What You Need to Know About OpenMetrics

OPEN METRICS

Brian Brazil

Robust Perception

Richard Hartmann

Grafana Labs

Who are we?

- Brian Brazil
  - Founder of Robust Perception
  - One of the main developers of Prometheus
  - Author of Prometheus: Up&Running
- Richard “RichiH” Hartmann
  - Community Directory at Grafana Labs
  - Prometheus team
  - CNCF SIG Observability chair
  - OpenMetrics founder

Before Prometheus

- Historically, the closest to a standard is SNMP
- Many solutions based on ancient technology like ASN.1
- Often chatty and slow
- Many data formats are proprietary, hard to implement, or both
- Data models encourage per-vendor variations that follow the letter of the law, but not the spirit
- Hierarchical data models almost never fit a user's needs

After Prometheus

- Prometheus is the de-facto standard in cloud-native metric monitoring and beyond
  - Same is true for Prometheus exposition format
- Ease of exposing data has lead to an explosion in compatible metrics endpoints
  - Thousands of exporters and integrations
- Standard exporters and libraries make integrating this easy
  - New: Created an exporter scaffold so you can focus on your metrics, not writing HTTP endpoints
- Labelsets allow near-perfect access to your data

Politics

- Some other projects & vendors are torn about adopting something from a competing product or project
- Especially traditional vendors prefer to support official standards
- Re-use installed base of Prometheus
  - Ease of adoption
  - Reject kitchen sink approach, do one thing well, and remain focused and opinionated
- Many competing companies collaborated on OpenMetrics and helped shape it
- The result is an actually neutral standard

Commitments, partial

- Chronosphere
- Cloudflare
- CNCF at large
- GitLab
- Google
- Grafana
- InfluxData
- **OpenTelemetry**
- Prometheus
- Robust Perception
- SpaceNet
- Uber
- more

People & process

- The marathon runners
  - Ben Kochie, GitLab
  - Brian Brazil, Robust Perception
  - Richard Hartmann, Grafana Labs
  - Rob Skillington, Chronosphere
  - Honorable mention: Sumeer Bhola
- We mainly work via consensus at fortnightly video calls
- Many attendees from many different companies over time

OpenMetrics and You

- The format is largely the same as the Prometheus format, with cleanups, and new features
- If you are using the Python client integration and Prometheus 2.5.0+, you have used OpenMetrics since late 2018
- Using our official client libraries will migrate you transparently to OpenMetrics without you noticing
- In general things will Just Work(™)

Breaking Changes

- Counters now require _total on the time series
  - Common convention, but now enforced
  - If your metric was `cpu_seconds`, our libraries will migrate you to `cpu_seconds_total`
- Timestamps are in seconds, for consistency
  - We use base units everywhere else
  - Used to be milliseconds
  - Exposing an explicit timestamp is possible, but usually an antipattern

Improvements & interoperability

- Cleaner and tighter specification, e.g. spacing, escaping
- Explicit EOF to detect incomplete scrapes
- Allowing for nanosecond resolution timestamps
- 64-bit integer values
- Unit as new metadata
- _created for metric creation & resets
- Considerations for both pull and push
- Text format still mandatory, reintroduce optional protobuf

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 219.3
foo_created 1520430000.123
```

Already in a Prometheus branch and in mainline Cortex and Thanos!

Current Status: Prometheus

1. Prometheus Python client is the reference implementation, and uses the OpenMetrics data model internally
2. Go has optional limited support, to allow for exemplars
3. Prometheus will negotiate preferentially for OpenMetrics when scraping
4. Info & Enum are now first class features
    - No need to implement them by hand
    - Automatically exposed backwards-compatible if scraped via the Prometheus text format

Current Status: Other Implementations

- DataDog supports ingestion of OpenMetrics, and contributed performance improvements to the Python parser
- OpenTelemetry plans to support OpenMetrics as a first-class wire format
  - Prometheus experience helping to shape OpenTelemetry Metrics design

Caution: In part due to our delays, most other projects claiming OpenMetrics compatibility actually mean the Prometheus text format

Spotting OpenMetrics

Prometheus exposition format 0.0.4:

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

OpenMetrics 1.0 (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

- Text format & protobuf specs are done
- Long-form prose version done
  - Being wordsmithed and compressed into something we can submit as Internet Draft
  - ~52 A4 pages of pure spec, 8 pages compressed by ~50%, 36 pages to go
- Official standard compliance test suite for parsers almost ready
  - Based on Prometheus Python client's parser's unittests
  - Python client can be used for exposition compliance testing.

Next Steps

- Complete the Internet Draft, submit for RFC process
- Contact with IETF Ops WG established, active interest in adoption
  - Special thanks to Warren Kumari
- Support OpenMetrics in the other official Prometheus client libraries
- Prometheus exemplar support
- Downstream projects like M3DB, 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.
- Client library & custom integration authors: ensure you're sending an appropriate Content-Type if you plan on continuing to expose Prometheus text format
- Scraper & ingester authors set Accept header to negotiate the Prometheus and/or OpenMetrics format as needed

Questions?

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