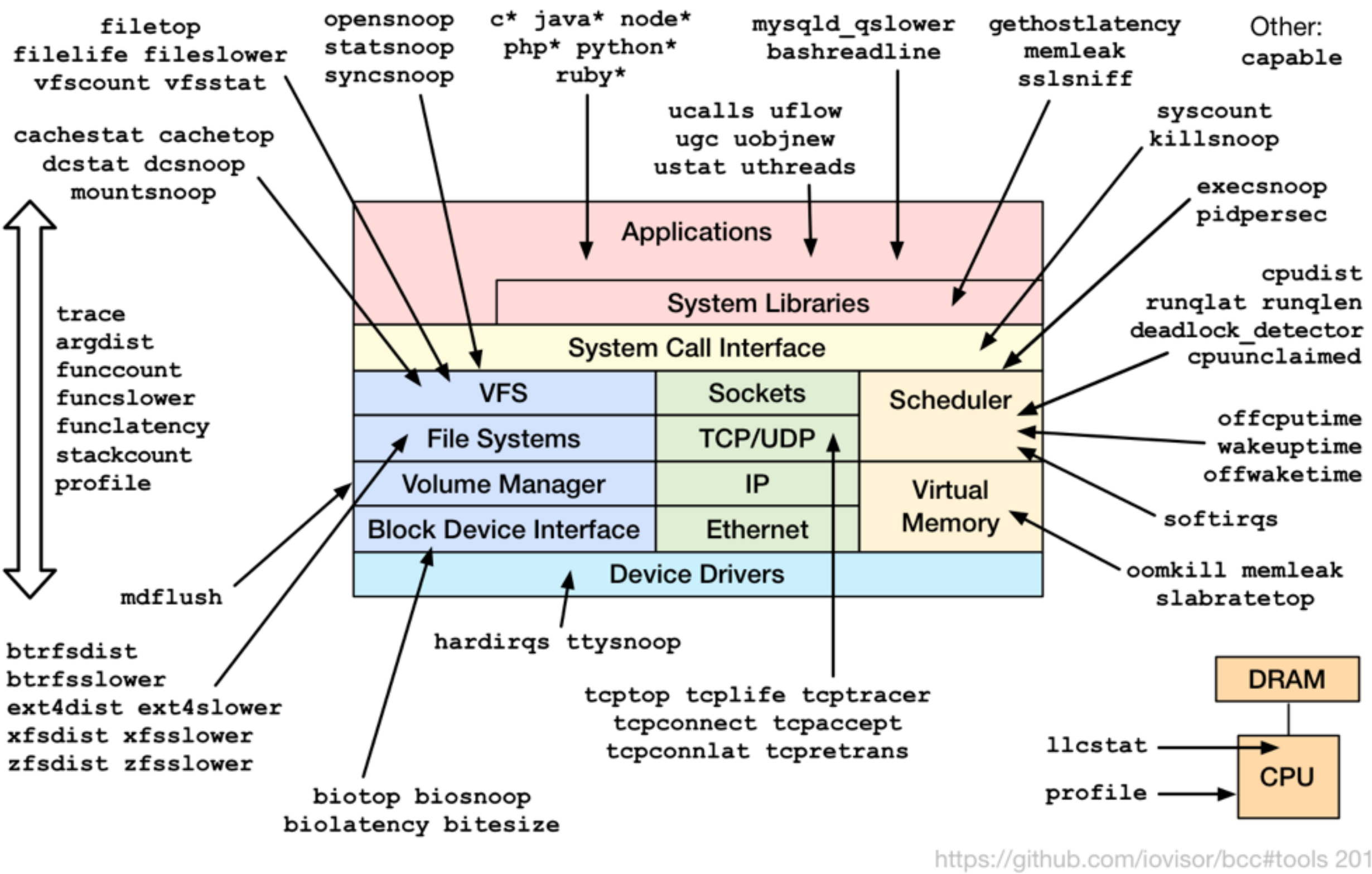
LINUX PERFORMANCE ANALYSIS IN 60,000 MILLISECONDS

```
# uptime
# dmesg | tail
# vmstat 1
# mpstat -P ALL 1
# pidstat 1
# iostat -xz 1
# free -m
# sar -n DEV 1
# sar -n TCP,ETCP 1
# top
```



Source: http://www.brendangregg.com/Perf/linux_perf_tools_full.png

Linux bcc/BPF Tracing Tools



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

Source: http://www.brendangregg.com/Perf/bcc_tracing_tools.png

STRACE

- Hard to customise
- Impacts performance heavily

```
# /usr/share/bcc/tools/opensnoop -n python3
PID    COMM               FD ERR PATH
20526  python3             3   0 /etc/ld.so.cache
20526  python3             3   0 /lib/x86_64-linux-gnu/libpthread.so.
20526  python3             3   0 /lib/x86_64-linux-gnu/libc.so.6
20526  python3             3   0 /lib/x86_64-linux-gnu/libdl.so.2
20526  python3             3   0 /lib/x86_64-linux-gnu/libutil.so.1
20526  python3             3   0 /lib/x86_64-linux-gnu/libexpat.so.1
20526  python3             3   0 /lib/x86_64-linux-gnu/libz.so.1
20526  python3             3   0 /lib/x86_64-linux-gnu/libm.so.6
20526  python3             3   0 /usr/lib/locale/locale-archive
20526  python3             3   0 /usr/lib/x86_64-linux-gnu/gconv/
gconv-modules.cache
```

```
# strace python3 --version
execve("/usr/bin/python3", ["python3", "--version"], [/* 20 vars */]) = 0
brk(NULL)                               = 0x27d5000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f698a5f1000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=26515, ...}) = 0
mmap(NULL, 26515, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f698a5ea000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/x86_64-linux-gnu/libpthread.so.0", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\260`\0\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=138696, ...}) = 0
mmap(NULL, 2212904, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f698a1b1000
mprotect(0x7f698a1c9000, 2093056, PROT_NONE) = 0
mmap(0x7f698a3c8000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x17000) = 0x7f698a3c8000
mmap(0x7f698a3ca000, 13352, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f698a3ca000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\t\2\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=1868984, ...}) = 0
mmap(NULL, 3971488, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f6989de7000
mprotect(0x7f6989fa7000, 2097152, PROT_NONE) = 0
mmap(0x7f698a1a7000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1c0000) = 0x7f698a1a7000
mmap(0x7f698a1ad000, 14752, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f698a1ad000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/x86_64-linux-gnu/libdl.so.2", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\240\r\0\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0644, st_size=14608, ...}) = 0
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f698a5e9000
mmap(NULL, 2109680, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f6989be3000
mprotect(0x7f6989be6000, 2093056, PROT_NONE) = 0
mmap(0x7f6989de5000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f6989de5000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/x86_64-linux-gnu/libutil.so.1", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0`\16\0\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0644, st_size=10656, ...}) = 0
mmap(NULL, 2105608, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f69899e0000
mprotect(0x7f69899e2000, 2093056, PROT_NONE) = 0
mmap(0x7f6989be1000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7f6989be1000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/x86_64-linux-gnu/libexpat.so.1", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0;\0\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0644, st_size=166032, ...}) = 0
mmap(NULL, 2261096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f69897b7000
mprotect(0x7f69897dd000, 2097152, PROT_NONE) = 0
mmap(0x7f69899dd000, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x26000) = 0x7f69899dd000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/x86_64-linux-gnu/libz.so.1", O_RDONLY|O_CLOEXEC) = 3
```

```
#!/usr/bin/python
from __future__ import print_function
from bcc import BPF
import argparse
from socket import inet_ntop, ntohs, AF_INET, AF_INET6
from struct import pack
import ctypes as ct
from time import strftime

bpf_text = """
#include <uapi/linux/ptrace.h>
#define KBUILD_MODNAME "foo"
#include <linux/tcp.h>
#include <net/sock.h>
#include <bcc/proto.h>

BPF_HASH(birth, struct sock *, u64);

// separate data structs for ipv4 and ipv6
struct ipv4_data_t {
    // XXX: switch some to u32's when supported
    u64 ts_us;
    u64 pid;
    u64 saddr;
    u64 daddr;
    u64 ports;
    u64 rx_b;
    u64 tx_b;
    u64 span_us;
    char task[TASK_COMM_LEN];
};
BPF_PERF_OUTPUT(ipv4_events);

struct ipv6_data_t {
    u64 ts_us;
    u64 pid;
    unsigned __int128 saddr;
    unsigned __int128 daddr;
    u64 ports;
    u64 rx_b;
    u64 tx_b;
    u64 span_us;
    char task[TASK_COMM_LEN];
};
BPF_PERF_OUTPUT(ipv6_events);

struct id_t {
    u32 pid;
    char task[TASK_COMM_LEN];
};
BPF_HASH(whoami, struct sock *, struct id_t);

int kprobe__tcp_set_state(struct pt_regs *ctx, struct sock *sk, int state)
{
    u32 pid = bpf_get_current_pid_tgid() >> 32;

    // lport is either used in a filter here, or later
    u16 lport = sk->__sk_common.skc_num;
    FILTER_LPORT

    // dport is either used in a filter here, or later
    u16 dport = sk->__sk_common.skc_dport;
    FILTER_DPORT

    /*
     * This tool includes PID and comm context. It's best effort, and may
     * be wrong in some situations. It currently works like this:
     * - record timestamp on any state < TCP_FIN_WAIT1
     * - cache task context on:
```

THIS IS WHERE THINGS START GETTING MORE MAGICAL

perf

PERF

```
perf record -F 99 -a -g -- sleep 60

perf report --sort comm,dso
```

FLAME GRAPHS

```
perf record -F 99 -a -g -- sleep 60

perf script > out.perf

./stackcollapse-perf.pl out.perf > out.folded

./flamegraph.pl out.folded > perf.svg
```

Example: http://www.brendangregg.com/FlameGraphs/example-perf.svg

```
#!/usr/bin/python
from __future__ import print_function
from bcc import BPF
import argparse
from socket import inet_ntop, ntohs, AF_INET, AF_INET6
from struct import pack
import ctypes as ct
from time import strftime

bpf_text = """
#include <uapi/linux/ptrace.h>
#define KBUILD_MODNAME "foo"
#include <linux/tcp.h>
#include <net/sock.h>
#include <bcc/proto.h>

BPF_HASH(birth, struct sock *, u64);

// separate data structs for ipv4 and ipv6
struct ipv4_data_t {
    // XXX: switch some to u32's when supported
    u64 ts_us;
    u64 pid;
    u64 saddr;
    u64 daddr;
    u64 ports;
    u64 rx_b;
    u64 tx_b;
    u64 span_us;
    char task[TASK_COMM_LEN];
};
BPF_PERF_OUTPUT(ipv4_events);

struct ipv6_data_t {
    u64 ts_us;
    u64 pid;
    unsigned __int128 saddr;
    unsigned __int128 daddr;
    u64 ports;
    u64 rx_b;
    u64 tx_b;
    u64 span_us;
    char task[TASK_COMM_LEN];
};
BPF_PERF_OUTPUT(ipv6_events);

struct id_t {
    u32 pid;
    char task[TASK_COMM_LEN];
};
BPF_HASH(whoami, struct sock *, struct id_t);

int kprobe__tcp_set_state(struct pt_regs *ctx, struct sock *sk, int state)
{
    u32 pid = bpf_get_current_pid_tgid() >> 32;

    // lport is either used in a filter here, or later
    u16 lport = sk->__sk_common.skc_num;
    FILTER_LPORT

    // dport is either used in a filter here, or later
    u16 dport = sk->__sk_common.skc_dport;
    FILTER_DPORT

    /*
     * This tool includes PID and comm context. It's best effort, and may
     * be wrong in some situations. It currently works like this:
     * - record timestamp on any state < TCP_FIN_WAIT1
     * - cache task context on:
```

AND THEN IT GETS EVEN MORE MAGICAL

eBPF

Tools for Witchcraft

- BPF_PERF_OUTPUT
- BPF_TABLE
- BPF_HASH
- BPF_STACK_TRACE
- BPF_ARRAY
- BPF_HISTOGRAM
- BPF_PERF_ARRAY
- BPF_PERCPU_ARRAY

Hello World

```
#!/usr/bin/env python
from bcc import BPF
from time import sleep

b = BPF(text="""
int kprobe__sys_clone(void *ctx) {
    bpf_trace_printk("Hello, World!\\n");
    return 0;
}
""")

# tail /sys/kernel/debug/tracing/trace_pipe
bpf.trace_print()
```

```
# ./examples/hello_world.py
World!      sshd-1708  [001] d... 48979.876007: : Hello,
World!     <...>-23754 [001] d... 48979.895665: : Hello,
World!      sshd-23754 [001] d... 48981.184813: : Hello,
World!        sh-23757 [001] d... 48981.187325: : Hello,
World! run-parts-23758 [000] d... 48981.190149: : Hello,
World! 00-header-23759 [001] d... 48981.191524: : Hello,
World! 00-header-23759 [001] d... 48981.192774: : Hello,
World! 00-header-23759 [001] d... 48981.194310: : Hello,
World! run-parts-23758 [000] d... 48981.196453: : Hello,
World! run-parts-23758 [000] d... 48981.198322: : Hello,
World! run-parts-23758 [000] d... 48981.200000: : Hello,
World! 90-updates-avai-23765 [000] d... 48981.201821: : Hello,
World! run-parts-23758 [000] d... 48981.203598: : Hello,
World! 91-release-upgr-23767 [000] d... 48981.205087: : Hello,
World!     <...>-23768 [001] d... 48981.205652: : Hello,
World! 91-release-upgr-23768 [001] d... 48981.205895: : Hello,
World! 91-release-upgr-23767 [001] d... 48981.347179: : Hello,
World!      sshd-23754 [001] d... 48981.184813: : Hello,
World!        sh-23757 [001] d... 48981.187325: : Hello,
World! run-parts-23758 [000] d... 48981.190149: : Hello,
```

Hello More Fields

```
#!/usr/bin/env python

from bcc import BPF

b = BPF(text="""
int kprobe__sys_clone(void *ctx) {
    bpf_trace_printk("Hello, World!\\n");
    return 0;
}
""")

print("%-18s %-16s %-6s %s" % ("TIME(s)", "COMM", "PID", "MESSAGE"))

while True:
    try:
        (task, pid, cpu, flags, ts, msg) = b.trace_fields()
    except ValueError:
        continue
    print("%-18.9f %-16s %-6d %s" % (ts, task, pid, msg))
```

```
# examples/tracing/hello_fields.py
TIME(s)            COMM             PID    MESSAGE
50770.163158000    sshd             1708   Hello, World!
50770.171591000    <...>            24194  Hello, World!
50771.414130000    sshd             24194  Hello, World!
50771.415548000    sh               24199  Hello, World!
50771.416817000    sshd             1708   Hello, World!
50771.417001000    run-parts        24200  Hello, World!
50771.417860000    <...>            24201  Hello, World!
50771.418627000    00-header        24201  Hello, World!
50771.419441000    00-header        24201  Hello, World!
50771.420249000    run-parts        24200  Hello, World!
50771.421026000    run-parts        24200  Hello, World!
50771.421918000    run-parts        24200  Hello, World!
50771.422607000    90-updates-avai  24208  Hello, World!
50771.423555000    run-parts        24200  Hello, World!
50771.424070000    91-release-upgr  24210  Hello, World!
50771.424206000    91-release-upgr  24211  Hello, World!
```

Hello BPF_PERF_OUTPUT

```
#!/usr/bin/env python
from bcc import BPF
import ctypes as ct

b = BPF(text="""
#include <linux/sched.h>

// define output data structure in C
struct data_t {
    u32 pid;
    u64 ts;
    char comm[TASK_COMM_LEN];
};
BPF_PERF_OUTPUT(events);

int hello(struct pt_regs *ctx) {
    struct data_t data = {};
    data.pid = bpf_get_current_pid_tgid();
    data.ts = bpf_ktime_get_ns();
    bpf_get_current_comm(&data.comm, sizeof(data.comm));

    events.perf_submit(ctx, &data, sizeof(data));

    return 0;
}
""")

TASK_COMM_LEN = 16    # linux/sched.h
class Data(ct.Structure):
    _fields_ = [("pid", ct.c_uint),
                ("ts", ct.c_ulonglong),
                ("comm", ct.c_char * TASK_COMM_LEN)]

print("%-18s %-16s %-6s %s" % ("TIME(s)", "COMM", "PID", "MESSAGE"))

start = 0
def print_event(cpu, data, size):
    global start
    event = ct.cast(data, ct.POINTER(Data)).contents
    if start == 0:
            start = event.ts
    time_s = (float(event.ts - start)) / 1000000000
    print("%-18.9f %-16s %-6d %s" % (time_s, event.comm, event.pid,
        "Hello, perf_output!"))

b["events"].open_perf_buffer(print_event)
```

```
# ./examples/tracing/hello_perf_output.py
TIME(s)            COMM             PID    MESSAGE
0.000000000        sshd             1708   Hello, perf_output!
0.020348293        sshd             23934  Hello, perf_output!
1.709378491        sshd             23934  Hello, perf_output!
1.710825313        sh               23939  Hello, perf_output!
1.712730637        run-parts        23940  Hello, perf_output!
1.713539655        00-header        23941  Hello, perf_output!
1.714276330        00-header        23941  Hello, perf_output!
1.714813753        00-header        23941  Hello, perf_output!
1.715577598        run-parts        23940  Hello, perf_output!
1.716294354        run-parts        23940  Hello, perf_output!
1.717268885        run-parts        23940  Hello, perf_output!
1.718018553        90-updates-avai  23947  Hello, perf_output!
1.719002588        run-parts        23940  Hello, perf_output!
1.719772776        91-release-upgr  23949  Hello, perf_output!
1.720038403        91-release-upgr  23950  Hello, perf_output!
1.720197178        91-release-upgr  23950  Hello, perf_output!
1.792543698        release-upgrade  23949  Hello, perf_output!
1.794060920        release-upgrade  23949  Hello, perf_output!
1.795520303        release-upgrade  23949  Hello, perf_output!
1.796643253        run-parts        23940  Hello, perf_output!
1.797447438        97-overlayroot   23956  Hello, perf_output!
1.797651202        97-overlayroot   23957  Hello, perf_output!
1.797803991        97-overlayroot   23957  Hello, perf_output!
1.800281656        run-parts        23940  Hello, perf_output!
1.801679488        update-motd-fsc  23960  Hello, perf_output!
1.803061663        update-motd-fsc  23960  Hello, perf_output!
1.803530425        update-motd-fsc  23962  Hello, perf_output!
1.810352576        update-motd-fsc  23960  Hello, perf_output!
1.811627500        update-motd-fsc  23960  Hello, perf_output!
1.812744521        run-parts        23940  Hello, perf_output!
2.129480732        sshd             23934  Hello, perf_output!
2.134354001        bash             23967  Hello, perf_output!
2.136230270        bash             23968  Hello, perf_output!
```

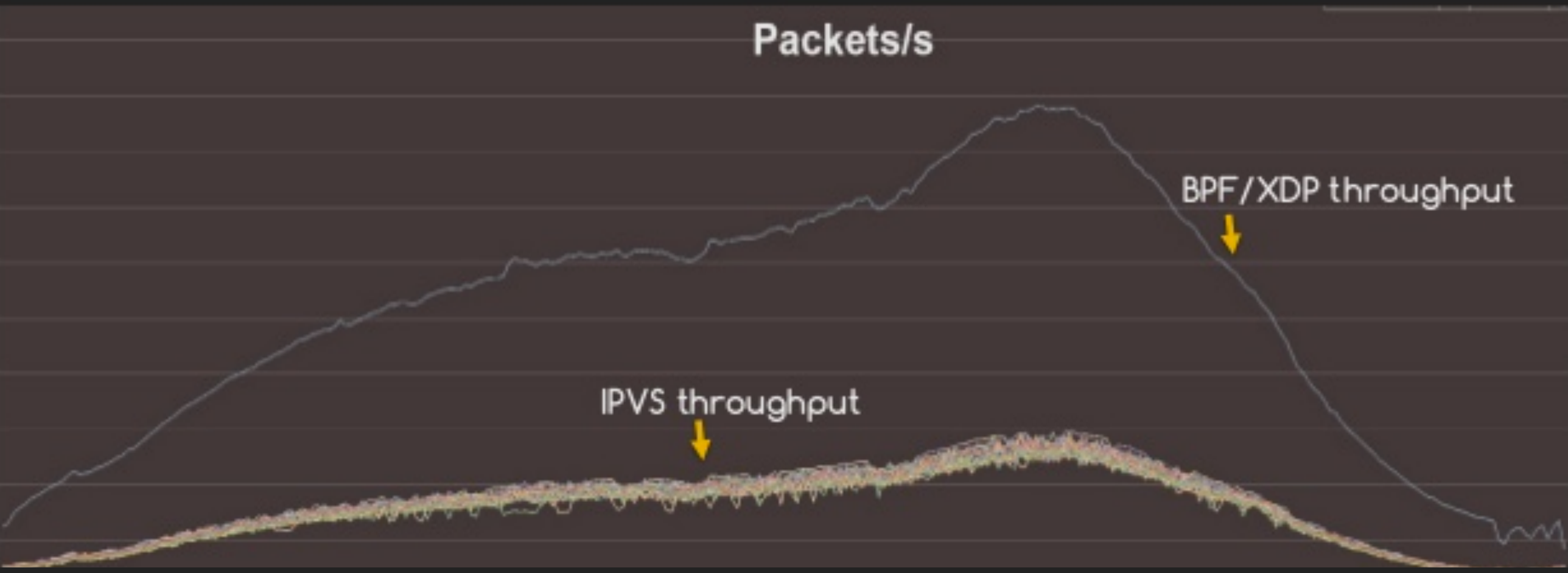
THE FUTURE AND BEYOND, BUT ALSO THE PRESENT

GOING FORWARD

Ply, inspired by dtrace, and therefore awk

```
ply -c 'kretprobe:SyS_read { @.quantize(retval()) }'

ply -c 'kprobe:SyS_read / (arg(2) > 1024) / { @[pid()].quantize(arg(2)); }'

ply -c 'kprobe:SyS_open { printf("%16s(%5d): %s\n", comm(), pid(), mem(arg(0), "128s")) }'

ply -c 'kprobe:SyS_* { @[func()].count() }'

ply -c 'kprobe:SyS_* { @[comm(), pid()].count() }'

ply -c 'kprobe:schedule { @[stack()].count() }'
```

Facebook published BPF/XDP numbers for L3/L4 LB at Netdev 2.1



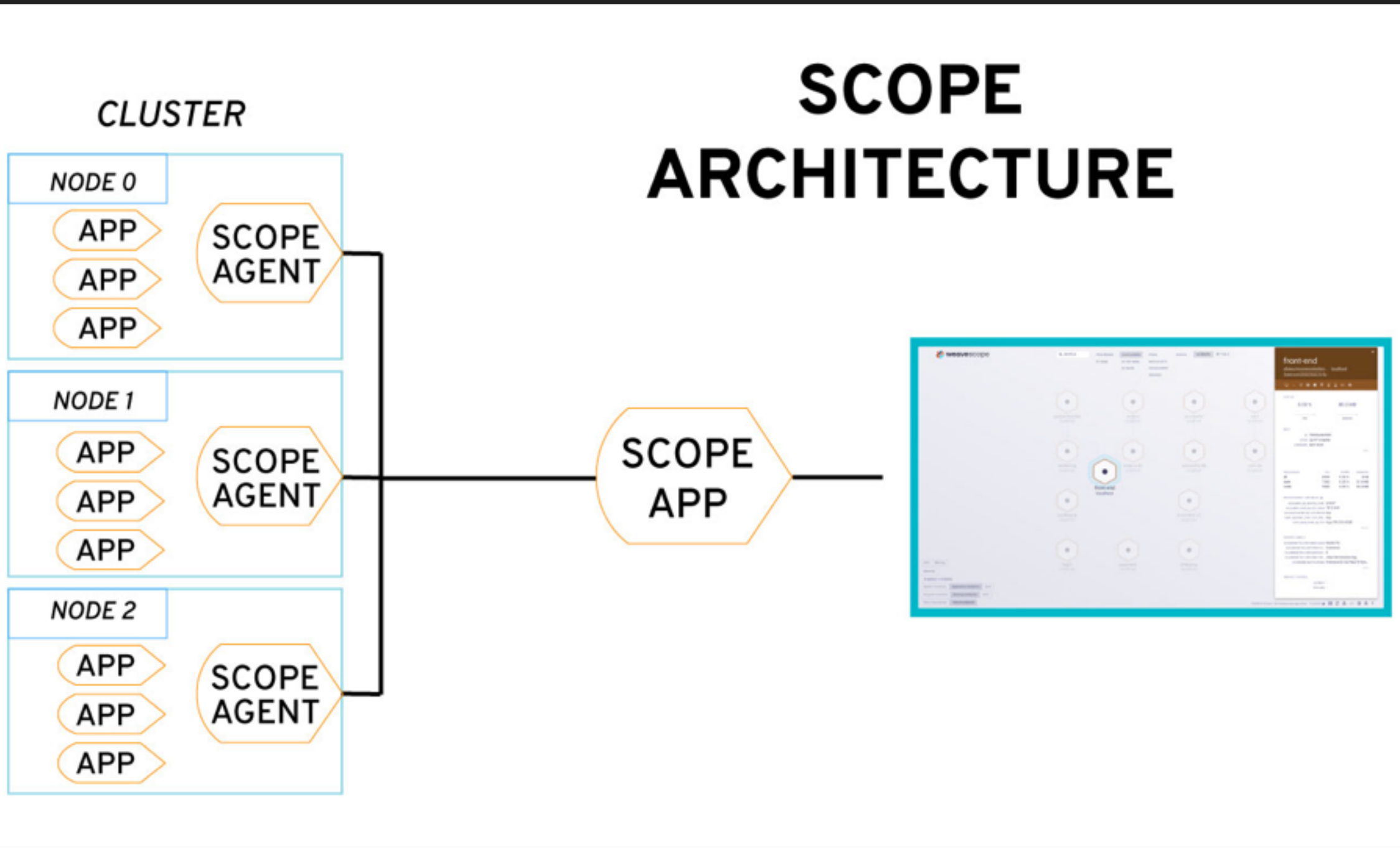
Source of annotations: https://www.slideshare.net/ThomasGraf5/cilium-network-security-for-microservices/9

Original Source: https://www.netdevconf.org/2.1/slides/apr6/zhou-netdev-xdp-2017.pdf

CILIUM

```
[{
    "endpointSelector": {"matchLabels":{"id.empire.deathstar":""}},
    "ingress": [{
        "fromEndpoints": [
          {"matchLabels":{"id.spaceship":""}},
          {"matchLabels":{"reserved:host":""}}
        ]
    }]
},{
    "endpointSelector": {"matchLabels":{"id.spaceship":""}},
    "egress": [{
    "toPorts": [{
        "ports": [
                {"port": "80", "protocol": "tcp"}
            ],
        "rules": {
                "HTTP": [
                    {
                        "method": "GET",
                        "path": "/v1/"
                    },{
                        "method": "POST",
                        "path": "/v1/request-landing/"
                    },{
                        "method": "PUT",
                        "path": "/v1/exhaust-port/",
                        "headers": ["X-Has-Force: true"]
```

WEAVEWORKS' SCOPE



DISCLAIMER II

QUESTIONS?