Granular CPU Capacity Management at Scale with eBPF

SREcon24 EMEA

George Brighton & Cameron Howes, Market Data Engineering SRE

31 October 2024

Engineering Division

Previous Talks

Goldman Sachs @ SREcon EMEA

Market Data: Applying SRE Techniques to Legacy Designs

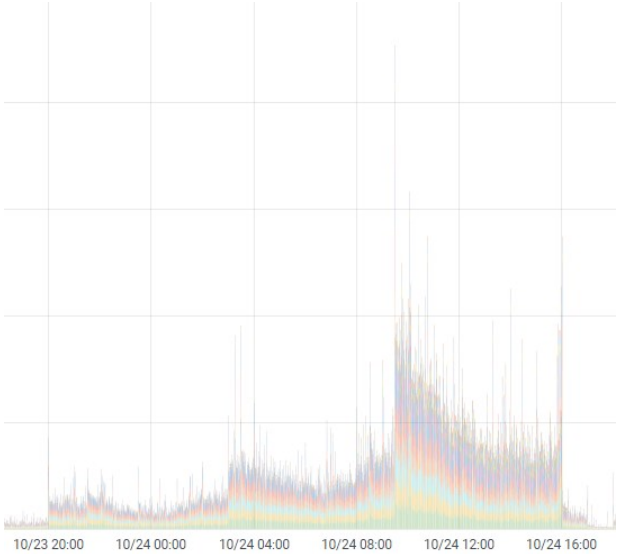
2022

Dealing with Dark Debt: Lessons Learnt at Goldman Sachs

2018

Agenda

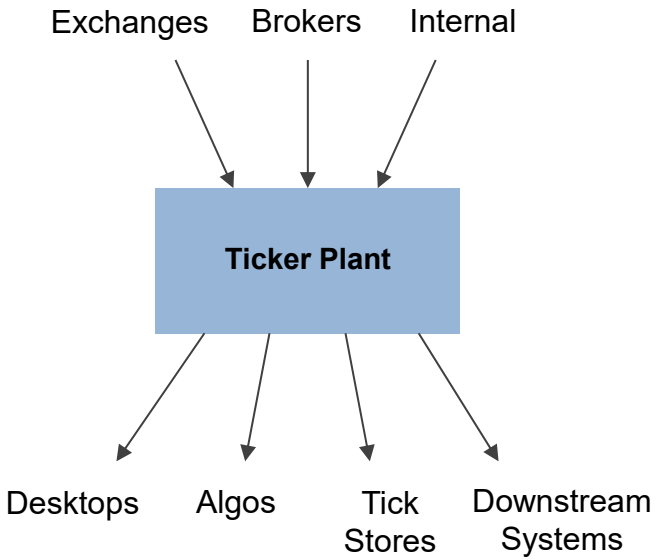
1. Prerequisite knowledge
2. The outage
3. Our requirements
4. PSI and BPF
5. BPF implementation playbook



Ticker Plant

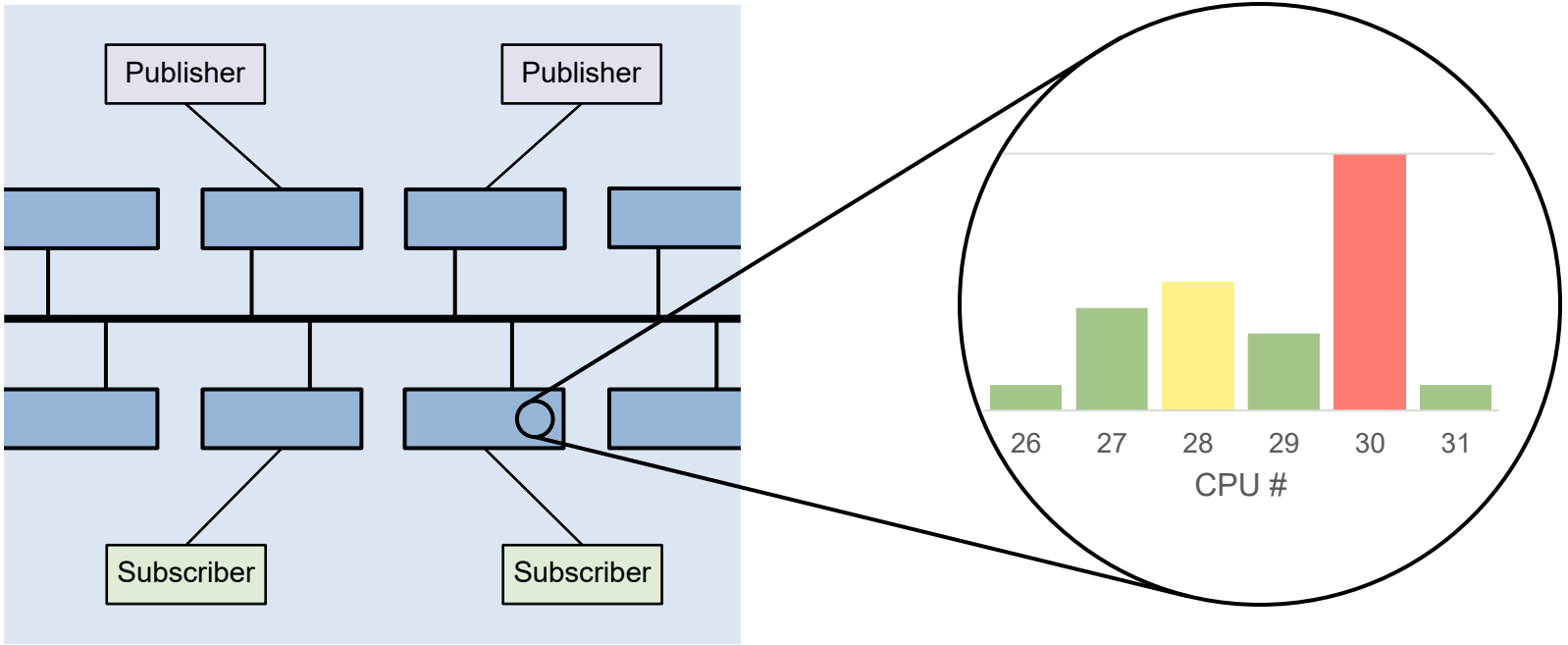
Overview

- Provision and distribution of pricing and trade-related data for financial instruments
  - 100s of feeds
  - 10,000s of internal clients
  - 100,000,000,000s of updates per day
- Salient properties
  - Vendor software
  - Bare metal, with manual CPU affinity
  - Multicast



Ticker Plant

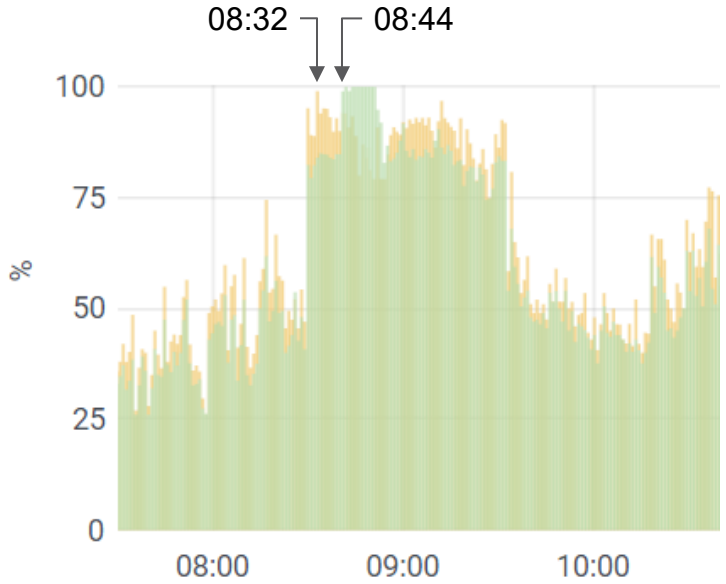
Deployment



The Outage

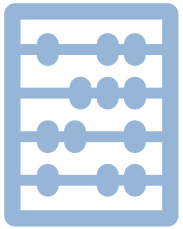
14 February 2023

08:30	Markets open
08:32	Jan CPI report released, update rates surge, first receiver buffer reaches capacity
08:32:28	Multiple alerts fire
08:44	A second core becomes saturated
08:50	Engineers begin disabling feeds
09:35	Impact mitigated



All times EST.

Our Requirements



Single headroom multiple per cluster

Time to market

Global consistency

Pressure Stall Information (PSI)

Linux 4.20+
(backported to RHEL8)

Accumulated microseconds tasks have spent waiting due to lack of CPU

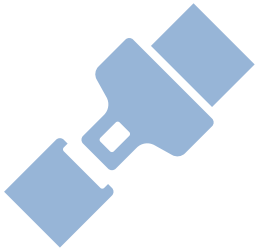
```
$ cat /proc/pressure/cpu
some avg10=0.05 avg60=0.04 avg300=0.01 total=5230672
```

Wait time *some* tasks have spent waiting as a ratio of the last 10s/1m/5m (may be >1)

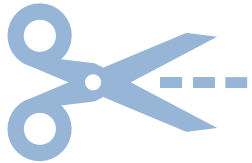
https://docs.kernel.org/accounting/psi.html

Differentiators

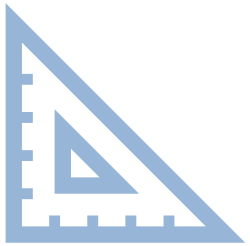
BPF



Safety



Decoupling from application



Granularity

March of Progress

BPF

bcc-tools

- Off-the-shelf utilities
- Minimal BPF knowledge required

bpftrace

- AWK-like language
- Excellent for prototypes

libbpf

- C-like language
- Full access to BPF

CPU Metrics

```
process_cpu_seconds_total 633.37
```

30 seconds later...

```
process_cpu_seconds_total 634.37
```

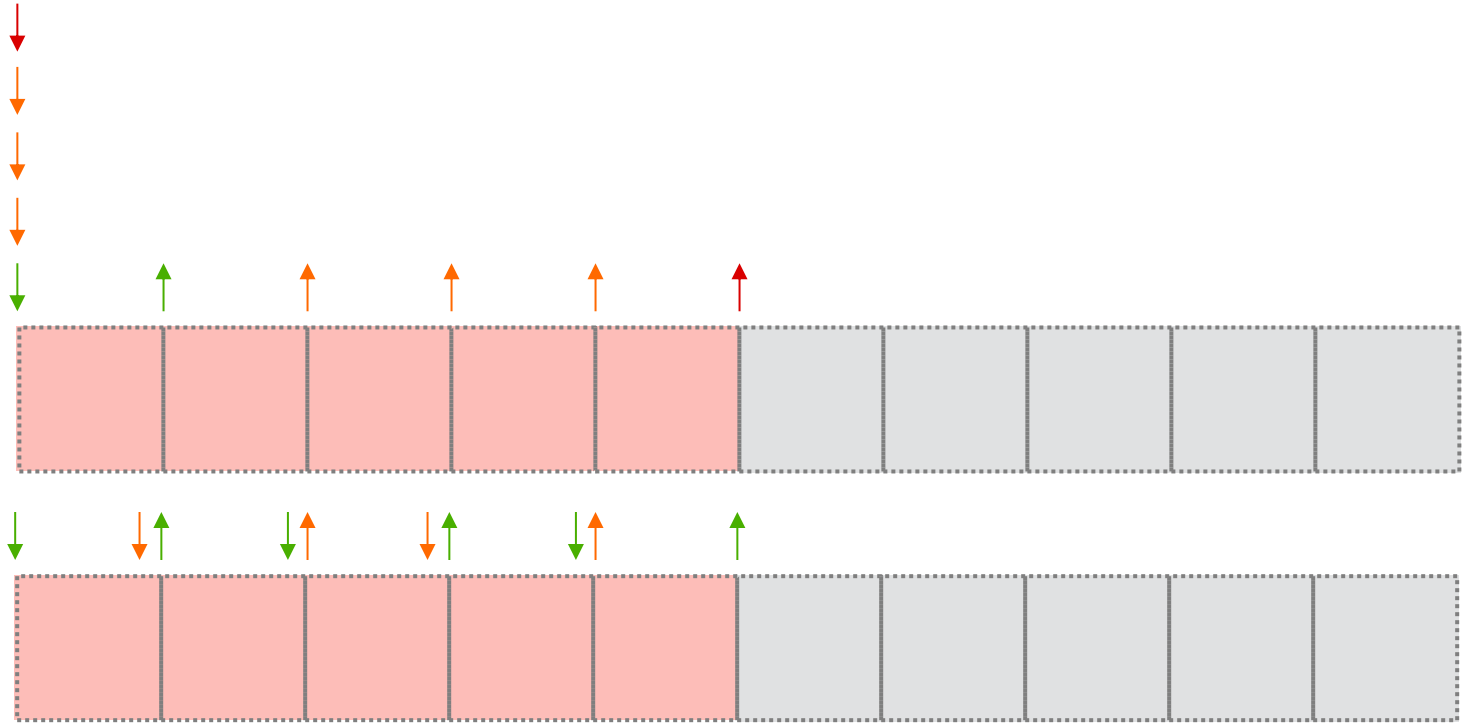
Each second of this scrape interval could have been 33.3ms busy.

Or busy for a full second, and idle the rest of the time...

Threads, cores and percentages!

Goldman Sachs

CPU Usage



Goldman Sachs

bpftrace

```
$ sudo bpftrace -e 'interval:ms:100 { printf("time: %lld\n", nsecs); }'
Attaching 1 probe...
time: 18531631164155
time: 18531731163454
time: 18531831163657
time: 18531931164055

$ sudo bpftrace -e 'profile:ms:100 / cpu == 0 / { printf("time: %lld\n", nsecs); }'
Attaching 1 probe...
time: 18612564274639
time: 18612664274206
time: 18612764274150
time: 18612864274384

$ cat /proc/stat | grep cpu0
cpu0 88627 42771 95002 2586614 118 3645 1292 0 0 0
```

Goldman Sachs

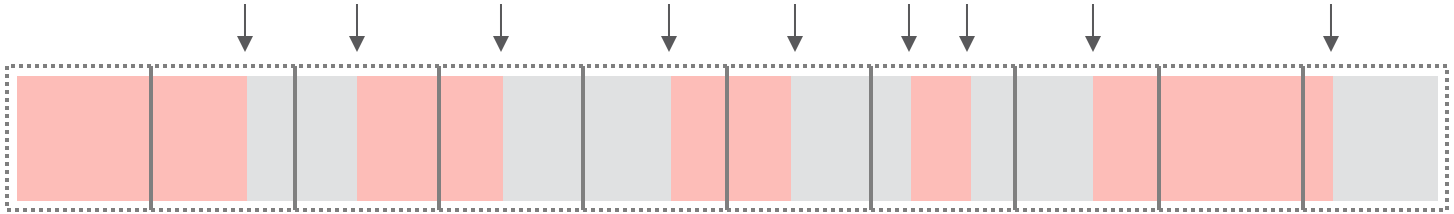
bpftrace

```
$ sudo /usr/share/bcc/tools/tplist | wc -l
2135

$ sudo /usr/share/bcc/tools/tplist | grep sched: | head -n 10
sched:sched_kthread_stop
sched:sched_kthread_stop_ret
sched:sched_waking
sched:sched_wakeup
sched:sched_wakeup_new
sched:sched_switch
sched:sched_migrate_task
sched:sched_process_free
sched:sched_process_exit
sched:sched_wait_task
```

bpftrace

```
tracepoint:sched:sched_switch(char prev_comm[16], pid_t prev_pid, int prev_prio, long prev_state,
                              char next_comm[16], pid_t next_pid, int next_prio) {}
```



Goldman Sachs

libbpf

bpftrace	libbpf
`@variable[cpu]`	`BPF_MAP_TYPE_PERCPU_ARRAY`
`printf`	`BPF_MAP_TYPE_PERF_EVENT_ARRAY` `bpf_perf_event_output`

Goldman Sachs

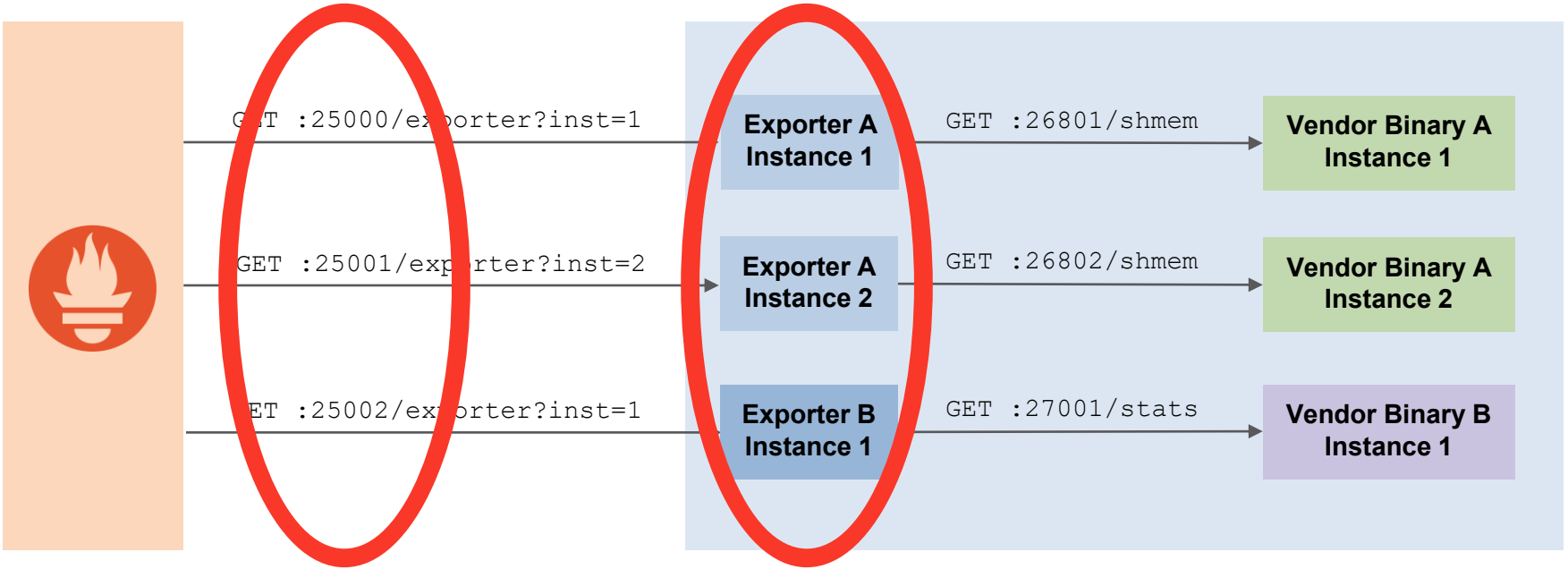
libbpf

Testing?

- https://docs.cilium.io/en/stable/contributing/testing/bpf/
- QEMU
- More creative options...

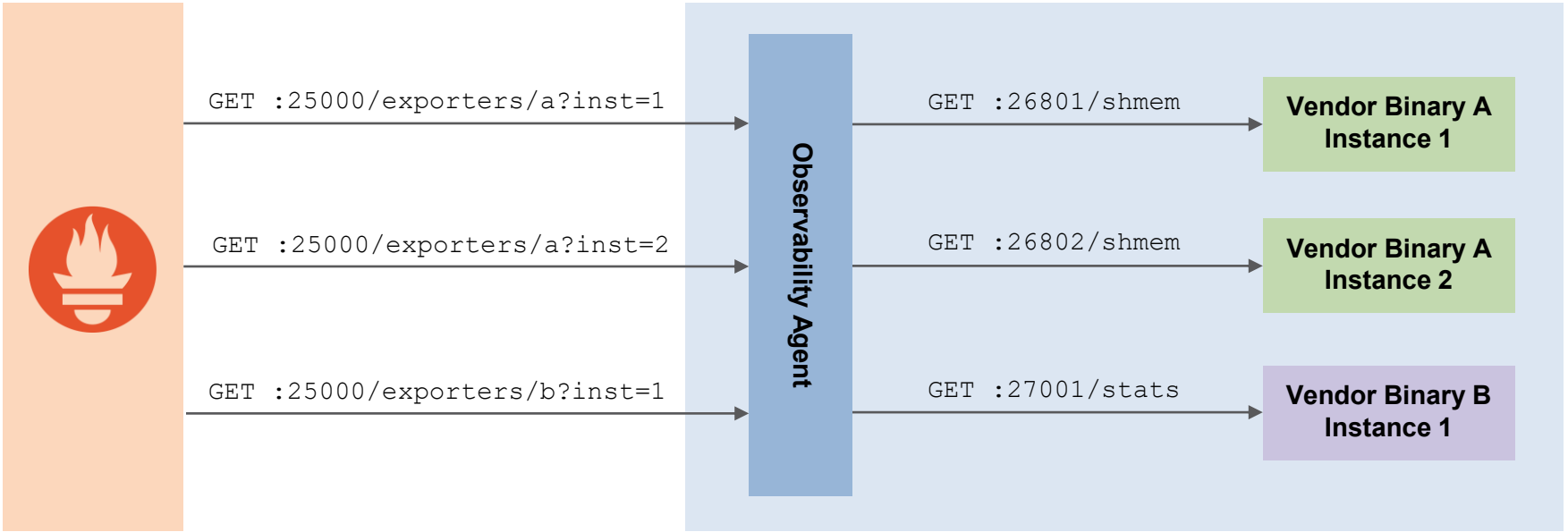
Metrics Exposition

Consolidated Exporter



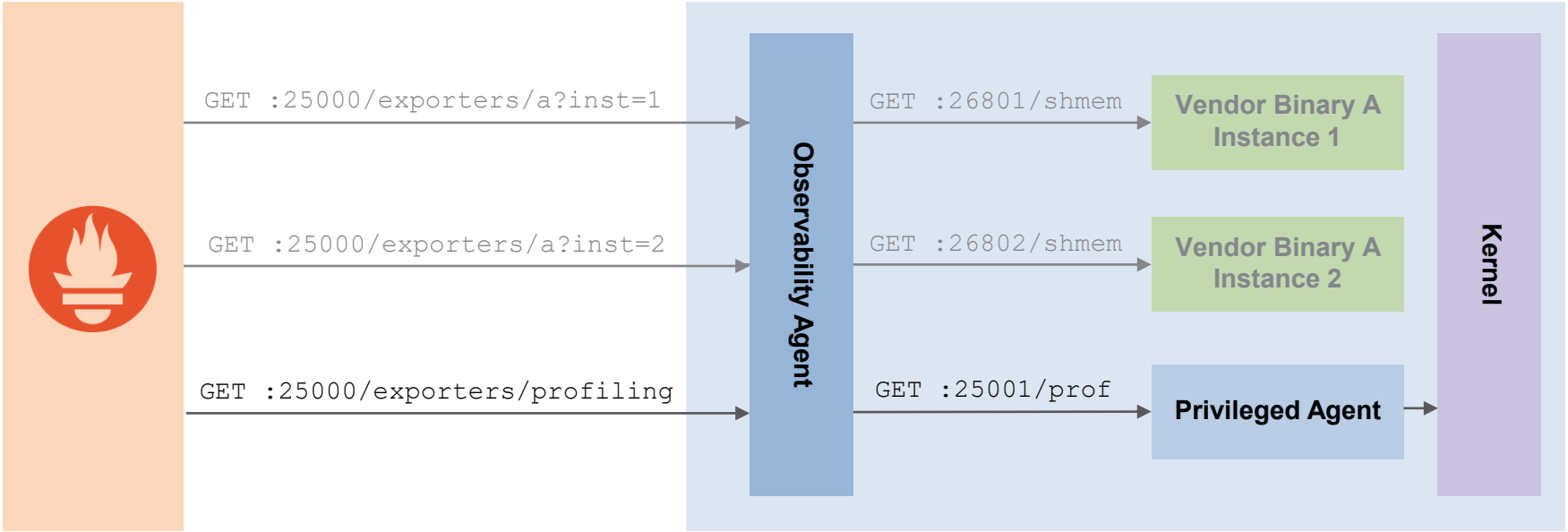
Metrics Exposition

Consolidated Exporter



Metrics Exposition

Privileged Agent



BPF Implementation Playbook

1. Audit kernel versions
   – 3.x: upgrade
   – pre-5.8: CAP_SYS_ADMIN or root required
2. Begin conversations with Linux teams, as necessary
   – Capabilities or sudo approach
3. bpftrace
   – Proof of concept
4. libbpf
   – Only if necessary

Questions