Grafana Labs

SCaLE

A Series of Fortunate Invents

An Open-Source Tour of Solutions for Scaling Prometheus

Éamon Ryan - March 16 2024

Éamon Ryan

Senior Principal Field Engineer

Grafana Labs

Disclosure

I work at
Grafana Labs

One of the projects
is maintained by
Grafana Labs

The goal is to be
Unbiased

Dimensional Data

Many integrations

Powerful queries

Many client libraries

Great visualization

Precise alerting

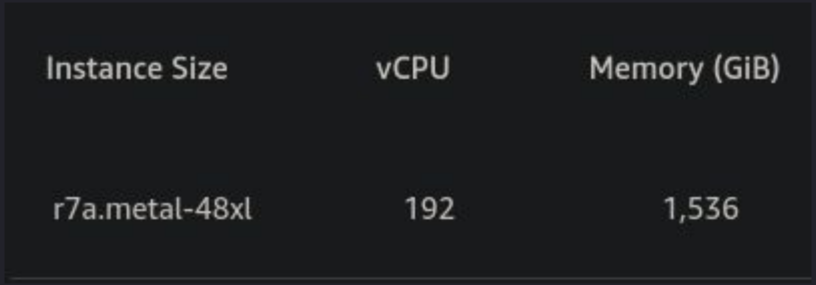
Efficient storage

Simple operation

https://prometheus.io/

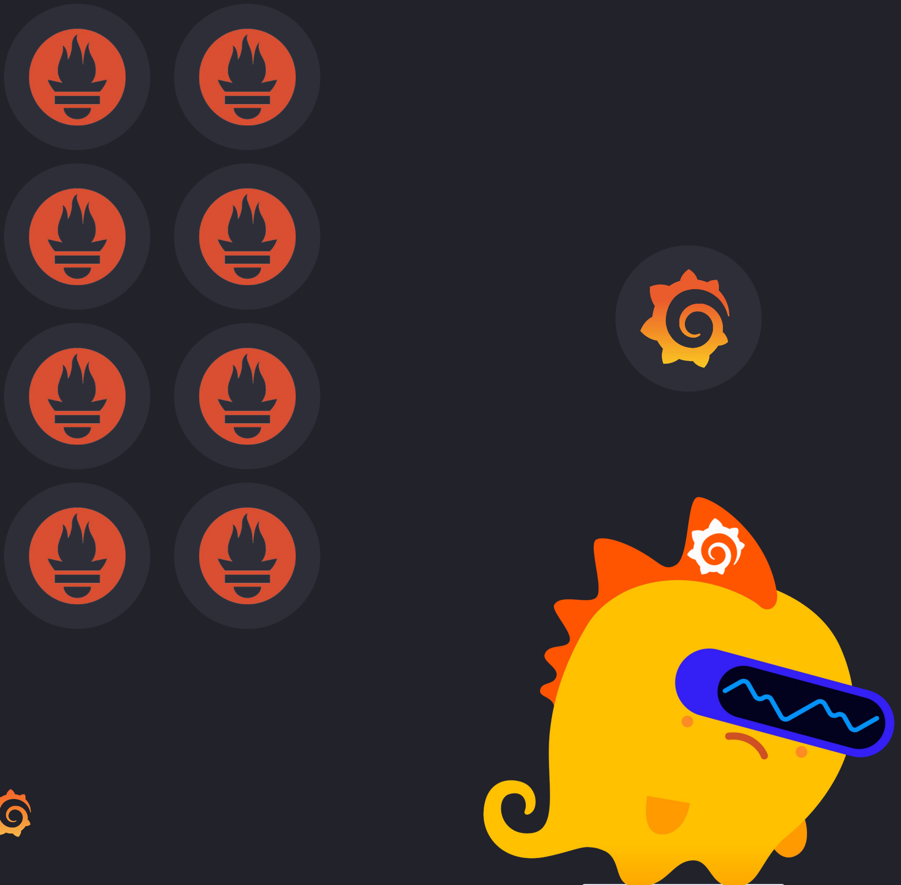
Prometheus Limitations - Size

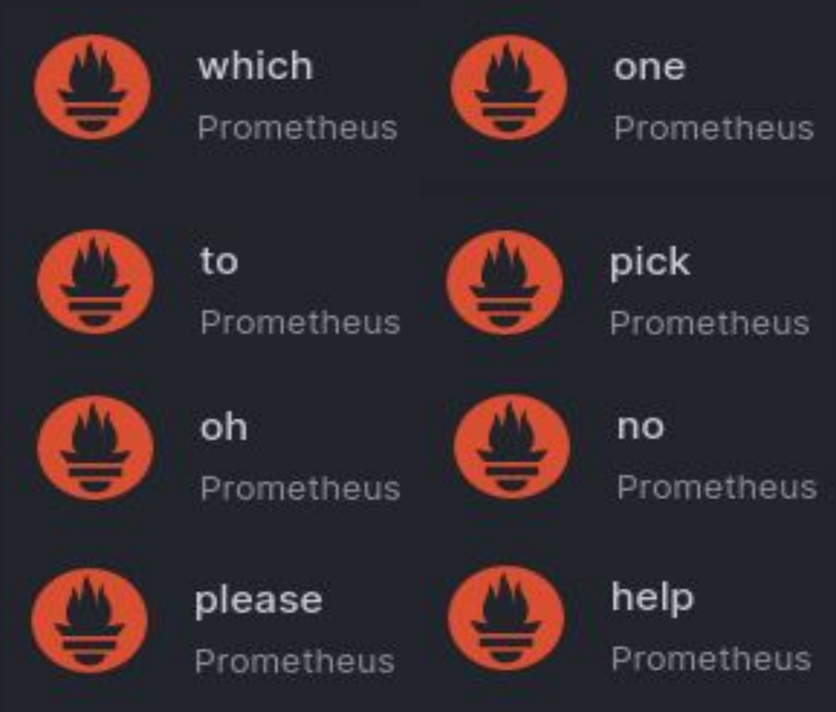
You can only scale Up - not Out



Instance Size	vCPU	Memory (GiB)
r7a.metal-48xl	192	1,536

Prometheus Limitations - Disparate Instances





which
Prometheus

one
Prometheus

to
Prometheus

pick
Prometheus

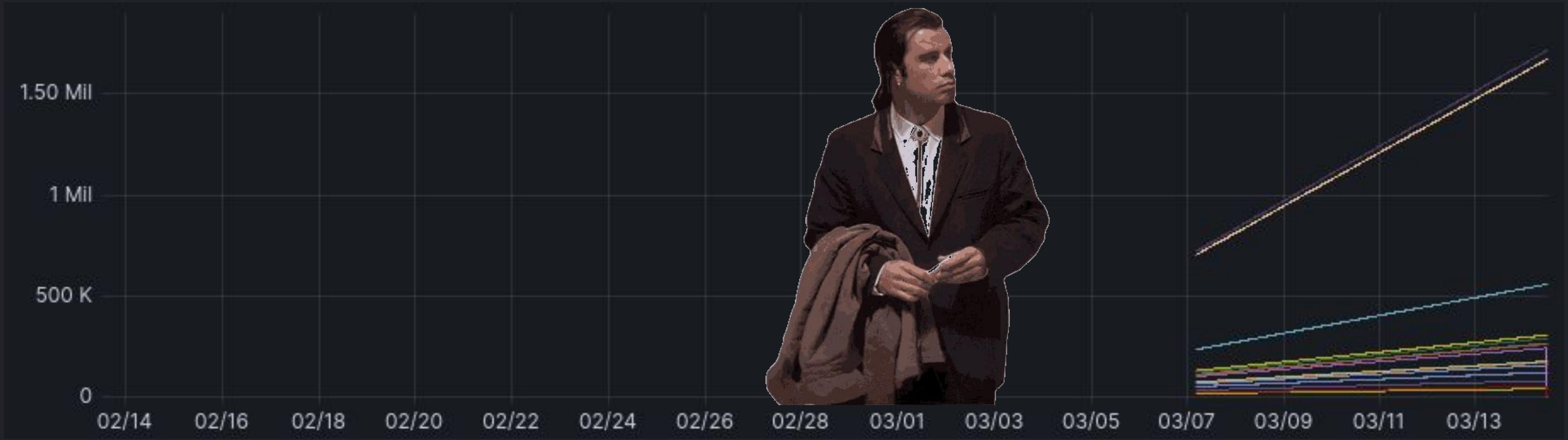
oh
Prometheus

no
Prometheus

please
Prometheus

help
Prometheus

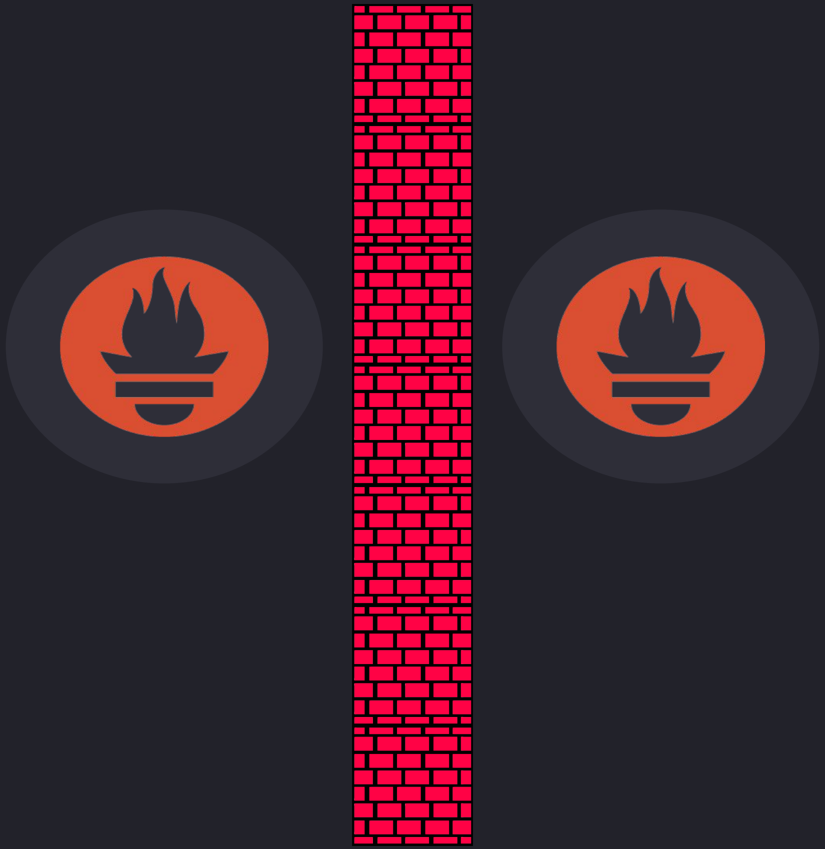
Prometheus Limitations - Retention



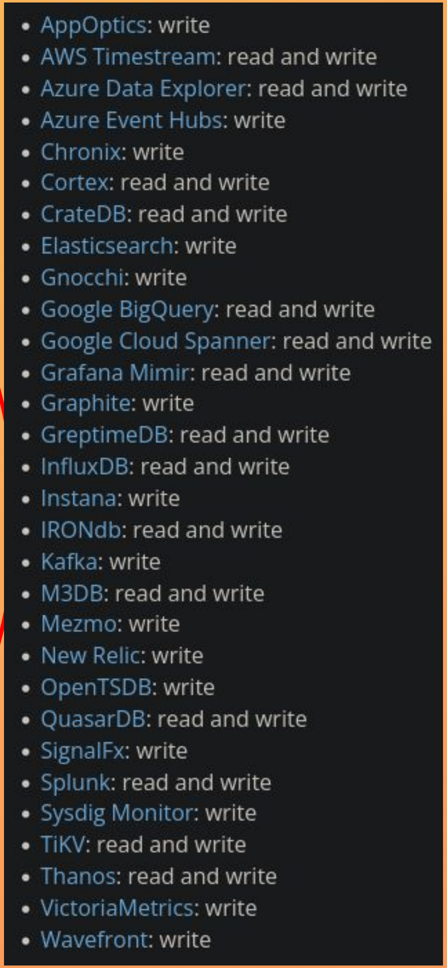
Prometheus Limitations - Tenancy

Prometheus Limitations - HA/Resiliency

- Config
- Scrape targets
- Disks
- TSDB
- No backfill



- Config
- Scrape targets
- Disks
- TSDB
- No backfill



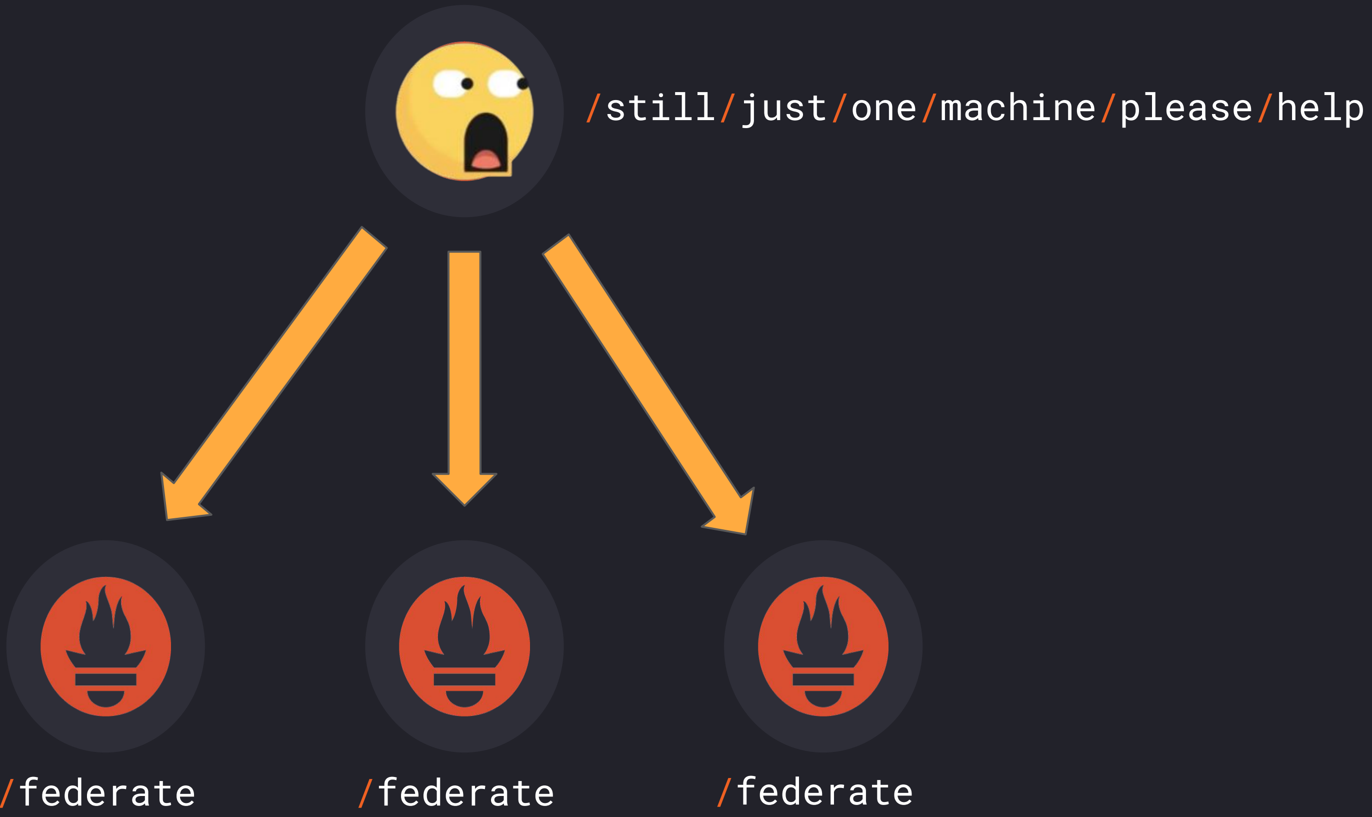
- AppOptics: write
- AWS Timestream: read and write
- Azure Data Explorer: read and write
- Azure Event Hubs: write
- Chronix: write
- Cortex: read and write
- CrateDB: read and write
- Elasticsearch: write
- Gnocchi: write
- Google BigQuery: read and write
- Google Cloud Spanner: read and write
- Grafana Mimir: read and write
- Graphite: write
- GreptimeDB: read and write
- InfluxDB: read and write
- Instana: write
- IRONdb: read and write
- Kafka: write
- M3DB: read and write
- Mezmo: write
- New Relic: write
- OpenTSDB: write
- QuasarDB: read and write
- SignalFx: write
- Splunk: read and write
- Sysdig Monitor: write
- TiKV: read and write
- Thanos: read and write
- VictoriaMetrics: write
- Wavefront: write

Source: prometheus.io

First-round eliminations

- CrateDB - Has a Prometheus adapter, v0.5.1, one maintainer, maybe future
- Elastic - Only supports writes, not reads as PromQL, non-OSI approved license
- Gnocchi - Only support writes, not reads as PromQL
- Graphite - Only support writes, not reads as PromQL, legacy
- GreptimeDB - Not at 1.0 yet (v0.7.1) PromQL at 82.12%, one to watch!
- InfluxDB - Clustering and HA not available in OSS version
- M3DB - Last release was v1.5.0 on April 7th 2022
- OpenTSDB - Last release was v2.4.1 on September 2nd, 2021
- Promscale - Discontinued / Deprecated in February 2023

Okay, what about Federation?



The Contenders

Cortex

VictoriaMetrics

Thanos

Grafana Mimir

Performance is not enough

Article: https://motherduck.com/blog/perf-is-not-enough/

“Performance in general, and general-purpose benchmarking in particular, is a poor way to choose a database.”

“You’re better off making decisions based on ease of use, ecosystem, velocity of updates, or how well it integrates with your workflow.”

- *Jordan Tigani, MotherDuck*

The Criteria

- Operational Mode
- How is the long-term data stored?
- Is OpenTelemetry (OTLP) native ingestion supported?
- PromQL Compatibility
- Known scale via blogs or articles etc.
- Multi-tenancy support
- Per-tenant limits support
- Native Histograms support
- Downsampling support
- Number of at least minor releases in the past 2 years

Cortex

Mode	Centralized - clients remote_write to central cluster
Storage	Block storage for recent data, object storage for rest
OTLP Ingestion	Work in progress
PromQL	100% Compatibility
Known Scale	Millions of series
Multi-tenancy	Yes - header-based using X-Scope-OrgID
Per-tenant limits	Yes
Native Histograms	Work in progress
Downsampling	Work in progress
Velocity	~2 minors/year over last 2 years

Cortex - References

Mode: https://cortexmetrics.io/docs/architecture/

Storage: https://cortexmetrics.io/docs/architecture/#blocks-storage

OpenTelemetry: https://github.com/cortexproject/cortex/issues/4981

PromQL: https://promlabs.com/promql-compliance-tests/

Known Scale: https://cortexmetrics.io/docs/case-studies/gojek/

Multi-tenancy: https://cortexmetrics.io/docs/guides/auth/

Per-tenant limits/stats: https://cortexmetrics.io/docs/configuration/configuration-file/#limits_config

Native Histograms: https://github.com/cortexproject/cortex/issues/5060

Downsampling: https://github.com/cortexproject/cortex/issues/4322

Velocity: https://github.com/cortexproject/cortex/releases

VictoriaMetrics

Mode	Centralized - clients remote_write to central cluster
Storage	Block storage - everything on disks
OTLP Ingestion	Yes
PromQL	74.16% Compatibility (MetricsQL)
Known Scale	1B series
Multi-tenancy	Yes, but multi-tenant rules in are Enterprise-only
Per-tenant limits	Enterprise-only, including per-tenant statistics
Native Histograms	No
Downsampling	Enterprise-only
Velocity	~5-10 minors/year over last 2 years

VictoriaMetrics - References

Mode: https://docs.victoriametrics.com/vmagent/

Storage: https://github.com/VictoriaMetrics/VictoriaMetrics/issues/38

OpenTelemetry: https://docs.victoriametrics.com/#sending-data-via-opentelemetry

PromQL: https://promlabs.com/promql-compliance-tests/ and https://docs.victoriametrics.com/metricsql/

Known Scale: https://medium.com/criteo-engineering/victoriametrics-a-prometheus-remote-storage-solution-57081a3d8e61

Multi-tenancy: https://docs.victoriametrics.com/cluster-victoriametrics/#multitenancy but multi-tenant ruler is Enterprise-only: https://docs.victoriametrics.com/operator/resources/vmrule/#multitenancy

Per-tenant limits/stats: https://docs.victoriametrics.com/pertenantstatistic/ and https://victoriametrics.com/products/enterprise/

Native Histograms: https://github.com/VictoriaMetrics/VictoriaMetrics/issues/3733

Downsampling: https://docs.victoriametrics.com/cluster-victoriametrics/#downsampling

Velocity: https://github.com/VictoriaMetrics/VictoriaMetrics/releases

Thanos

Mode	Sidecar <-> Prometheus / Centralized with Receiver
Storage	Block storage for recent data, object storage for rest
OTLP Ingestion	Not currently being worked on
PromQL	100% Compatibility
Known Scale	1B series
Multi-tenancy	Yes - using external_labels
Per-tenant limits	Experimental and in Receiver only
Native Histograms	Yes
Downsampling	Yes
Velocity	~3-4 minors/year over last 2 years

Thanos - References

Mode: https://thanos.io/tip/components/sidecar.md/ and https://thanos.io/tip/components/receive.md/

Storage: Above links as well as https://thanos.io/tip/components/store.md/

OpenTelemetry: https://github.com/thanos-io/thanos/issues/6932

PromQL: https://promlabs.com/promql-compliance-tests/

Known Scale: https://thanos.io/blog/2022-09-08-thanos-at-medallia/

Multi-tenancy: https://thanos.io/tip/operating/multi-tenancy.md/

Per-tenant limits/stats: https://thanos.io/tip/components/receive.md/#limits--gates-experimental and https://github.com/thanos-io/thanos/issues/3819

Native Histograms: https://github.com/thanos-io/thanos/issues/5907

Downsampling: https://thanos.io/tip/components/compact.md/#downsampling

Velocity: https://github.com/thanos-io/thanos/releases

Grafana Mimir

Mode	Centralized - clients remote_write to central cluster
Storage	Block storage for recent data, object storage for rest
OTLP Ingestion	Yes
PromQL	100% Compatibility
Known Scale	1B series
Multi-tenancy	Yes - header-based using X-Scope-OrgID
Per-tenant limits	Yes
Native Histograms	Yes
Downsampling	Work in progress
Velocity	~5 minors/year over last 2 years

Grafana Mimir - References

Mode: https://grafana.com/docs/mimir/latest/get-started/about-grafana-mimir-architecture/

Storage: https://grafana.com/docs/mimir/latest/get-started/about-grafana-mimir-architecture/#long-term-storage

OpenTelemetry: https://grafana.com/docs/mimir/latest/configure/configure-otel-collector/#otlp

PromQL: https://promlabs.com/promql-compliance-tests/

Known Scale: https://grafana.com/blog/2022/04/08/how-we-scaled-our-new-prometheus-tsdb-grafana-mimir-to-1-billion-active-series/

Multi-tenancy: https://grafana.com/docs/mimir/latest/manage/secure/authentication-and-authorization/

Per-tenant limits/stats: https://grafana.com/docs/mimir/latest/references/configuration-parameters/#limits

Native Histograms: https://grafana.com/docs/mimir/latest/configure/configure-native-histograms-ingestion/

Downsampling: https://github.com/grafana/mimir/pull/5028

Velocity: https://github.com/grafana/mimir/releases

Solution				
Modes	Centralized	Centralized	Sidecar/Centralized	Centralized
Storage	Block, Object	Block/Disk only	Block, Object	Block, Object
OTLP Ingestion	WIP	Yes	No	Yes
PromQL	100%	74.16%	100%	100%
Known Scale	Millions of series	1B series	1B series	1B series
Multi-tenancy	Yes (headers)	Yes* (URIs/labels)	Yes (labels)	Yes (headers)
Per-tenant limits	Yes	Enterprise-only	Experimental	Yes
Native Histograms	WIP	No	Yes	Yes
Downsampling	No	Enterprise-only	Yes	WIP
Velocity	2 minors/year	5-10 minors/year	3-4 minors/year	5 minors/year

Lots of good options

Choose what's right for your needs

No single best answer

6:15pm - “So you want to build an Incident Response stack using OpenTelemetry?” - Annanay Agarwal, Grafana Labs - Room 107

@eamon@grafana.social

eamonrryan

Q&A

Come to the booth!

grafana.com/oss/