Observability Day

NORTH AMERICA

Observability Day
NORTH AMERICA

Inference Is a Black Box:

How We Built GPU-Aware Observability with Fluent Bit and OpenTelemetry

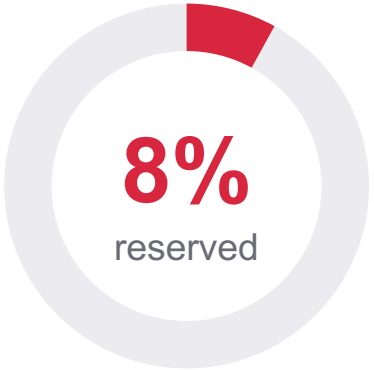
Danilo Silva
Senior Delivery Consultant, AWS

Henrique Santana
Principal Cloud Engineer, AWS

Which one is in trouble?

Two vLLM pods on our test cluster

Pod A



of GPU memory at startup

```
--gpu-memory-utilization=0.08
```

same model

same GPU type

same traffic

Pod B



of GPU memory at startup

```
--gpu-memory-utilization=0.90
```

HANDS UP

- A Pod A
- B Pod B
- C both
- D can't tell

GPU memory stays flat under load

Observability Day
NORTH AMERICA

KnativeCon
NORTH AMERICA

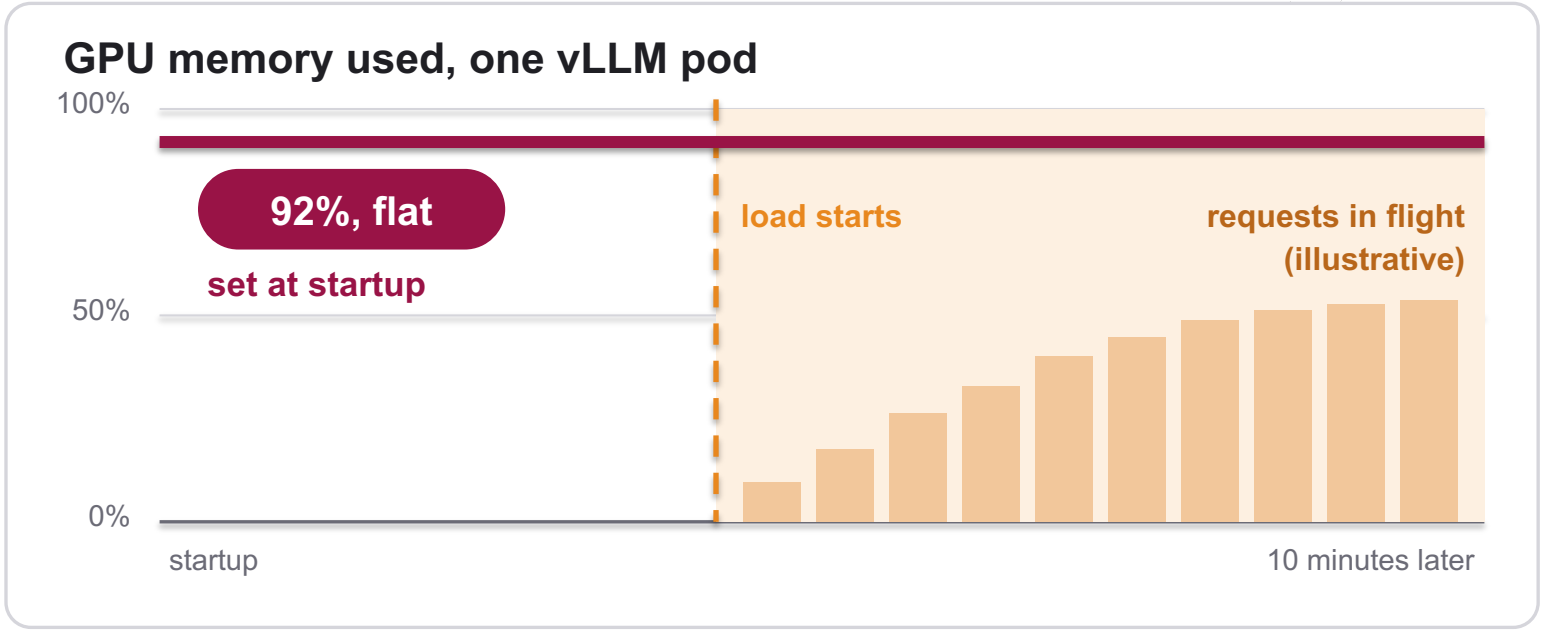
CPU
how busy the cores are

Memory
RAM in use

Network
bytes in and out

Disk
reads and writes

Node and container metrics. None shows requests waiting in vLLM.



vLLM reserves 92% of GPU memory when it starts.

vLLM default: gpu_memory_utilization = 0.92 (vllm/config/cache.py, vLLM main, Oct 2026).

The KV cache runs out first

vLLM logs how full it is every 10 seconds.

One GPU, as vLLM uses it



KV cache: a fixed pool of slots, sized by `--gpu-memory-utilization`

GPU memory used
92%
unchanged since startup

requests waiting
4



time to first token



KV cache usage
100%

every 10 s

```
10:15:10 Engine 000:  ... Running: 10 reqs, Waiting: 4 reqs, GPU KV cache usage: 100.0%, ...
10:15:20 Engine 000:  ... Running: 10 reqs, Waiting: 4 reqs, GPU KV cache usage: 100.0%, ...
```

Intro structure inspired by Jimmy Song, jimmysong.io/blog/gpu-to-token-observability

Observability Day
NORTH AMERICA

DEMO

Is 8% too low? Is 90% too high?

1 GPU › 2 Log › 3 Trace › 4 Answer

Two vLLM pods · same model · same GPU type · same traffic

Demo: which one is in trouble?

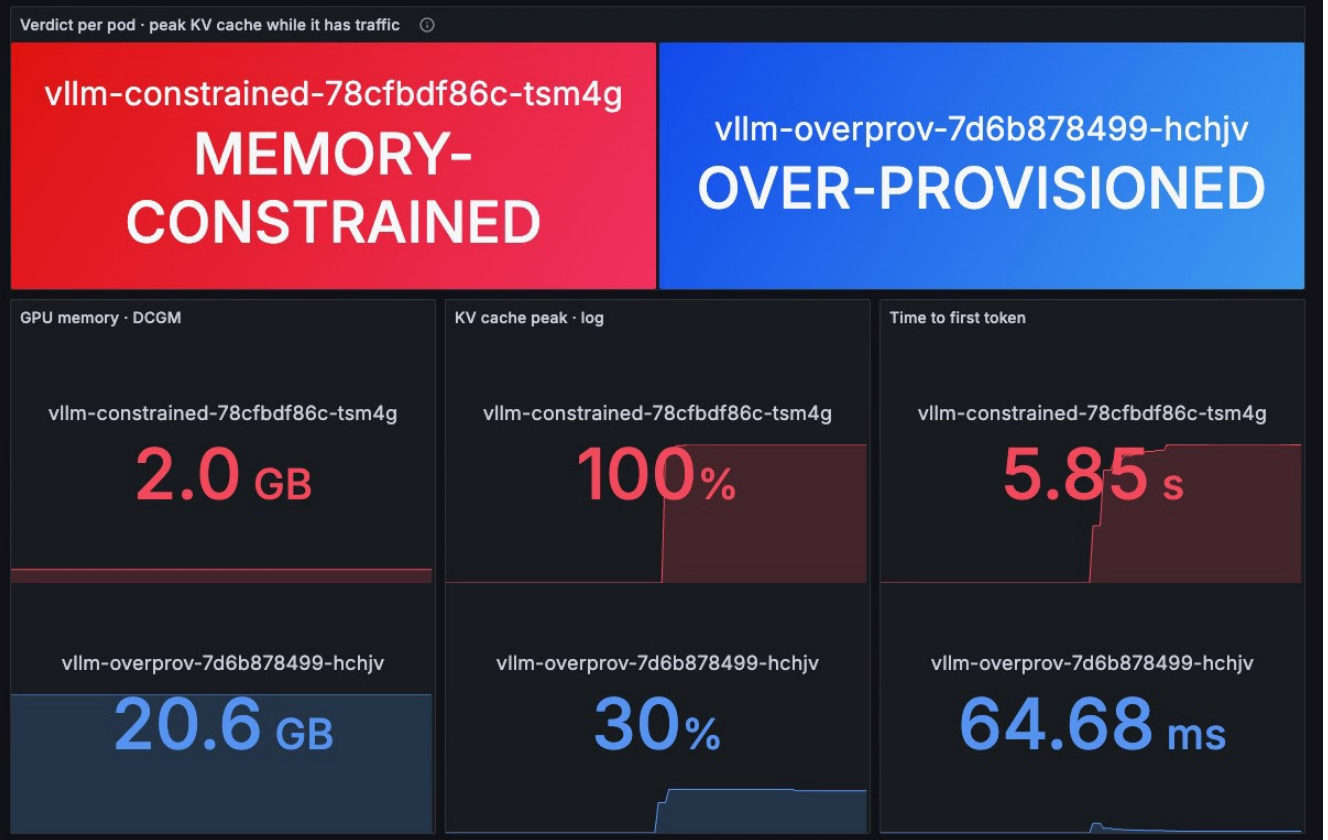
1 GPU › 2 Log › 3 Trace › **4 Answer**

Is 8% too low? Is 90% too high?

rule of thumb (ours, not vLLM's)

```
peak KV cache under real traffic
  > 90%  memory-constrained
  < 40%  over-provisioned
```

Watch the server's own signal, per pod. The GPU memory chart cannot answer this.



8% is too low. 90% is too high. Only the server's own signal could tell.

Who voted both? You were right.

KnativeCon
NORTH AMERICA

Pod A

reserved 8% of GPU memory

Too small

100% KV cache peak

5.9 s to first token

Hurts users.

Pod B

reserved 90% of GPU memory

Oversized

~30% KV cache peak

0.08 s to first token

Fast, but costs money.

GPU memory missed both problems. Watch the server's KV cache.

Measured 16 Oct 2026, our test cluster: vLLM v0.30.0, opt-125m (a small model), same traffic. One whole A10G per pod, so Pod B's spare memory blocks nobody.

What broke while we built it

Observability Day NORTH AMERICA

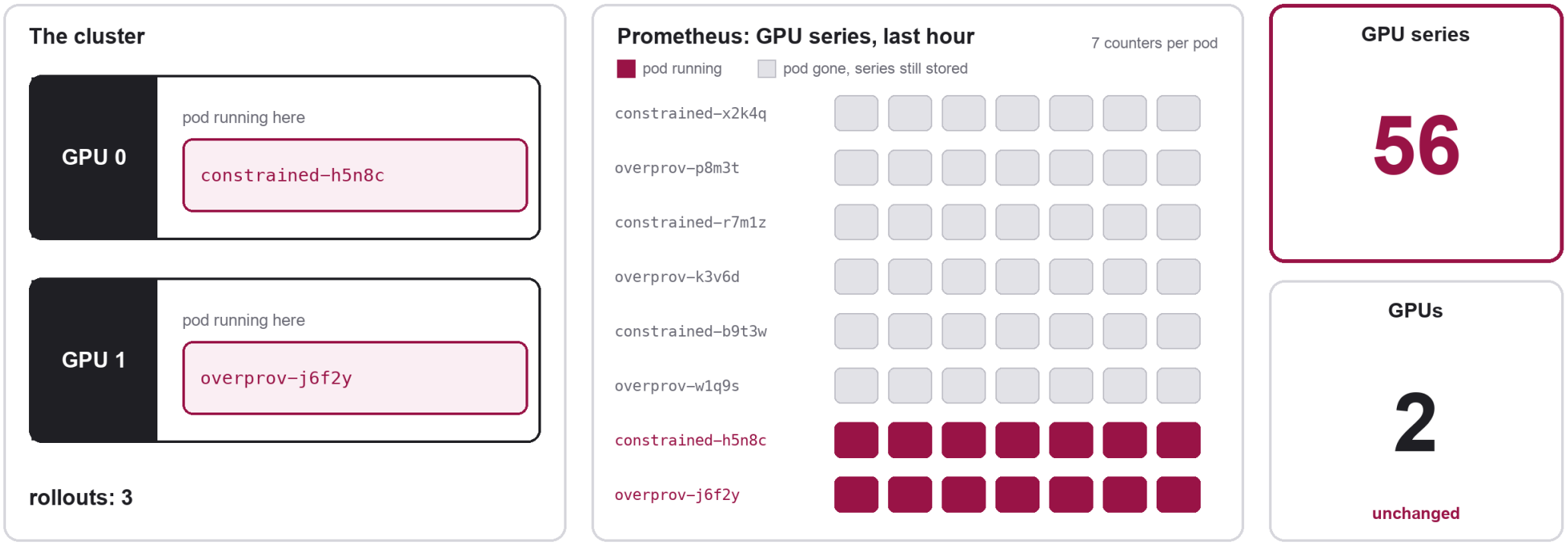
Two traps we hit building the demo.

KnativeCon NORTH AMERICA

2 of 2

Three rollouts later: same 2 GPUs, 56 series.

illustration



Test cluster: vLLM v0.30.0, facebook/opt-125m, one A10G per pod. Cardinality: 7 DCGM counters x 2 GPUs, measured 13 Oct 2026.

Three lines of config, one join key

Observability Day
NORTH AMERICA

Each signal carries the pod name, so one dashboard can line them up.

KnativeCon
NORTH AMERICA

dcgm-exporter

GPU metrics → pod

```
--kubernetes=true
```

Fluent Bit

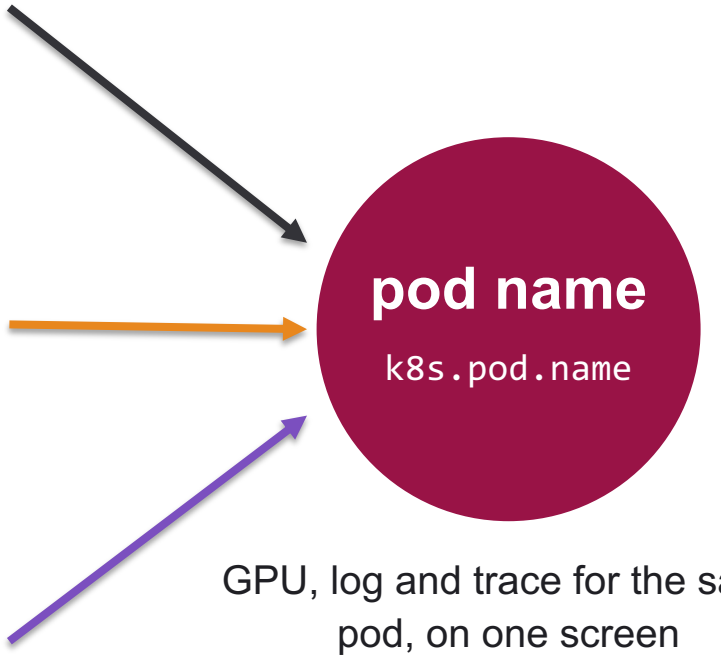
vLLM log → KV cache %

```
string.match(line,
"KV cache usage:%s*([%d%.]+)%%")
```

OTel Collector

traces → pod

```
k8sattributes:
  extract:
    metadata: [k8s.pod.name]
```



GPU, log and trace for the same pod, on one screen

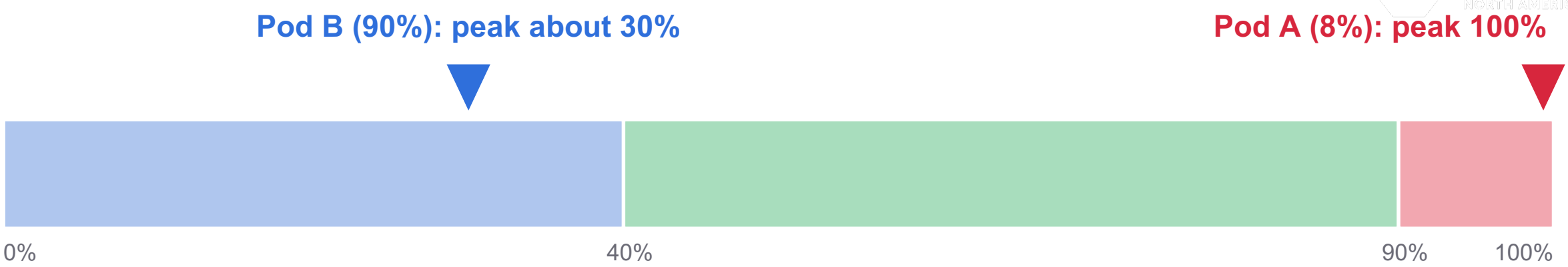
Same pod, three label names: exported_pod (DCGM in Prometheus), pod_name (Fluent Bit), k8s.pod.name (OTel). vLLM also exports KV cache % on /metrics.

Size it from peak KV cache

Observability Day
NORTH AMERICA

Peak KV cache usage, under your own traffic

KnativeCon
NORTH AMERICA



Below 40%: oversized
A smaller GPU, or more replicas per GPU.

40 to 90%: target range
Room for a burst.

Above 90%: too small
Little room for a burst.

1 **Measure**
peak KV cache under your own traffic

2 **Change**
`--gpu-memory-utilization`

3 **Measure again**
peak moves into range, TTFT steady

Everything we showed is in one repo

Observability Day NORTH AMERICA

Deploy it, then measure your own traffic.

KnativeCon NORTH AMERICA

Terraform: an EKS Auto Mode cluster with GPU nodes

dcgm-exporter in Kubernetes mode: GPU metrics per pod

Fluent Bit: a short Lua filter turns the vLLM log into metrics

OpenTelemetry Collector: traces with the pod on every span

All configs, deployable

`github.com/danilosilvase/gpu-observability-kubecon-na-2026`

Observability Day
NORTH AMERICA

Questions?

Danilo Silva

Senior Delivery Consultant, AWS

linkedin.com/in/danilosilvase

Henrique Santana

Principal Cloud Engineer, AWS

linkedin.com/in/henriquessantana

Inference Is a Black Box: How We Built GPU-Aware Observability with Fluent Bit and OpenTelemetry

Rate this session