OpenMetrics: What Does It Mean for You

Brian Brazil
Founder

Robust Perception

Who am I?

Engineer passionate about running software reliably in production.

- One of the main developers of Prometheus
- Author of Prometheus: Up&Running
- Founder of Robust Perception
- Based in Dublin
- Part of the core OpenMetrics group

OPEN METRICS

Last year RichiH was here talking about OpenMetrics.

Since then we've figured out almost all the details, so you can learn what it'll mean for you.

Robust **Perception**

A Recap

Historically the closest to a metrics standard is SNMP.

SNMP has many warts, such as being chatty without windowing and thus slow

Data model doesn't cover some common cases, so often there's per-vendor variation that follows the letter of the law but not the spirit.

Various other proprietary/monitoring system specific formats, none very common.

Enter Prometheus

Prometheus text format became a de-facto standard around 2-3 years ago.

Monitoring systems no longer have to code each individual integration, they can re-use our exporters and libraries! Labels are a big win!

Result: Hundreds of exporters, all the main monitoring players support the Prometheus text format.

Not for everyone

While common, the Prometheus format is still the ***Prometheus*** format.

Getting all relevant vendors to support it (and not invent their own thing) can be a challenge.

So take the Prometheus format, and evolve into an actual neutral standard by working with engineers from "competing" monitoring systems.

Commitments

Cloudflare
CNCF at large
GitLab
Google
Grafana
InfluxData
Prometheus
RobustPerception
SpaceNet
Uber
...

Process

We mainly work via consensus at bi-weekly VC meetings, and also had one bigger in-person meeting.

Have had attendees from many different companies over time.

At this late stage we're no longer trying to onboard people as we're discussing fine details. It took years to agree on all the terminology, concepts, and scope as-is.

Fixing warts, supporting other systems

Much tighter specification of the format e.g. spacing, escaping.

Allowing for nanosecond resolution timestamps. Int64 values.

Unit as new metadata.

_created for child creation.

Explicit EOF to detect interrupted metrics.

Considerations for both pull and push.

Optional support for protobufs.

What's New: Exemplars

Exemplars allow linking certain metrics to example traces:

```
# TYPE foo histogram
foo_bucket{le="0.01"} 0
foo_bucket{le="0.1"} 8 # {id="abc"} 0.043
foo_bucket{le="1"} 10 # {id="def"} 0.29
foo_bucket{le="10"} 17 # {id="ghi"} 7.73
foo_bucket{le="+Inf"} 18
foo_count 18
foo_sum 324789.3
foo_created  1520430000.123
```

Robust Perception

The Important Slide: Breaking Changes

Counter now requires _total on the time series.

If you're already following that convention all is good, no impact.

Otherwise your metrics will be changed as the convention is enforced as a standard.

Timestamps are in seconds rather than milliseconds - unlikely to be a big deal.

Current Status: Prometheus

Prometheus Python client is the reference implementation, and uses the OpenMetrics data model internally.

Prometheus will negotiate preferentially for OpenMetrics when scraping.

Info/Enum are now first class features, no need to implement them by hand. Gracefully handled if exposed via the Prometheus text format.

Python client's parser can be used to test exposition compliance.

Current Status: Other Implementations

DataDog supports ingestion of OpenMetrics, and contributed performance improvements to the Python parser.

OpenTelemetry is focused mainly on tracing, it plans to support OpenMetrics exposition for metrics.

Beware imitations: Many projects talking about supporting OpenMetrics exposition but actually mean the Prometheus text format.
Still a good thing, but a bit confusing.

Spotting OpenMetrics

Prometheus:

```
# TYPE foo_seconds_total counter
foo_seconds_total 1.0
```

OpenMetrics (including optional UNIT and _created):

```
# TYPE foo_seconds counter
# UNIT foo_seconds seconds
foo_seconds_total 1.0
foo_seconds_created 1572628096.0
# EOF
```

Current Status: Standard

Draft RFC in process of being written.

Basic text format spec is done. Proto in progress.

We keep getting delayed by life, and much wordsmithing is needed.

Official standard compliance test suite for parsers mostly there, based on Python client's parser's unittests. Python client can be used for exposition compliance testing.

Next Steps

Complete the draft RFC.

Bring the draft to the IETF.

Support OpenMetrics in more Prometheus client libraries.

Prometheus itself considering some exemplar support.

Encourage projects like Grafana and Loki to make use of new metadata.

Transitioning to OpenMetrics

Add in _total now for counters, so you can control that change.

Otherwise this should be a noop for those using existing client libraries.

If you're not using a client library, ensure you're sending an appropriate Content-Type if you plan on continuing to expose Prometheus text format.

Similarly, if you're writing a scraper and want Prometheus text format (or OpenMetrics) set your Accept header accordingly.

Questions?

https://github.com/OpenObservability/OpenMetrics/