

Journey from **Fluent Bit**, **Fluentd** and **Prometheus** to

The logo for OpenTelemetry, featuring a stylized telescope or microscope shape composed of blue and yellow segments. The word "OpenTelemetry" is written in blue, with "Open" in yellow, and "Collector" is written in yellow below it.

OpenTelemetry Collector

Marcin "Perk" Stożek



Collectors Zoo



Collectors Zoo



telegraf



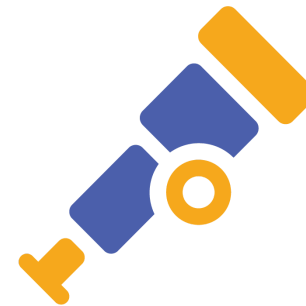
fluentbit



Prometheus



fluentd



OpenTelemetry

Collectors Zoo



OpenTelemetry

where it all started?

de facto k8s
telemetry data
collection standards
in early 2023

logs



metrics



Prometheus

traces



logs



a lot of plugins

community, documentation

written in Ruby

logs



logs



Exhibit 1 - [kubeclient gem](#)

used by more than 1k projects

1 (one) active maintainer

logs



Exhibit 2 - multithreading vs HPA

1 worker = 1 thread

`resources.requests.cpu = 2000`

`autoscaling.targetCPUUtilizationPercentage = 50`

logs



metrics



Prometheus

traces



logs



crazy fast

great memory usage

written in C

logs



logs



metrics



Prometheus

traces



metrics



Prometheus

community, documentation

memory usage

database, not a forwarder *

logs



metrics



Prometheus

traces



traces

?

traces



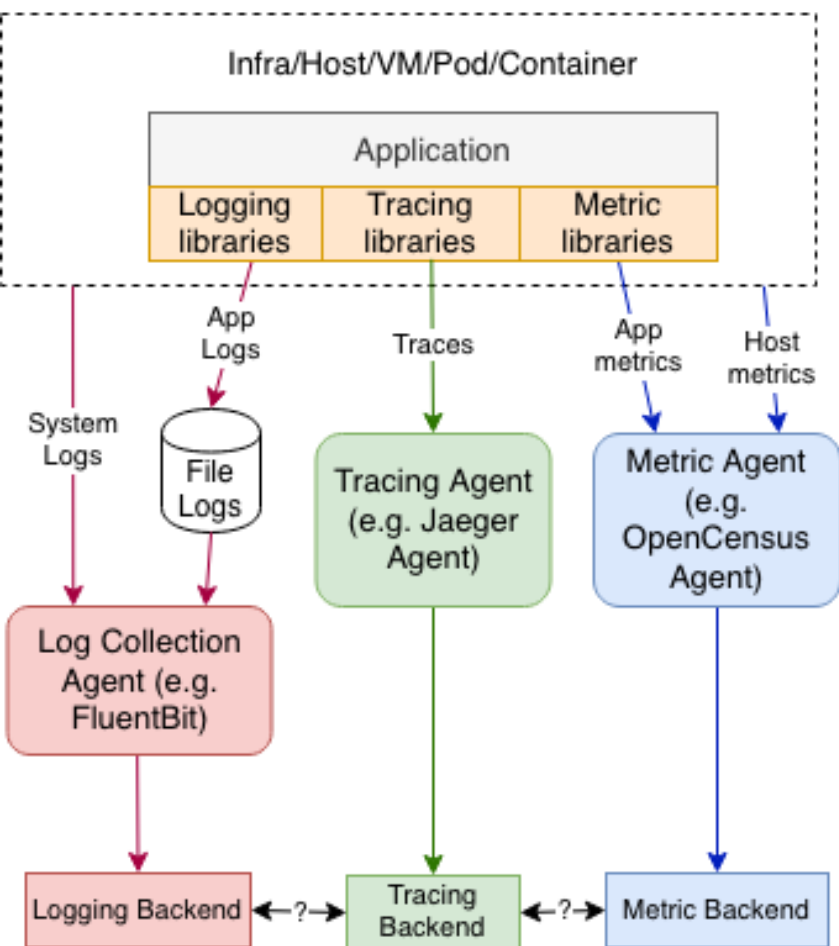
why?



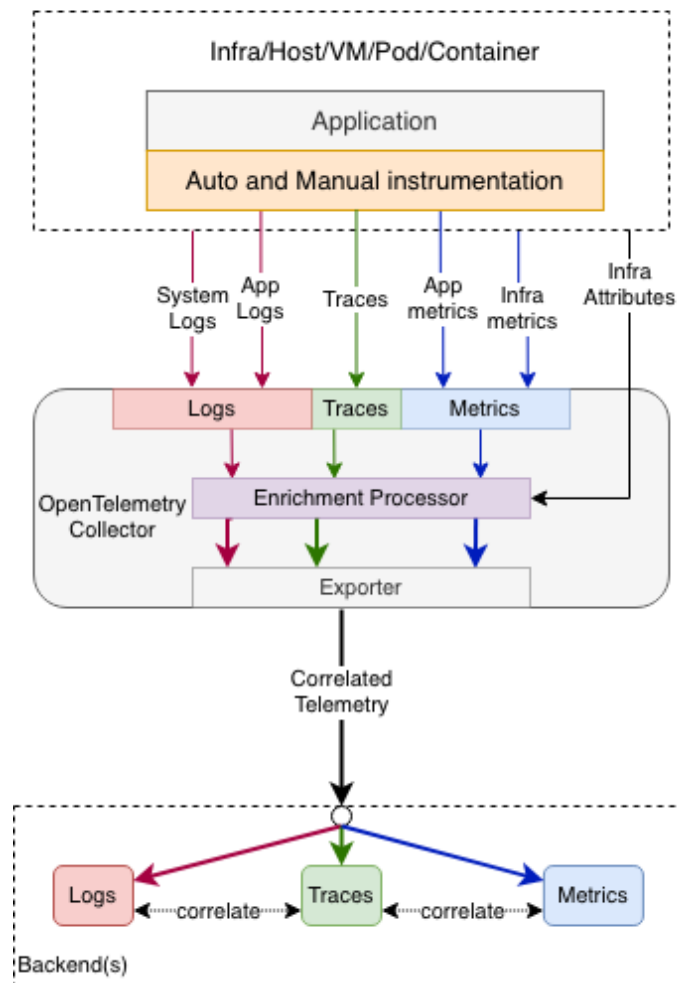
OpenTelemetry

101

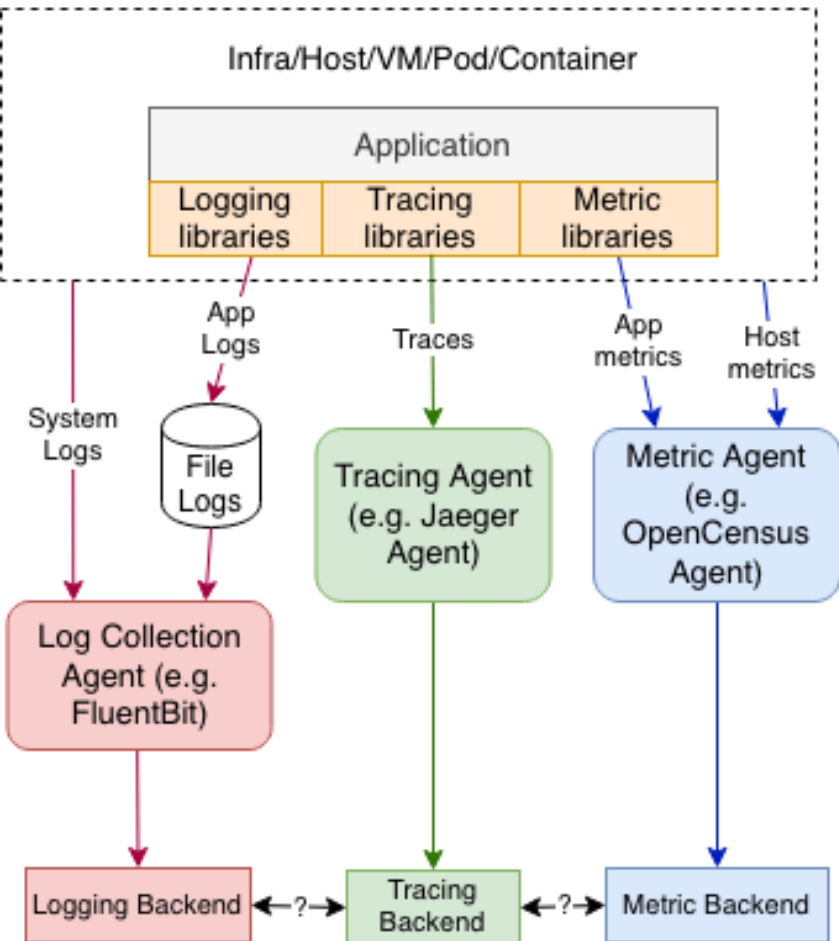
Separate Collection



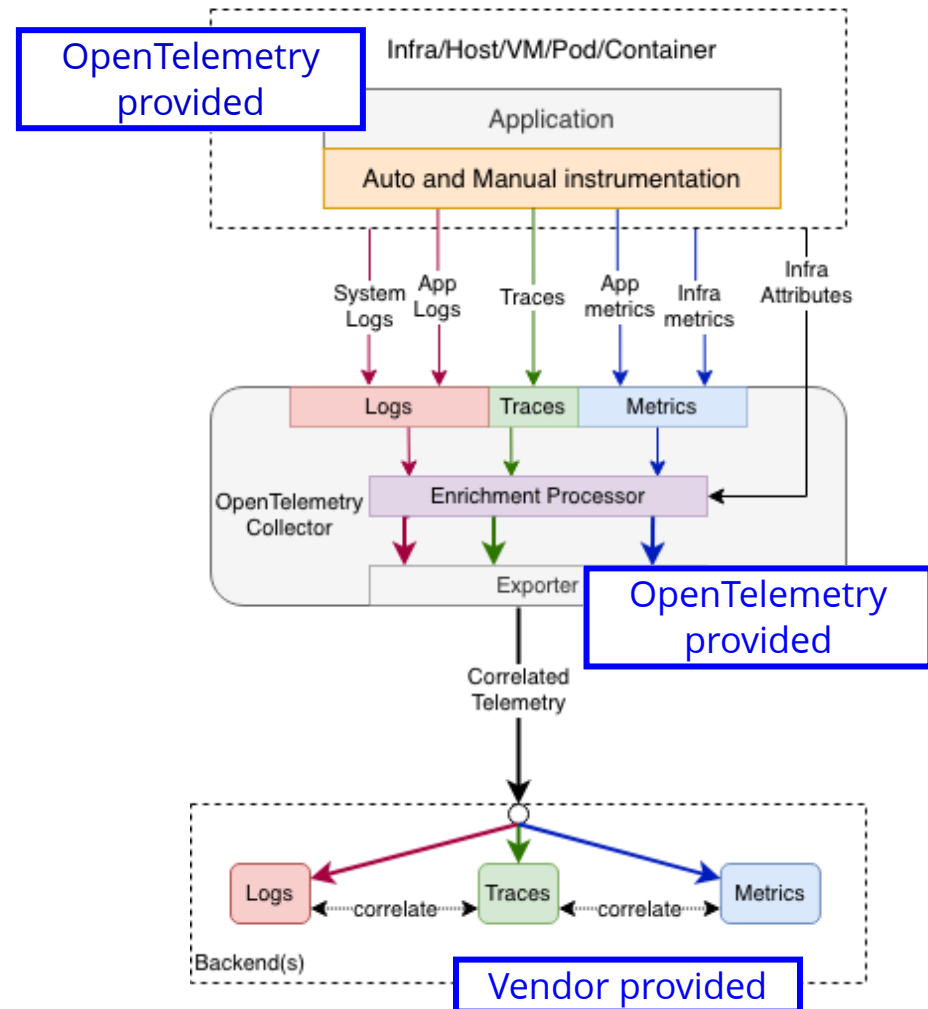
OpenTelemetry Collection



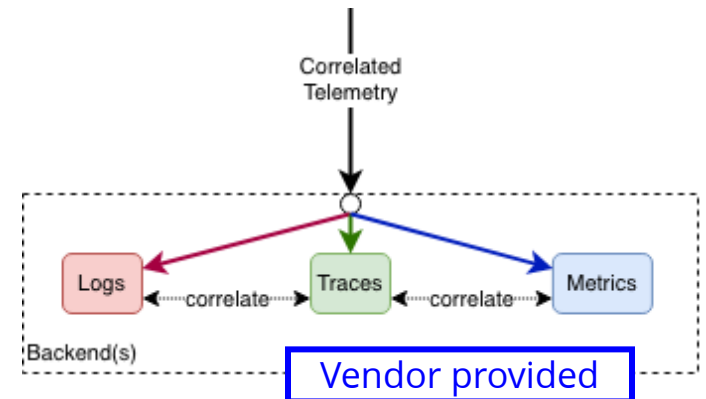
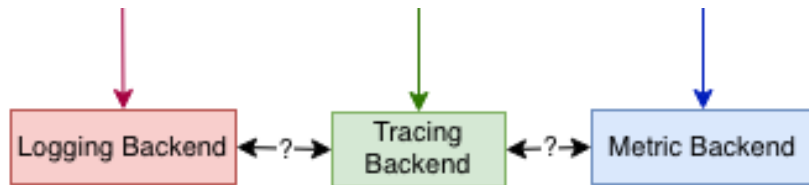
Separate Collection



OpenTelemetry Collection



No OpenTelemetry backend



Vendor support



dynatrace



Uptrace



elastic



sumo logic



Aspecto



DATADOG



Grafana

splunk® >



honeycomb.io



Lightstep

from ServiceNow

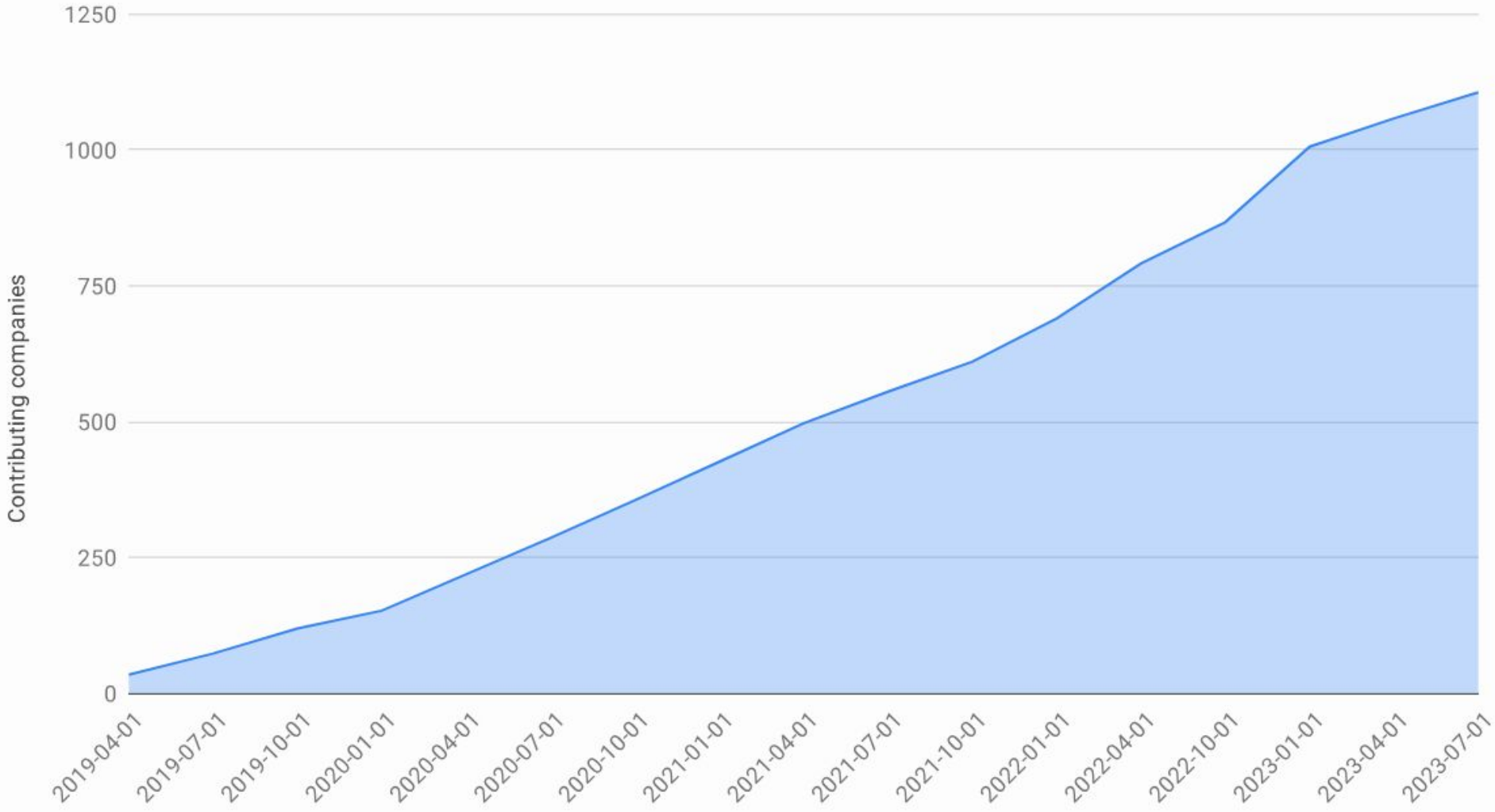


SENTRY
SOFTWARE



APPDYNAMICS

Cumulative number of companies contributing by quarter Q2 2019-Q2 2023



OpenTelemetry

Specification

Language SDKs

Collector

OpenTelemetry

Specification

Language SDKs

Collector

Specification

Guidelines - cross language requirements and expectations for all implementations

Semantic **conventions**

API, SDK

OTLP

<https://github.com/open-telemetry/opentelemetry-specification>

Specification

Guidelines - cross language requirements and expectations for all implementations

Semantic: **Community driven
industry standard**

API, SDK

OTLP

OpenTelemetry

Specification

Language SDKs

Collector

OpenTelemetry

Specification

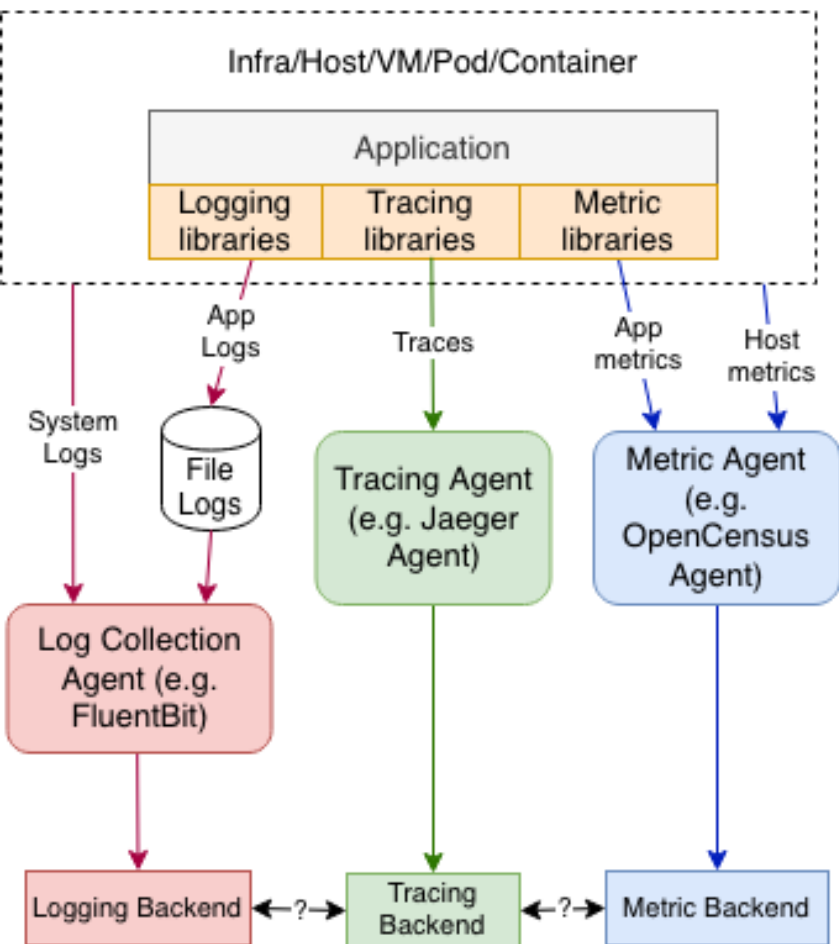
Language SDKs

Collector

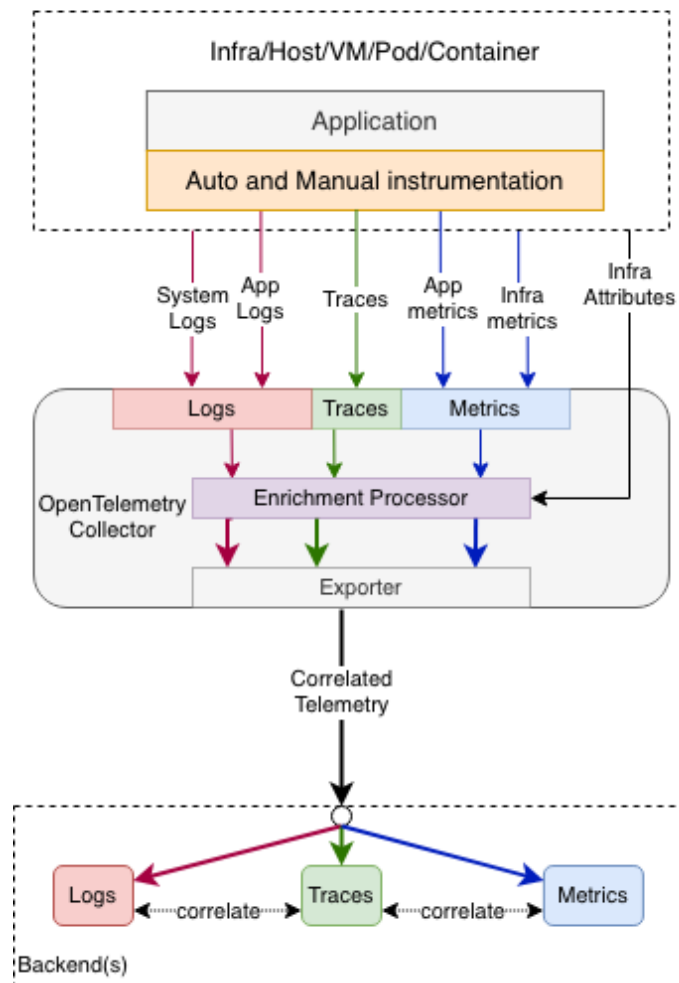
Language SDKs



Separate Collection

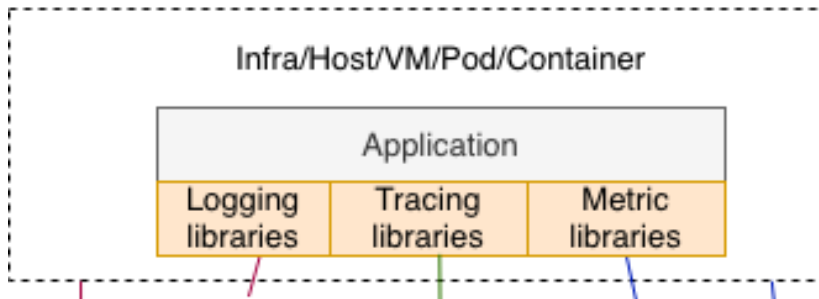


OpenTelemetry Collection

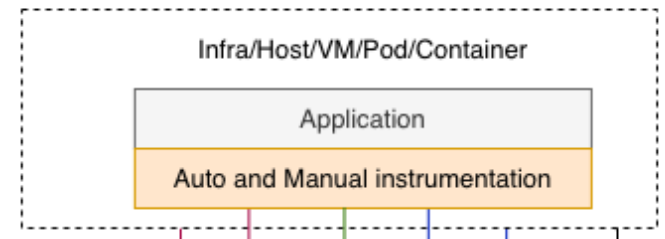


Instrumentation

Separate Collection



OpenTelemetry Collection



Language SDKs

Traces

Feature	Optional	Go	Java	JS	Python	Ruby	Erlang	PHP	Rust	C++	.NET	Swift
TracerProvider												
Create TracerProvider		+	+	+	+	+	+	+	+	+	+	+
Get a Tracer		+	+	+	+	+	+	+	+	+	+	+
Get a Tracer with schema_url		+	+							+		
Associate Tracer with InstrumentationScope												
Safe for concurrent calls		+	+	+	+	+	+	+	+	+	+	+
Shutdown (SDK only required)		+	+	+	+	+	+	+	+	+	+	+
ForceFlush (SDK only required)		+	+	-	+	+	+	+	+	+	+	+
Trace / Context interaction												
Get active Span		N/A	+	+	+	+	+	+	+	+	+	+
Set active Span		N/A	+	+	+	+	+	+	+	+	+	+
Tracer												
Create a new Span		+	+	+	+	+	+	+	+	+	+	+
Documentation defines adding attributes at span creation as preferred											+	
Get active Span		N/A	+	+	+	+	+	+	+	+	+	+

OpenTelemetry

Specification

Language SDKs

Collector

OpenTelemetry

Specification

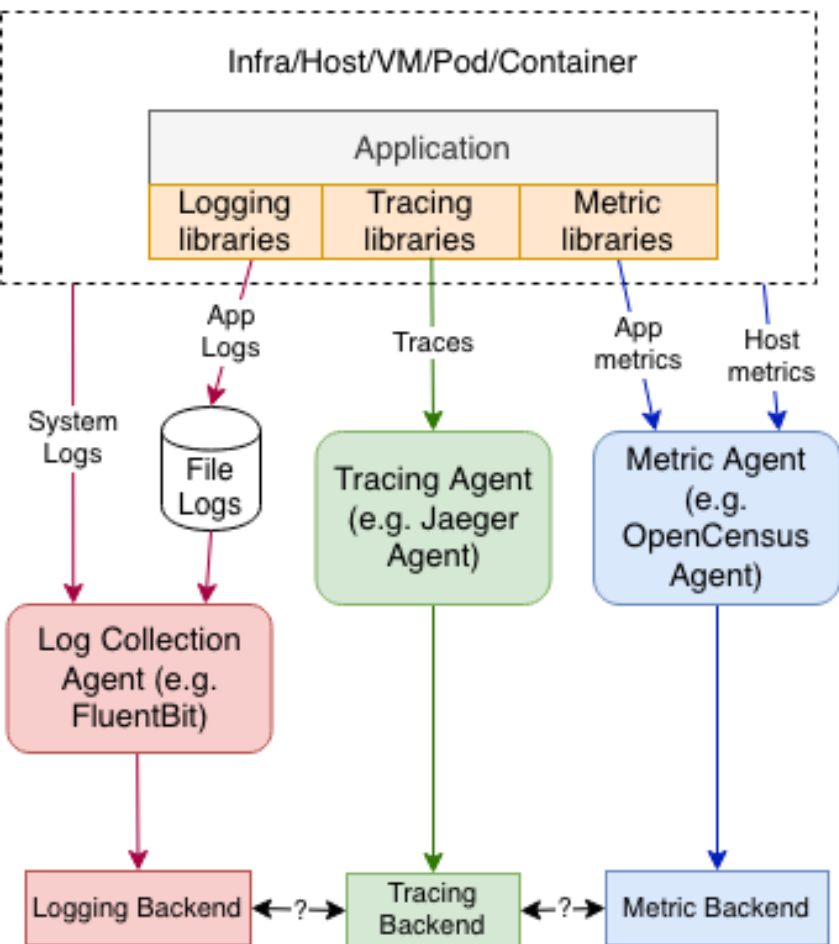
Language SDKs

Collector

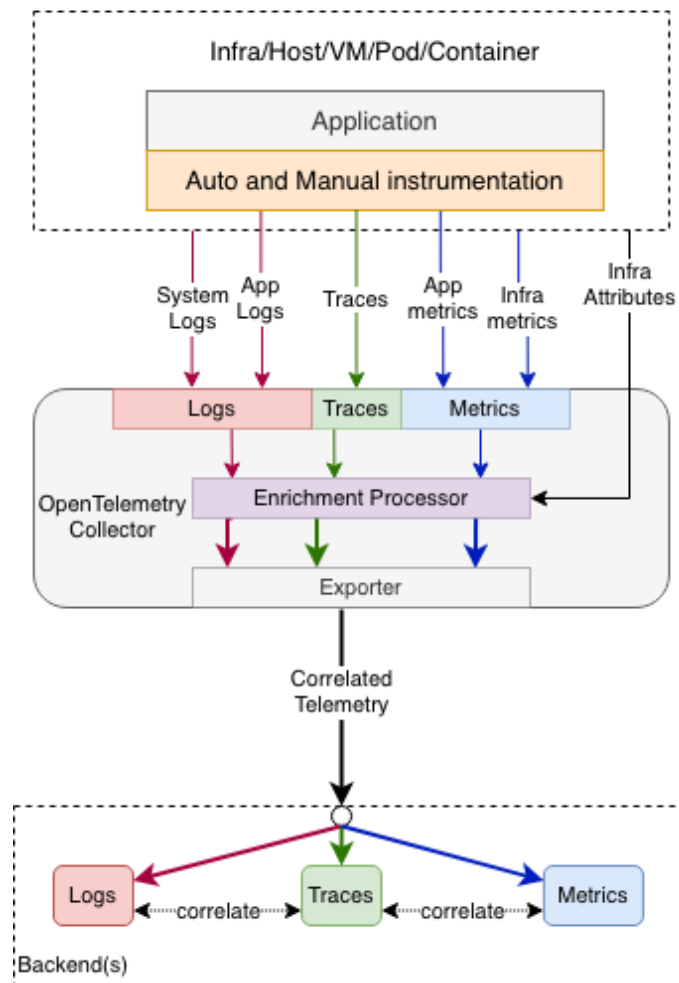
OpenTelemetry Collector

101

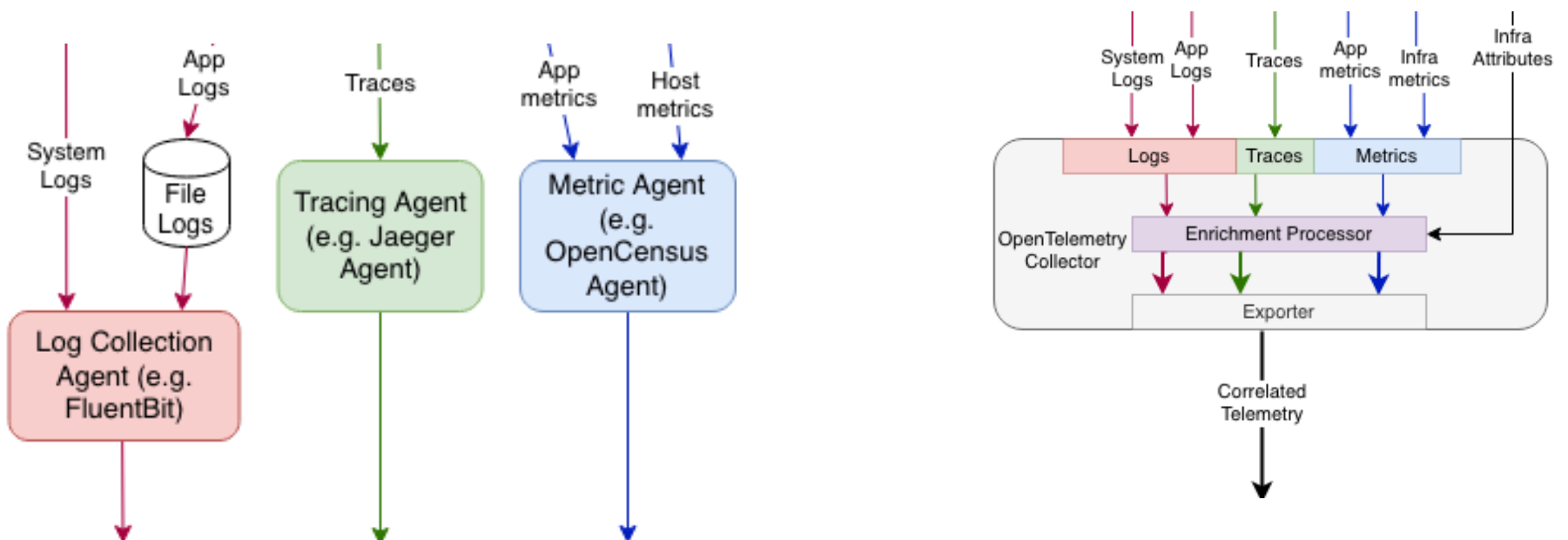
Separate Collection



OpenTelemetry Collection



Collection



OpenTelemetry Collector

OpenTelemetry Collector

Receivers

OpenTelemetry Collector

Receivers



Processors

OpenTelemetry Collector

Receivers



Processors



Exporters

OpenTelemetry Collector

Receivers



Processors

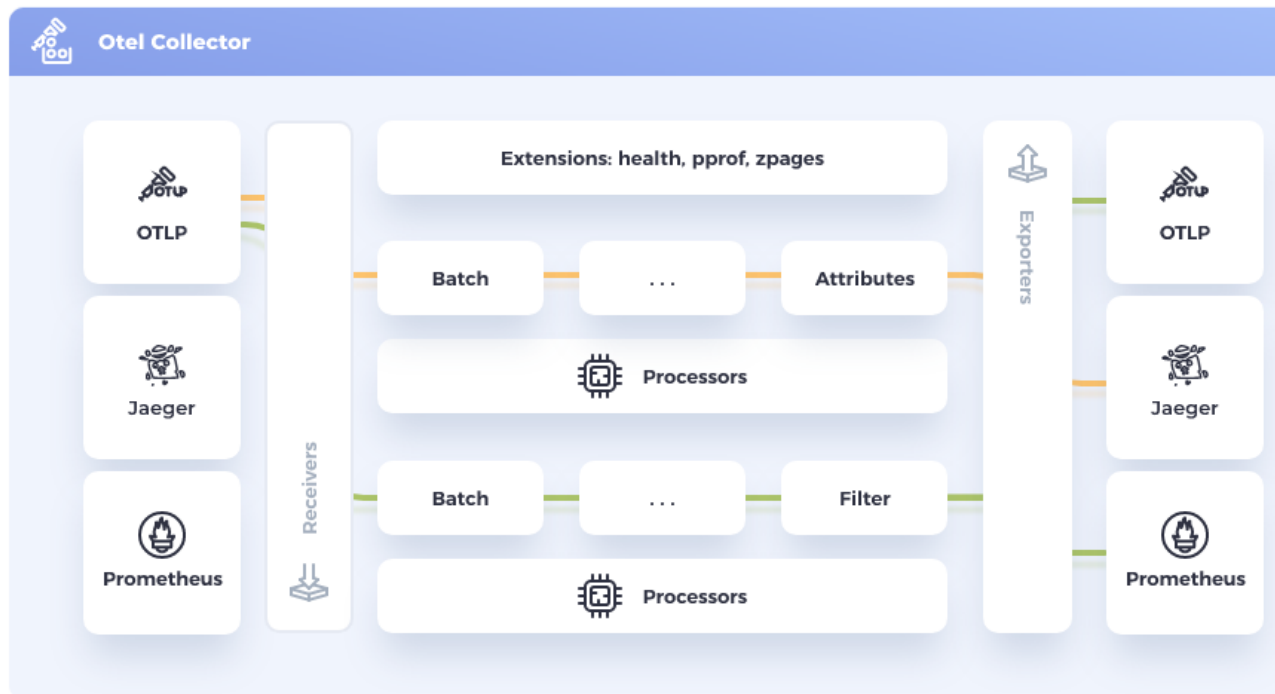


Exporters



Pipelines

OpenTelemetry Collector aka **otelcol**



OTEL COLLECTOR

Host Metrics Receiver

Host Metrics Receiver

Status	
Stability	beta
Supported pipeline types	metrics
Distributions	core , contrib , observiq , splunk , sumo

The Host Metrics receiver generates metrics about the host system scraped from various sources. This is intended to be used when the collector is deployed as an agent.

Getting Started

The collection interval, root path, and the categories of metrics to be scraped can be configured:

```
hostmetrics:
  collection_interval: <duration> # default = 1m
  root_path: <string>
  scrapers:
    <scraper1>:
    <scraper2>:
    ...
```



The available scrapers are:

Scraper	Supported OSs	Description
cpu	All except Mac ^[1]	CPU utilization metrics
disk	All except Mac ^[1]	Disk I/O metrics
load	All	CPU load metrics
filesystem	All	File System utilization metrics

Resource Detection Processor

Resource Detection Processor

Status	
Stability	beta : traces, metrics, logs
Distributions	contrib , observiq , splunk , sumo

The resource detection processor can be used to detect resource information from the host, in a format that conforms to the [OpenTelemetry resource semantic conventions](#), and append or override the resource value in telemetry data with this information.

Supported detectors

Environment Variable

Reads resource information from the `OTEL_RESOURCE_ATTRIBUTES` environment variable. This is expected to be in the format `<key1>=<value1>,<key2>=<value2>,...`, the details of which are currently pending confirmation in the OpenTelemetry specification.

Example:

```
processors:
  resourcedetection/env:
    detectors: [env]
    timeout: 2s
    override: false
```



System metadata

Note: use the Docker detector (see below) if running the Collector as a Docker container.

Queries the host machine to retrieve the following resource attributes:

```
* host.name
* host.id
* os.type
```



OTLP Exporter

OTLP gRPC Exporter

Status	
Stability	traces stable
	metrics stable
	logs beta
Supported pipeline types	traces, metrics, logs
Distributions	core , contrib

Export data via gRPC using [OTLP](#) format. By default, this exporter requires TLS and offers queued retry capabilities.

Getting Started

The following settings are required:

- `endpoint` (no default): host:port to which the exporter is going to send OTLP trace data, using the gRPC protocol. The valid syntax is described [here](#). If a scheme of `https` is used then client transport security is enabled and overrides the `insecure` setting.
- `tls`: see [TLS Configuration Settings](#) for the full set of available options.

Example:

```
exporters:
  otlp:
    endpoint: otelcol2:4317
    tls:
      cert_file: file.cert
      key_file: file.key
  otlp/2:
    endpoint: otelcol2:4317
    tls:
      insecure: true
```



Put it all together

```
1 receivers:
2   hostmetrics:
3     scrapers:
4       memory:
5
6 processors:
7   resourcedetection/detect-host-name:
8     detectors:
9     - system
10    system:
11      hostname_sources:
12      - os
13
14 exporters:
15   otlp:
16     endpoint: otelcol2:4317
17
18 service:
19   pipelines:
20     metrics:
21       receivers:
22       - hostmetrics
23       processors:
24       - resourcedetection/detect-host-name
25       exporters:
26       - otlp
```

Data Pipeline

```
1 receivers:
2   hostmetrics:
3     scrapers:
4       memory:
5
6 processors:
7   resourcedetection/detect-host-name:
8     detectors:
9     - system
10    system:
11      hostname_sources:
12      - os
13
14 exporters:
15   otlp:
16     endpoint: otelcol2:4317
17
18 service:
19   pipelines:
20     metrics:
21       receivers:
22       - hostmetrics
23       processors:
24       - resourcedetection/detect-host-name
25       exporters:
26       - otlp
```

More Data Pipelines

```
1 service:
2   pipelines:
3     metrics:
4       receivers: [hostmetrics]
5       processors: [resourcedetection/detect-host-name]
6       exporters: [otlp]
7
8     metrics/kafka:
9       receivers: [kafka]
10      exporters: [otlp/kafka]
11
12    logs:
13      receivers: [filelog]
14      exporters: [otlp]
15
16    logs/kafka:
17      receivers: [kafka]
18      exporters: [otlp, otlp/kafka]
19
20    traces:
21      receivers: [otlp, kafka]
22      processors: [resourcedetection/detect-host-name]
23      exporters: [otlp, otlp/kafka]
```

OTel Collector distros

OTel Collector **distros**

Core distribution

github.com/open-telemetry/opentelemetry-collector - components

OTel Collector **distros**

Core distribution

github.com/open-telemetry/opentelemetry-collector - components

Contrib distribution

<https://github.com/open-telemetry/opentelemetry-collector-contrib> - components

OTel Collector **distros**

Core distribution

github.com/open-telemetry/opentelemetry-collector - components

Contrib distribution

<https://github.com/open-telemetry/opentelemetry-collector-contrib> - components

Custom distributions

- Grafana Agent
- Sumo Logic
- ADOT (AWS)
- ...

K8s observability

Logs



Metrics



Traces



Metadata



sumo logic

K8s observability

Traces

Metadata

Logs

Metrics

K8s observability

Traces

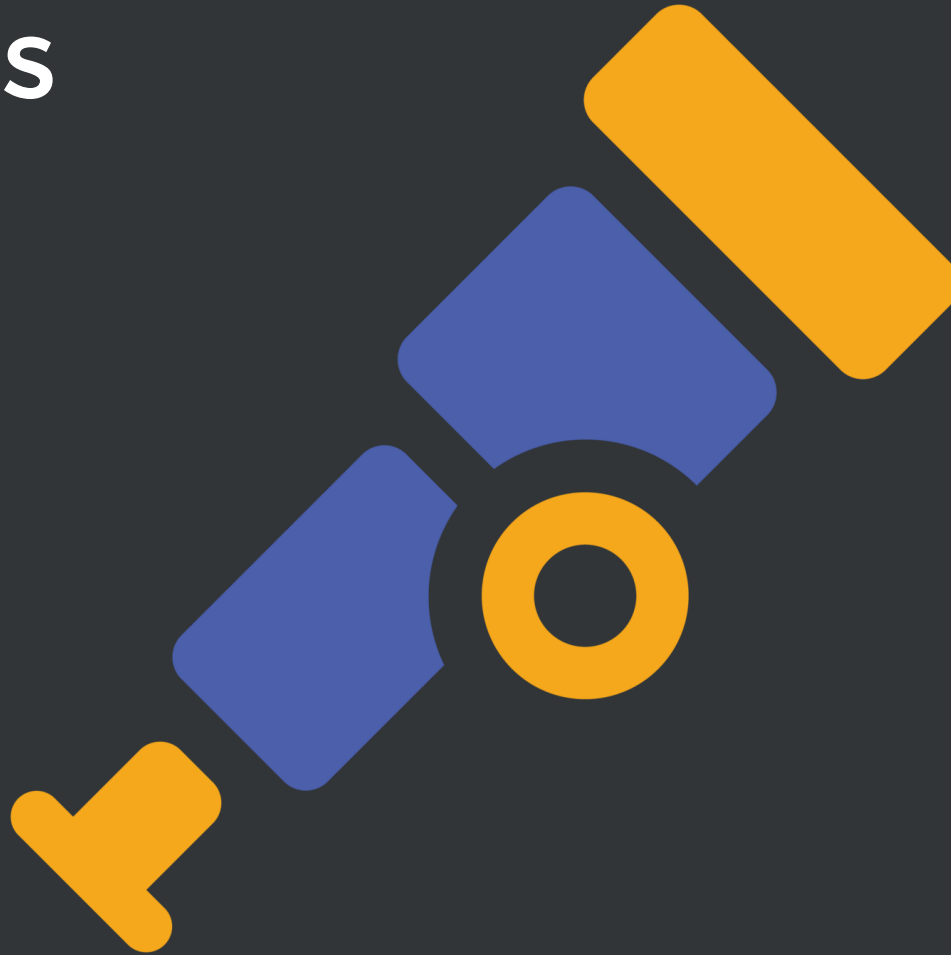
Metadata

Logs

Metrics

Traces

~March 2020





**data
flood**

K8s observability

Traces

Metadata

Logs

Metrics

K8s observability

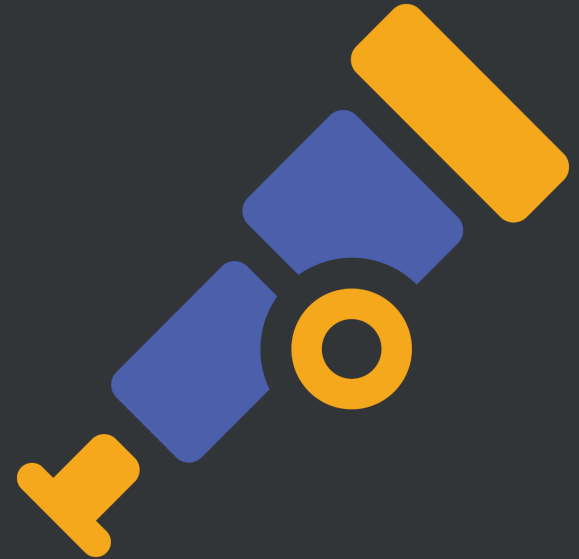
Traces

Metadata

Logs

Metrics

Metadata





Metadata



February 2022



Battle tested

Single threaded

Weak performance

Ruby magic



Metadata



February 2022



K8s Attributes in beta

Battle tested

Go-lang performance

Single threaded

Removed backpressure
from Prometheus'
remote-write





Weak performance

Ruby magic

Lowered Prometheus'
memory

Metadata: CPU sum

CPU usage for otelcol logs

```
#A  _collector= namespace= statefulset=-sumologic-otelcol-logs metric=container_cpu_usage_seconds_total | rate increasing | quantize to 1d using max | sum
```



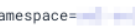

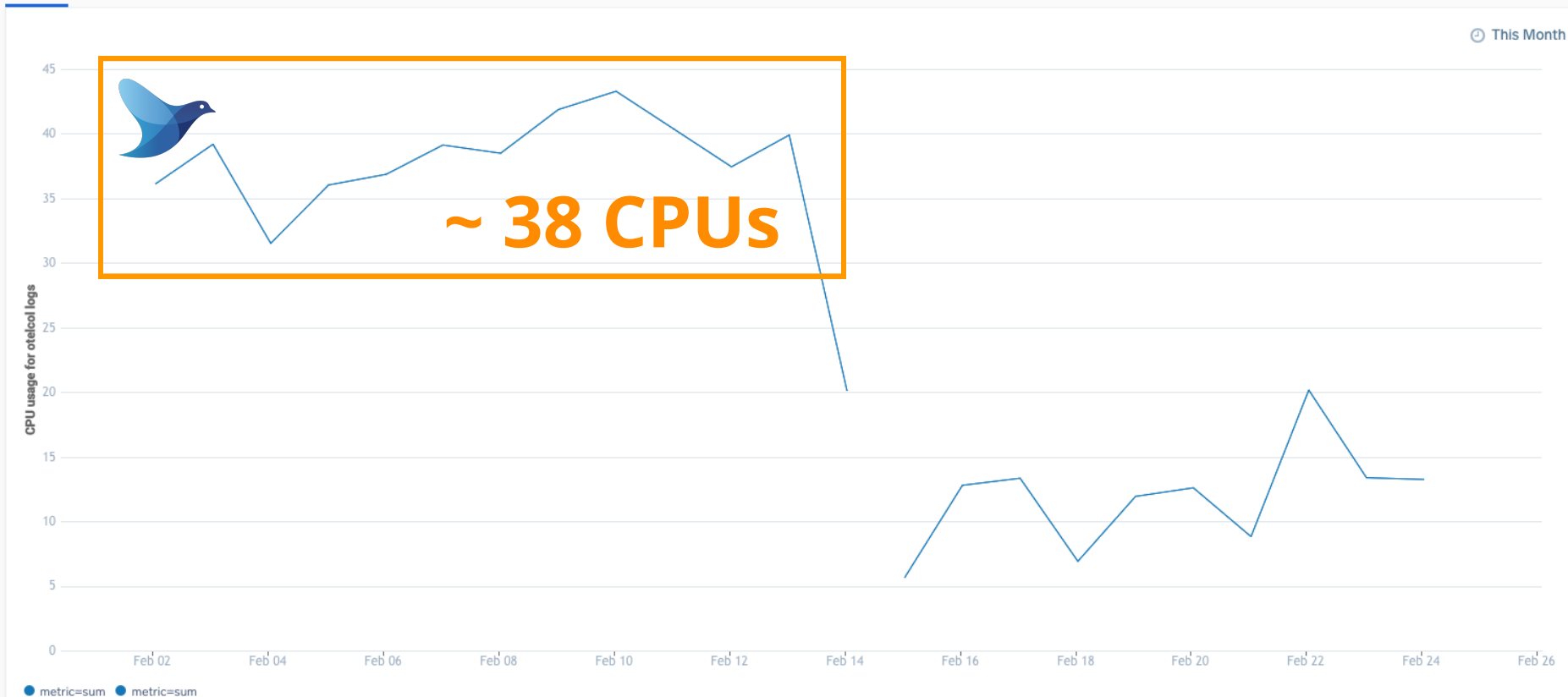
```
#B  _collector= namespace= statefulset=-sumologic-fluentd-logs metric=container_cpu_usage_seconds_total | rate increasing | quantize to 1d using max | sum
```

Chart Preview Table



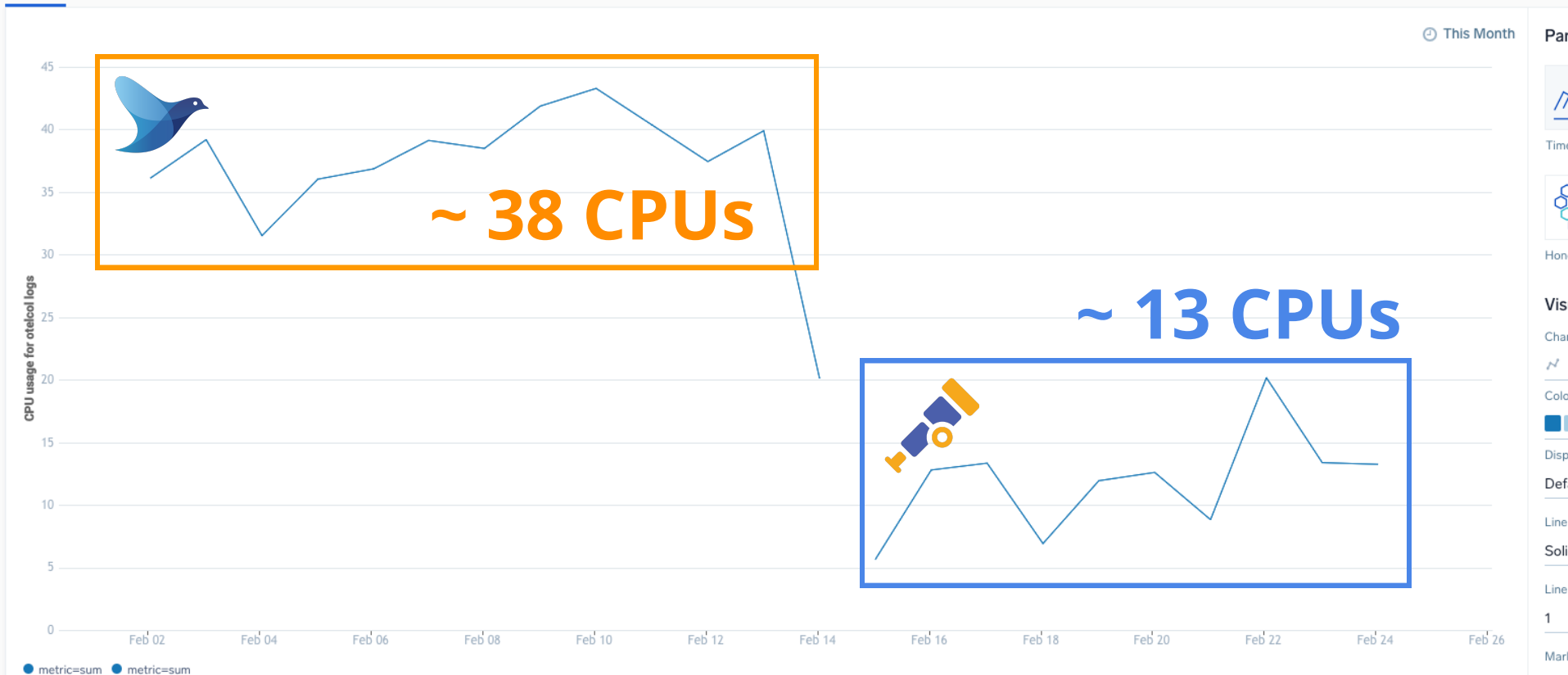
Metadata: CPU sum

CPU usage for otelcol logs

```
#A _collector=... namespace=... statefulset=...-sumologic-otelcol-logs metric=container_cpu_usage_seconds_total | rate increasing | quantize to 1d using max | sum
```

```
#B _collector=... namespace=... statefulset=...-sumologic-fluentd-logs metric=container_cpu_usage_seconds_total | rate increasing | quantize to 1d using max | sum
```

Chart Preview Table

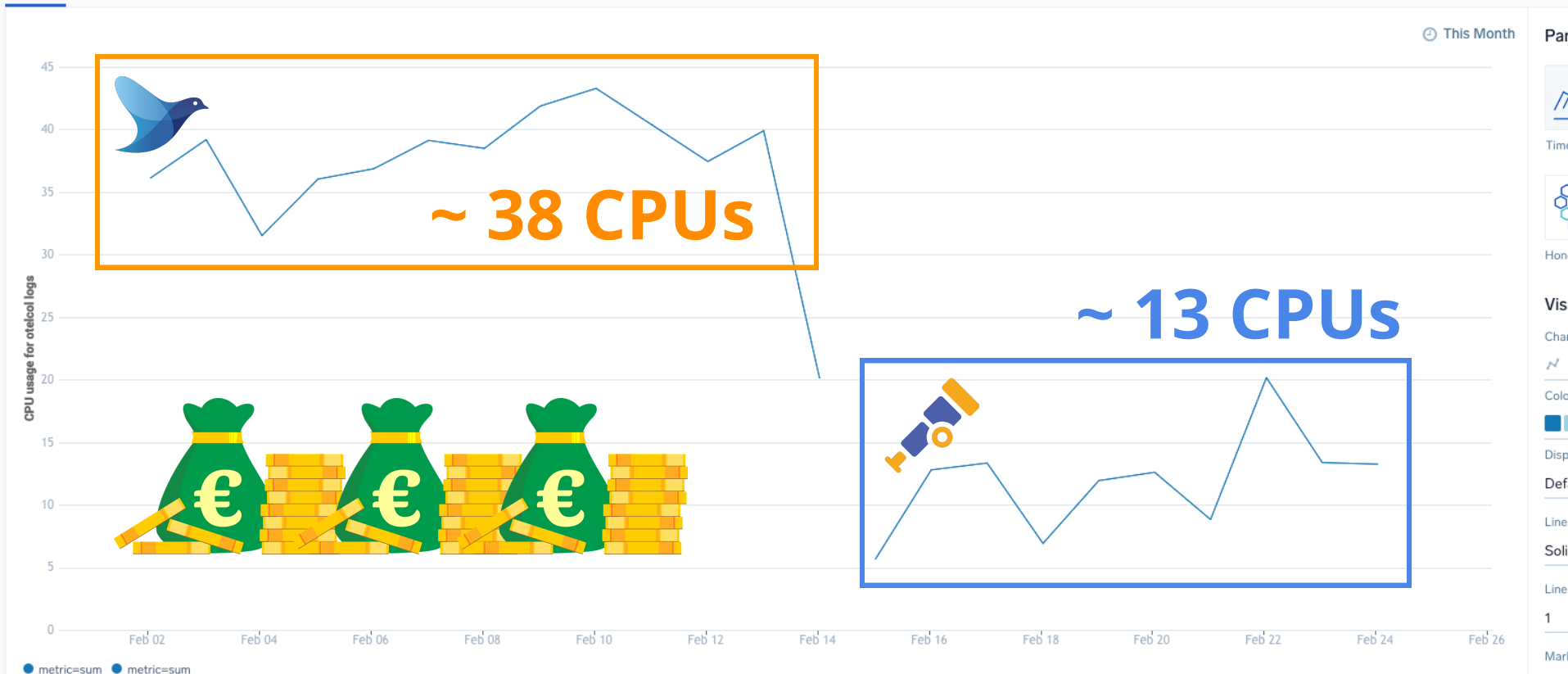


Metadata: CPU sum

CPU usage for otelcol logs

```
#A _collector=... namespace=... statefulset=...-sumologic-otelcol-logs metric=container_cpu_usage_seconds_total | rate increasing | quantize to 1d using max | sum  
#B _collector=... namespace=... statefulset=...-sumologic-fluentd-logs metric=container_cpu_usage_seconds_total | rate increasing | quantize to 1d using max | sum
```

Chart Preview Table



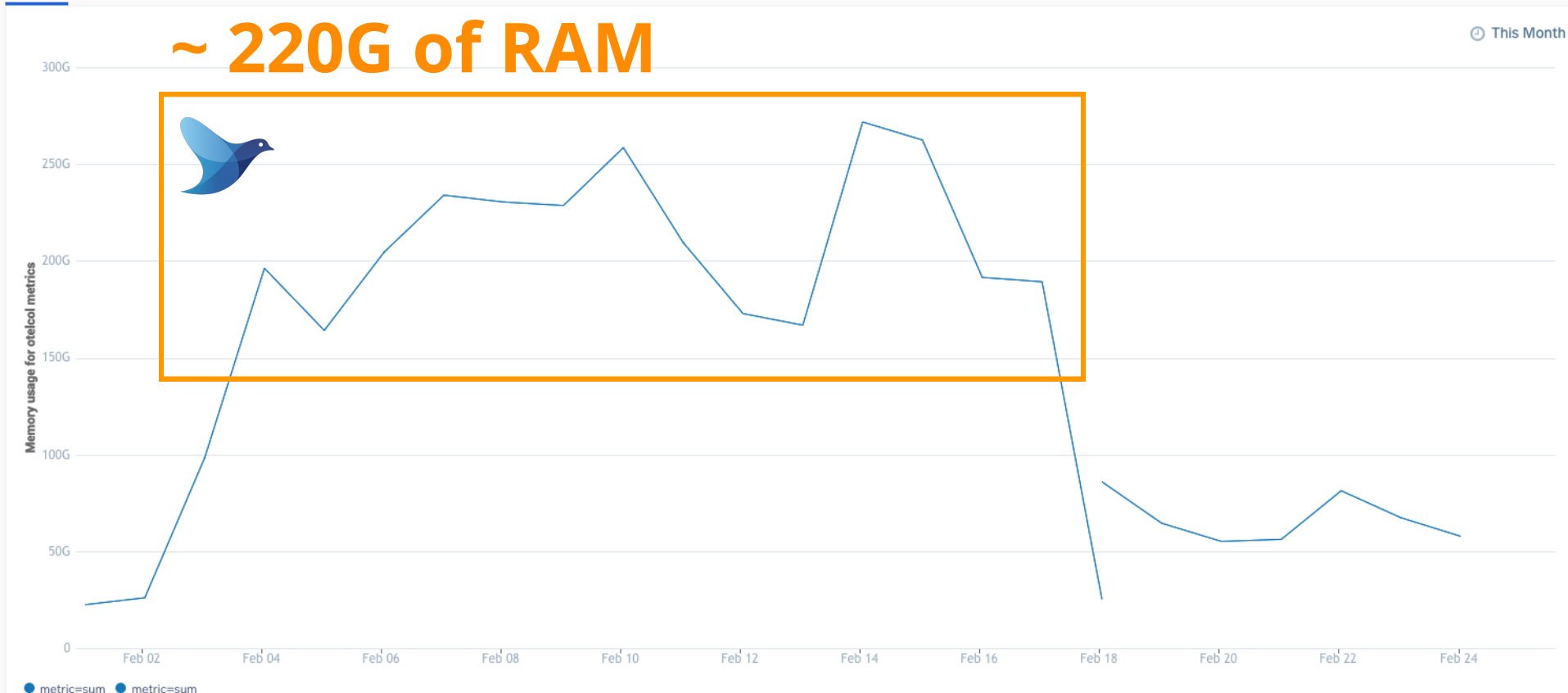
Metadata: memory sum

Memory usage for otelcol metrics [🔗](#) [📄](#)

#A `_collector=[redacted] namespace=[redacted] statefulset=[redacted] sumologic-fluentd-metrics metric=container_memory_working_set_bytes | quantize to 1d | sum`

#B `_collector=[redacted] namespace=[redacted] statefulset=[redacted] sumologic-otelcol-metrics metric=container_memory_working_set_bytes | quantize to 1d | sum`

Chart [Preview Table](#)



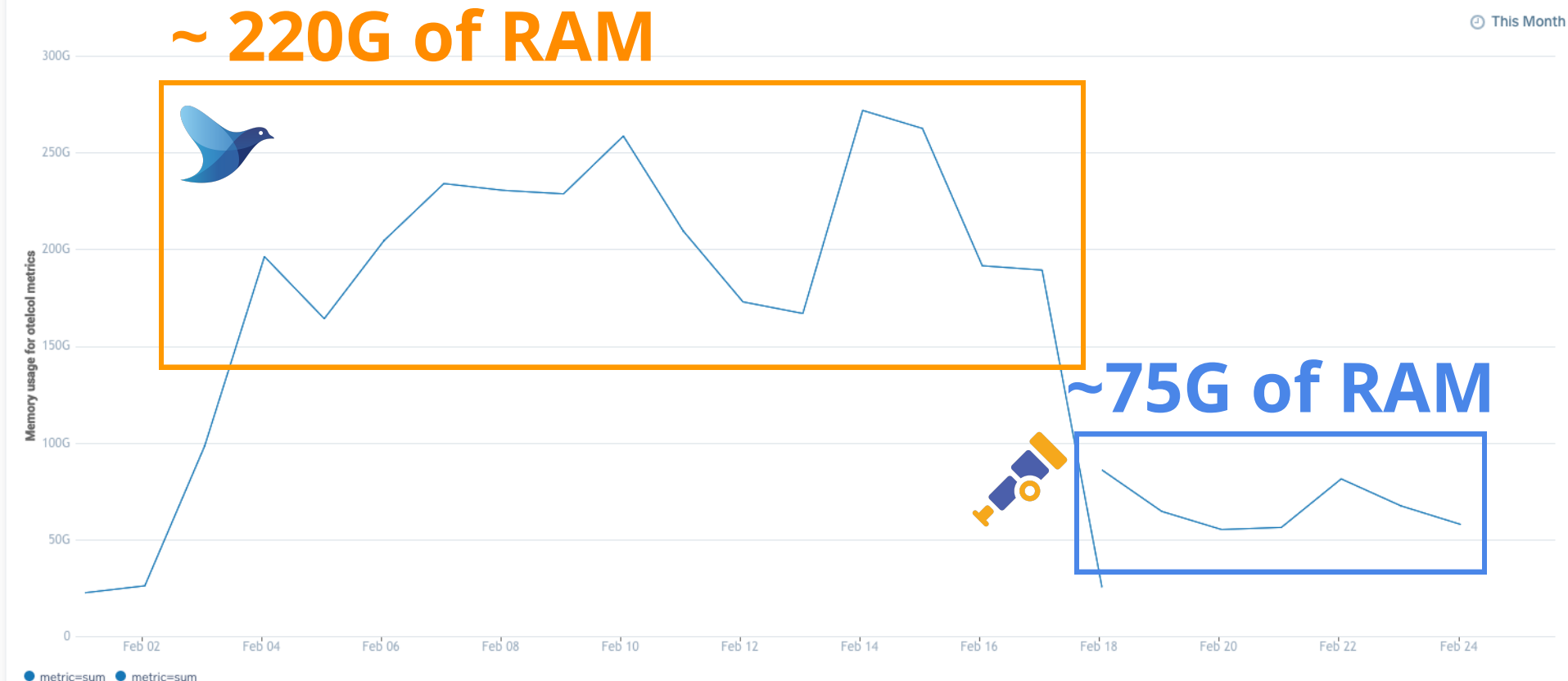
Metadata: memory sum

Memory usage for otelcol metrics [🔗](#) [📄](#)

```
#A _collector=... namespace=... statefulset=...-sumologic-fluentd-metrics metric=container_memory_working_set_bytes | quantize to 1d | sum
```

```
#B _collector=... namespace=... statefulset=...-sumologic-otelcol-metrics metric=container_memory_working_set_bytes | quantize to 1d | sum
```

Chart [Preview Table](#)



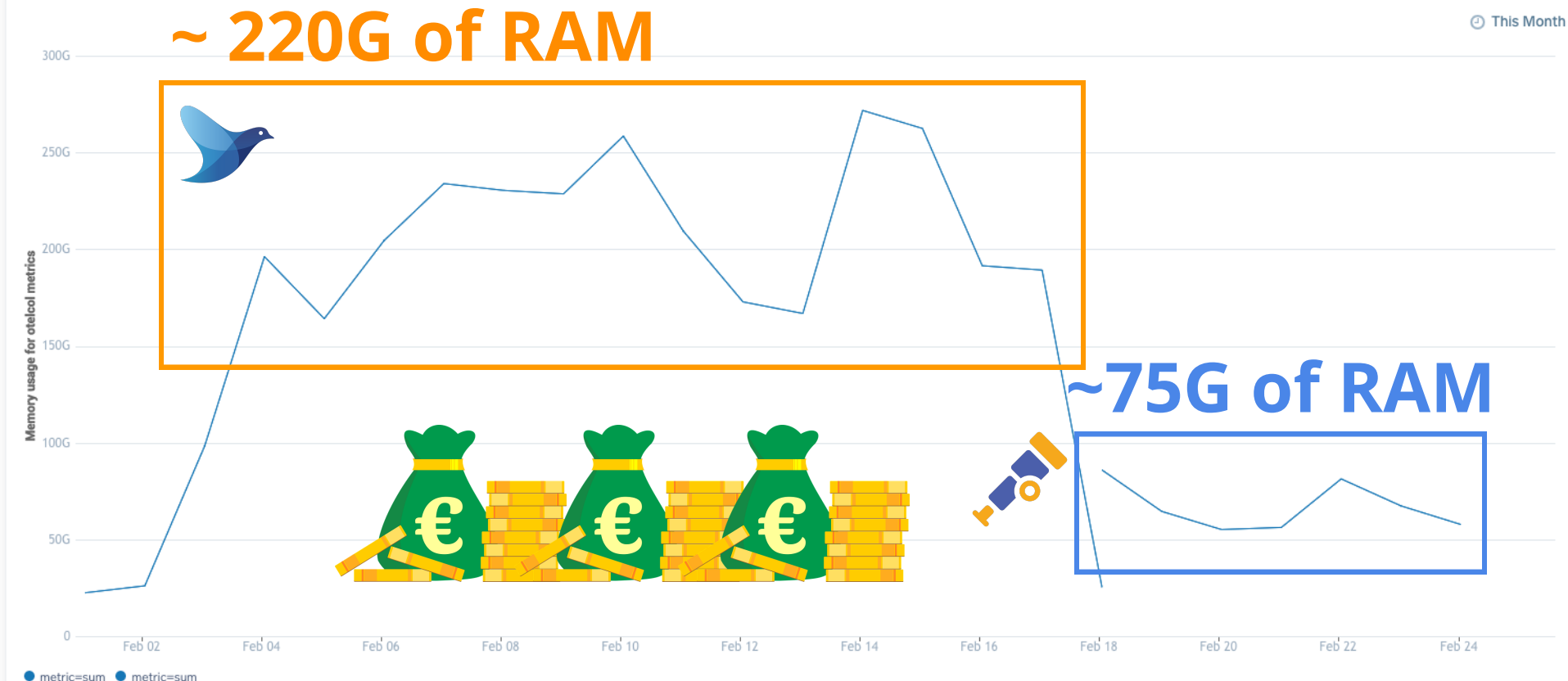
Metadata: memory sum

Memory usage for otelcol metrics [🔗](#) [📄](#)



```
#A _collector=... namespace=... statefulset=...-sumologic-fluentd-metrics metric=container_memory_working_set_bytes | quantize to 1d | sum
```

```
#B _collector=... namespace=... statefulset=...-sumologic-otelcol-metrics metric=container_memory_working_set_bytes | quantize to 1d | sum
```

Chart [Preview Table](#)



... memory after more tweaks?

Memory usage for otelcol metrics  

#A   `_collector=otel-collector namespace = otel-collector statefulset = otel-collector-sumologic-otelcol-metrics metric=container_memory_working_set_bytes | sum`

Chart Time Series



Metadata: instances

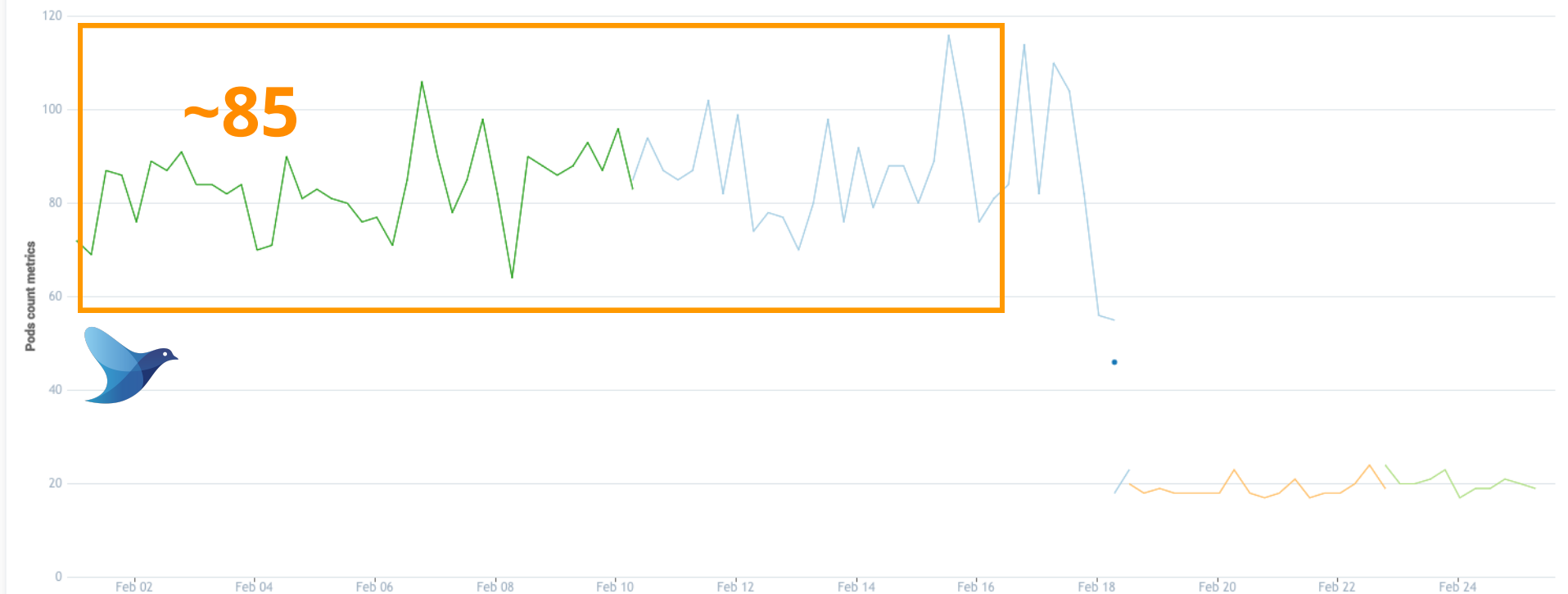
Pods count otelcol metrics  

#A  `_collector= namespace= statefulset= sumologic-fluentd-metrics metric=kube_statefulset_status_replicas | quantize using max`

#B  `_collector= namespace= statefulset= sumologic-otelcol-metrics metric=kube_statefulset_status_replicas | quantize using max`

Chart [Preview Table](#)

 This Month



Metadata: instances

Pods count otelcol metrics  

#A  `_collector=... namespace=... statefulset=...-sumologic-fluentd-metrics metric=kube_statefulset_status_replicas | quantize using max`

#B  `_collector=... namespace=... statefulset=...-sumologic-otelcol-metrics metric=kube_statefulset_status_replicas | quantize using max`

Chart [Preview Table](#)

 This Month



... instances after more tweaks?

Pods count otelcol metrics [🔗](#) [📄](#)

#A

  `_collector=otc-primary-eks namespace = ot-collectors statefulset = ot-collector-sumologic-otelcol-metrics metric=kube_statefulset_status_replicas | quantize using max`

Chart Time Series

7 matching time series [🔗](#)



... instances after more tweaks?

Pods count otelcol metrics [🔗](#) [📄](#)

#A

  `_collector=otelcol-primary-001 namespace = otelcol-001 statefulset = otelcol-001 sumologic-otelcol-metrics metric=kube_statefulset_status_replicas | quantize using max`

Chart Time Series

7 matching time series [🔗](#)



K8s observability

Traces

Metadata

Logs

Metrics

K8s observability

Traces

Metadata

Logs

Metrics

Logs





Logs



June 2022



Great CPU usage

Great memory usage

Hard to debug issues

Didn't support metrics and
traces at the time
(it does now)



Logs



June 2022



Great CPU usage

Great memory usage

Hard to debug issues

Didn't support metrics and traces at the time
(it does now)

Filelog Receiver in beta

Great CPU usage

Reasonable memory usage

No major feature missing

K8s observability

Traces

Metadata

Logs

Metrics

K8s observability

Traces

Metadata

Logs

Metrics

Metrics



Metrics



September 2023



Remote write

Load balancing

Memory usage

Metrics



September 2023



Remote write

Load balancing

Memory usage

Prometheus receiver
in beta

Metric names quirks

Small resource usage

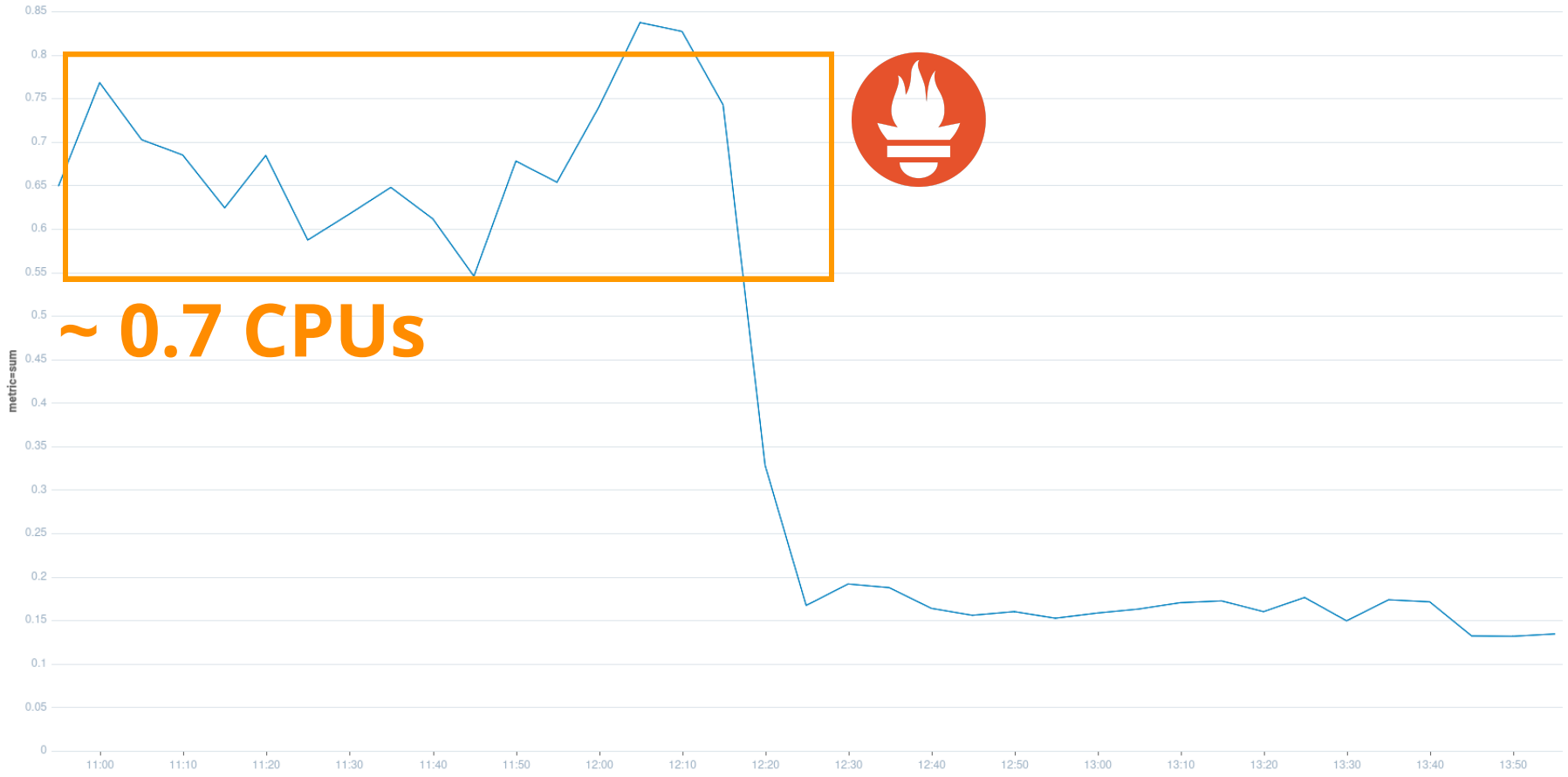
Metrics: CPU sum

Metrics Explorer

```
#A namespace=... metric=container_memory_working_set_bytes (statefulset=...sumologic-metrics-collector OR statefulset=prometheus-...-prometheus*) | sum | quantize 5m  
#B namespace=... metric=container_cpu_usage_seconds_total (statefulset=...sumologic-metrics-collector OR statefulset=prometheus-...-prometheus*) | rate counter | sum | quantize 5m
```

Chart Time Series

1 matching time series



~ 0.7 CPUs



Latest	Min	Max
0.13	0.13	0.84

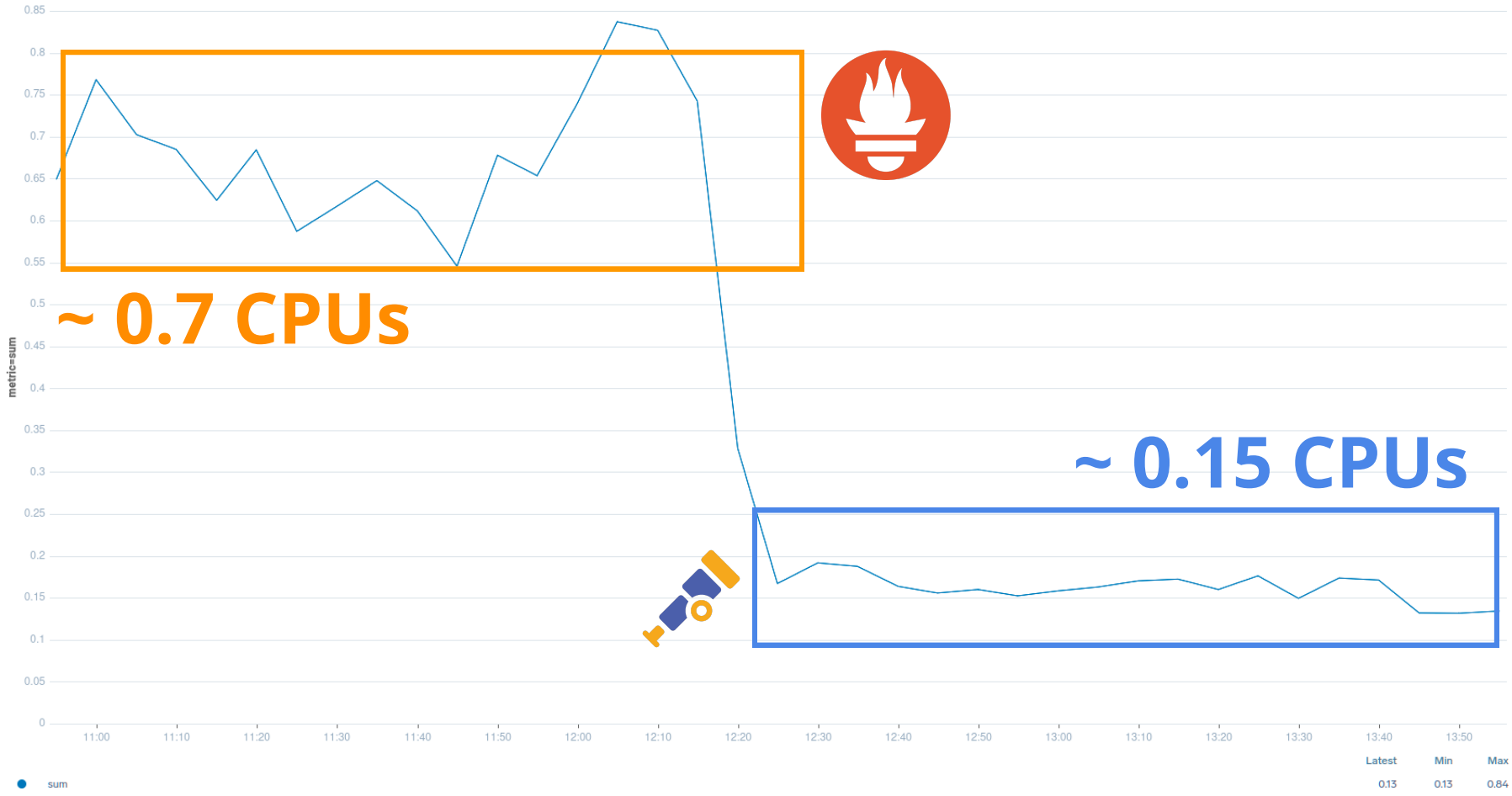
Metrics: CPU sum

Metrics Explorer

```
#A namespace=... metric=container_memory_working_set_bytes (statefulset=...-sumologic-metrics-collector OR statefulset=prometheus-...-prometheus*) | sum | quantize 5m  
#B namespace=... metric=container_cpu_usage_seconds_total (statefulset=...-sumologic-metrics-collector OR statefulset=prometheus-...-prometheus*) | rate counter | sum | quantize 5m
```

Chart Time Series

1 matching time series



Metrics: memory sum

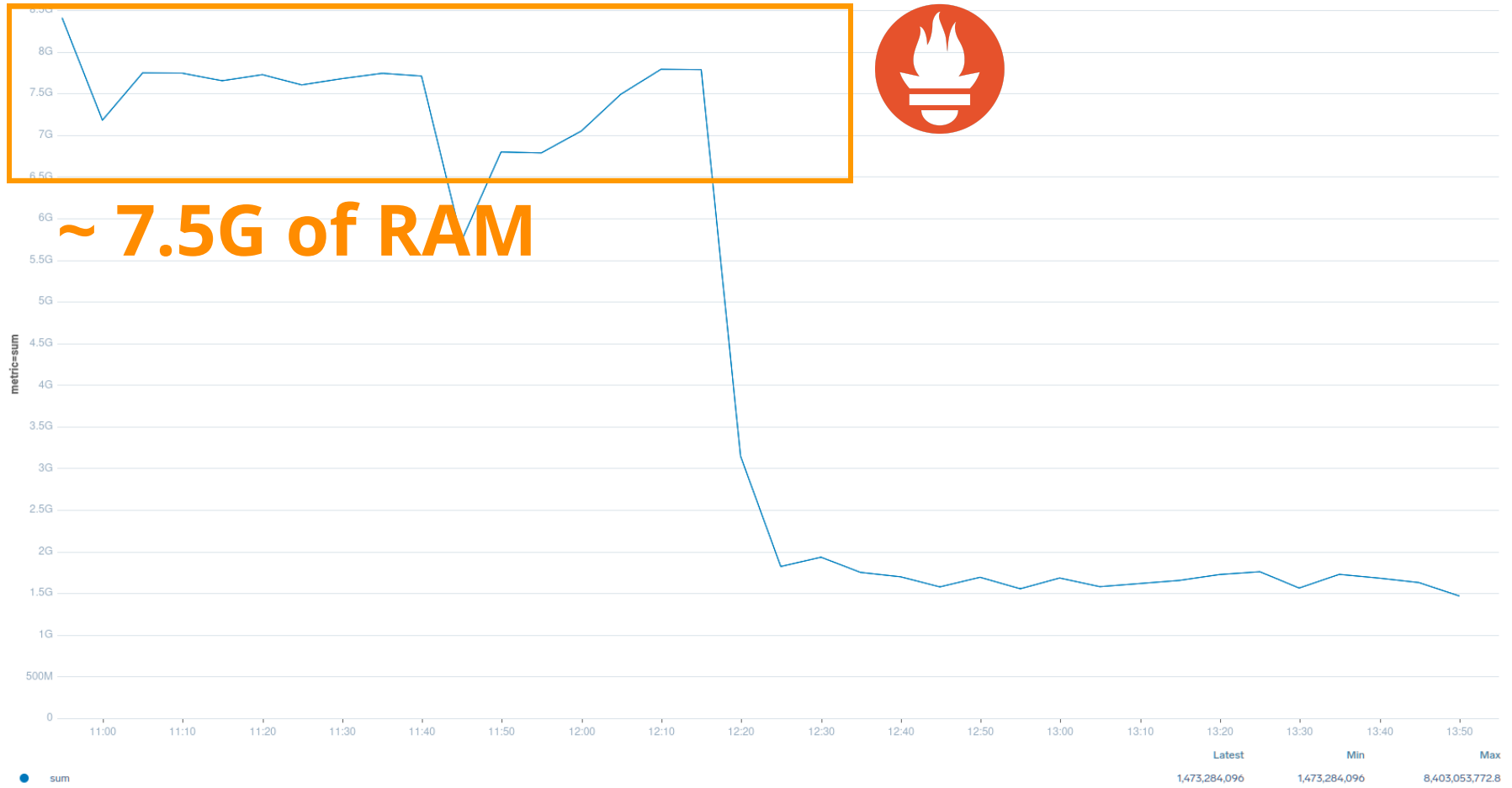
Metrics Explorer  

#A   namespace=[kubernetes](#) metric=container_memory_working_set_bytes (statefulset=[kubernetes-sumologic-metrics-collector](#) OR statefulset=prometheus-[kubernetes](#)-prometheus) | sum | quantize 5m

#B   namespace=[kubernetes](#) metric=container_cpu_usage_seconds_total (statefulset=[kubernetes-sumologic-metrics-collector](#) OR statefulset=prometheus-[kubernetes](#)-prometheus*) | rate counter | sum | quantize 5m



Chart Time Series

1 matching time series 



Metrics: memory sum

Metrics Explorer  

#A   namespace=`longhorn-system` metric=`container_memory_working_set_bytes` statefulset=`longhorn-backup-sumologic-metrics-collector` OR statefulset=`prometheus-longhorn-backup-prometheus` | sum | quantize 5m



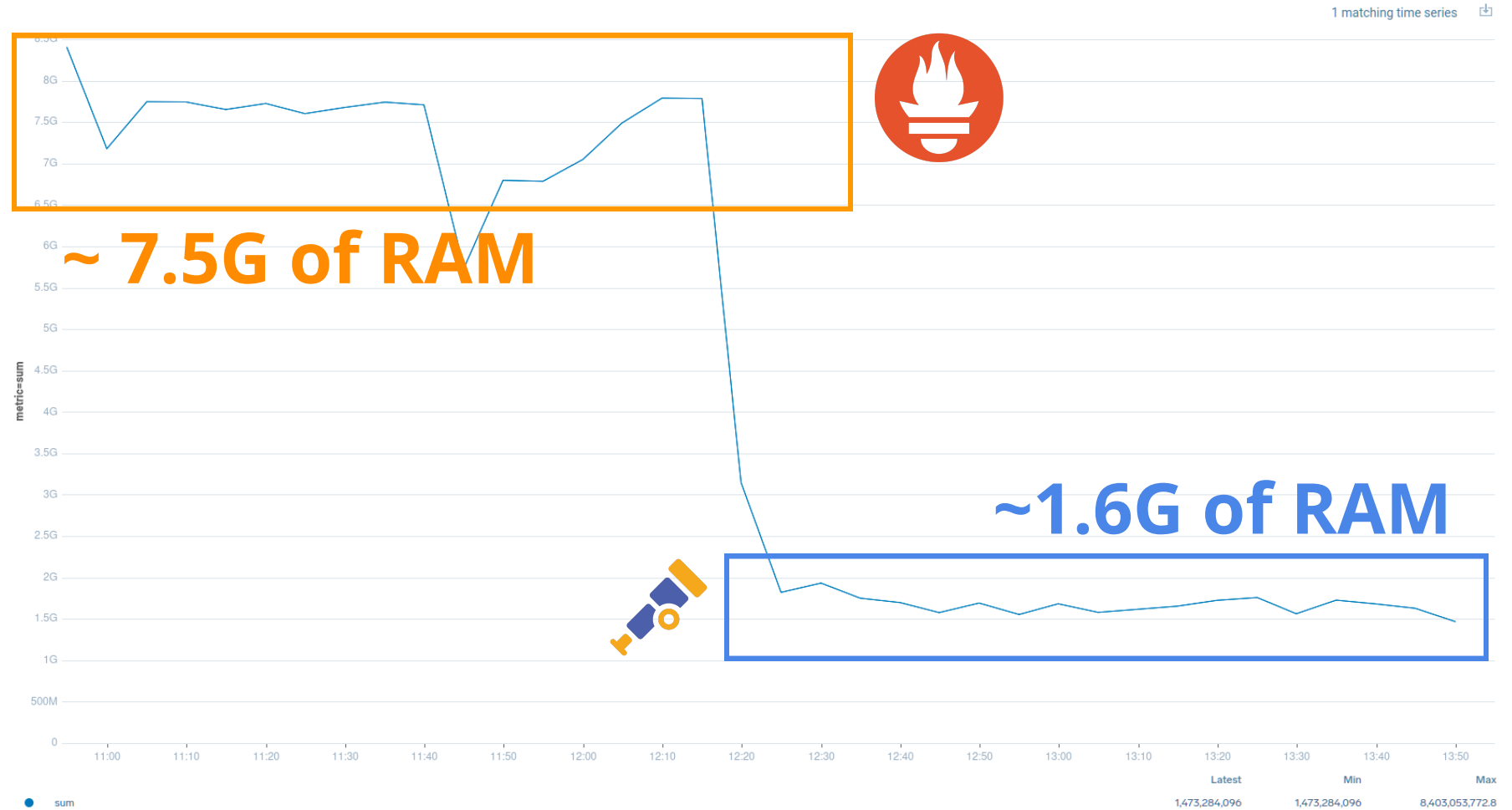
#B   namespace=`longhorn-system` metric=`container_cpu_usage_seconds_total` (statefulset=`longhorn-backup-sumologic-metrics-collector` OR statefulset=`prometheus-longhorn-backup-prometheus*`) | rate counter | sum | quantize 5m

Chart Time Series



K8s observability - OTel edition

Logs



Metrics



Traces



Metadata



sumo logic

K8s observability - OTel edition

Logs



Metrics



Traces



Metadata



sumo logic

K8s observability - OTel edition

Logs



Metrics



Traces



Metadata



sumo logic



OTC issues



Low hanging ~~fruit~~ bugs in 2022

✓ [receiver/filelogreceiver] Filelogreceiver doesn't read from file in case there are two or more empty lines at the beginning **bug** **priority:p2** **receiver/filelog**

#10128 by perk was closed on Jul 18, 2022

✓ [receiver/filelogreceiver] Filelogreceiver reads the same file twice in case there is an empty line at the beginning **bug** **priority:p2** **receiver/filelog**

#10127 by perk was closed on Jul 18, 2022

✓ [receiver/filelogreceiver] Filelogreceiver reads the same file in the loop in case there is an empty line between logs **bug** **priority:p2** **receiver/filelog**

#10125 by perk was closed on Jul 18, 2022



State as of 2023

Q is:open is:issue label:bug Labels

✕ Clear current search query, filters, and sorts

🕒 306 Open ✓ 1,011 Closed Author ▾ Label ▾ Projects

- 🕒 **Flaky test: TestRabbitmqIntegration** bug flaky test receiver/rabbitmq
#22134 opened yesterday by atoulme
- 🕒 **The awscloudwatch receiver isn't able to filter cloudwatch log streams reliably using a stream prefix** bug
needs triage receiver/awscloudwatch
#22123 opened yesterday by rnishtala-sumo
- 🕒 **replace_all_patterns function for attribute keys doesn not expand regex** bug processor/transform
#22094 opened 2 days ago by vainiusd
- 🕒 **[exporter/f5cloud] The link https://portal.cloudservices.f5.com is invalid** bug exporter/f5cloud
#22077 opened 3 days ago by astencel-sumo
- 🕒 **Metric export is failing due to buffer full error on Open telemetry** bug needs triage
#22071 opened 3 days ago by vermaprateek695
- 🕒 **[receiver/prometheus] Histograms without buckets are dropped** bug receiver/prometheus
#22070 opened 3 days ago by swiatekm-sumo

Some more #OpenTelemetry

opentelemetry.io

opentelemetry.io/docs/collector

opentelemetry.io/community

Community

- OpenTelemetry org at GitHub
 - Mailing lists
- GitHub discussions or Slack channel
 - Calendar
- Community page

Thank you!

Marcin "Perk" Stożek

[@marcinstozek](#) / [perk.pl](#)



slides.com/perk/journey-to-opentelemetry-collector