Evolution of Telemetry @ Bloomberg

GrafanaCon EU
March 1, 2018

Stig Sorensen
Head of Telemetry

Sean Hanson
Heart of Telemetry

TechAtBloomberg.com

Bloomberg

Agenda

- Background on Bloomberg
- Evolving Monitoring

Bloomberg at a glance

- **Bloomberg Professional Service**
  (325,000+ subscribers)
- Bloomberg News & Media (More reporters than The New York Times + Washington Post + Chicago Tribune)
- Bloomberg Enterprise

Financial Knowledge & Data

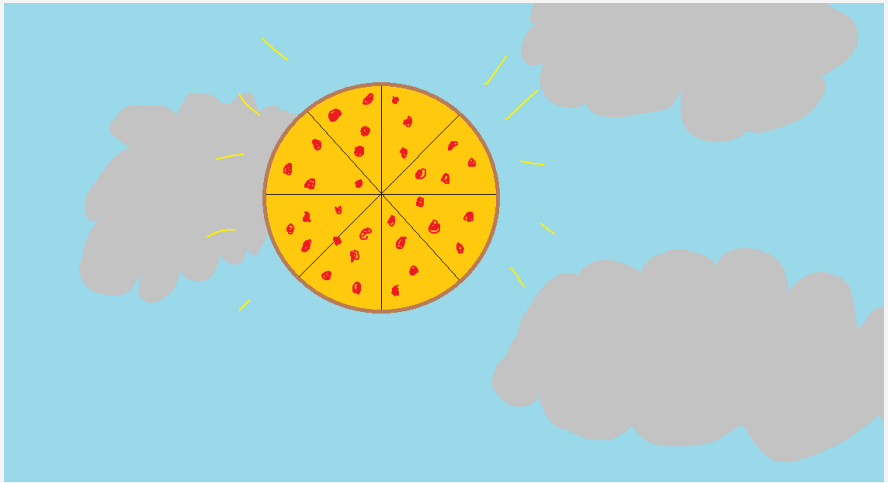
Massive engineering effort to accomplish this:
more than 5K software engineers (of 19K employees)

Background

- Our own datacenters
- Diverse Server architectures
- Mix of proprietary and an increasing amount of open-source technologies
- Multiple delivery platforms (Bloomberg Professional Service, Web, Mobile)

Team Mission

Deliver monitoring and alarming as a service for Bloomberg's infrastructure and applications.



TechAtBloomberg.com

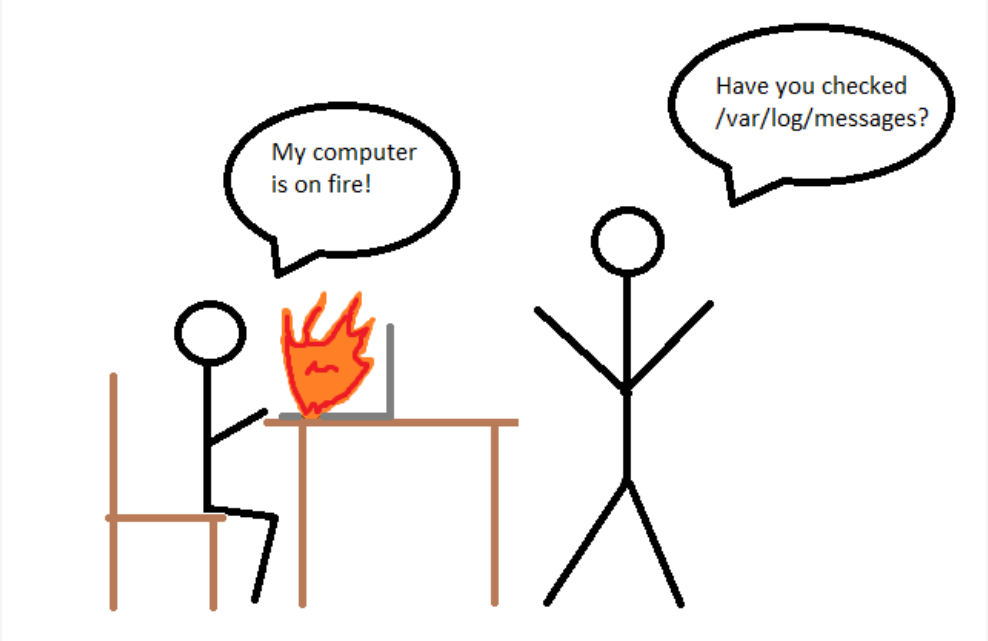
Bloomberg

Challenges

- Varied data sets, including network latency, process stats, machine stats and application stats
- Systems, application and service teams
- Engineers and support staff
- Flexible alarming
- Powerful graphing
- Single-system

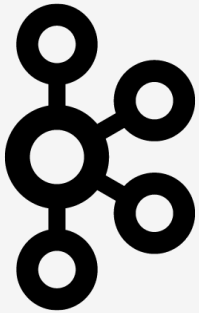
Legacy Monitoring Systems

- Monitoring Systems were tailor-made
- Data silo'd behind different:
  - APIs
  - UIs
  - Developers / Operations
- Not always isolated from outages



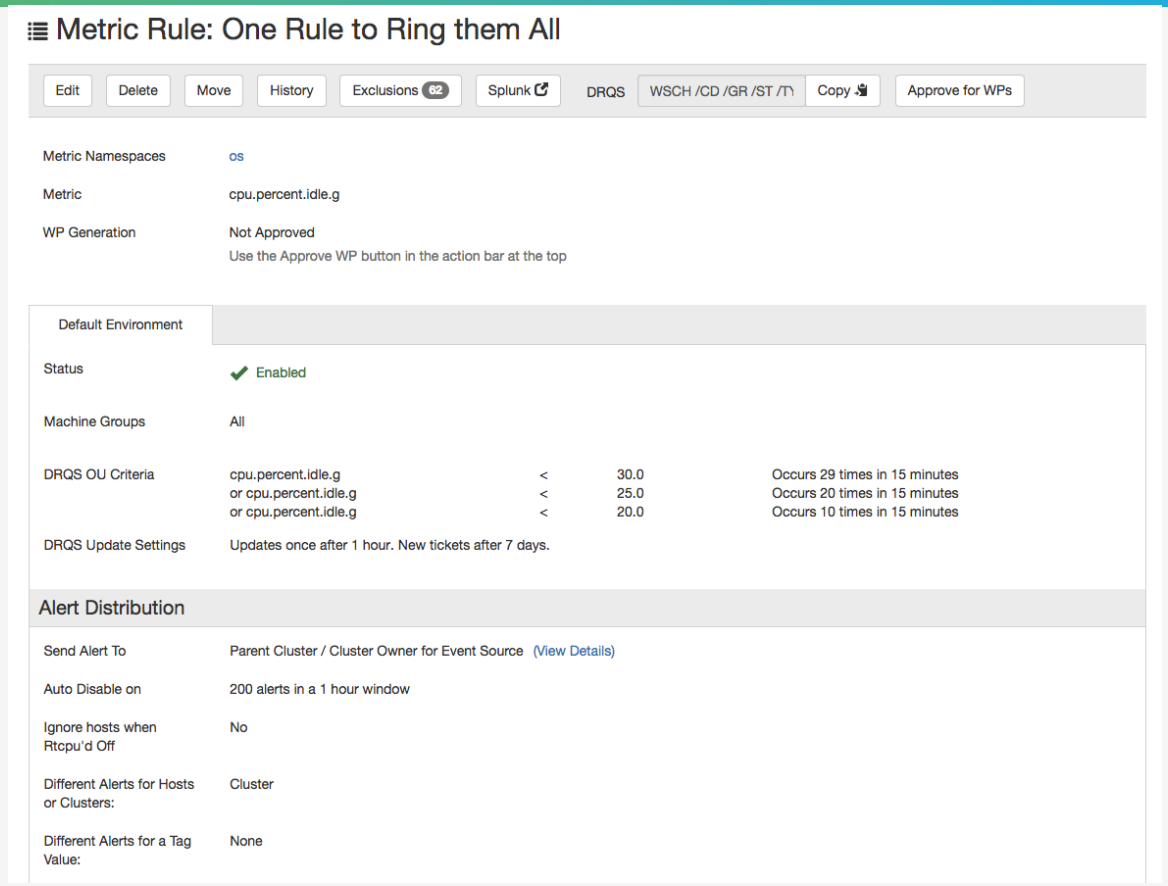
A New Hope

- No external Bloomberg dependencies (in the critical path)
- Liberal use of OSS



- Centralized Processing with Lightweight Client APIs
- Self-monitoring?

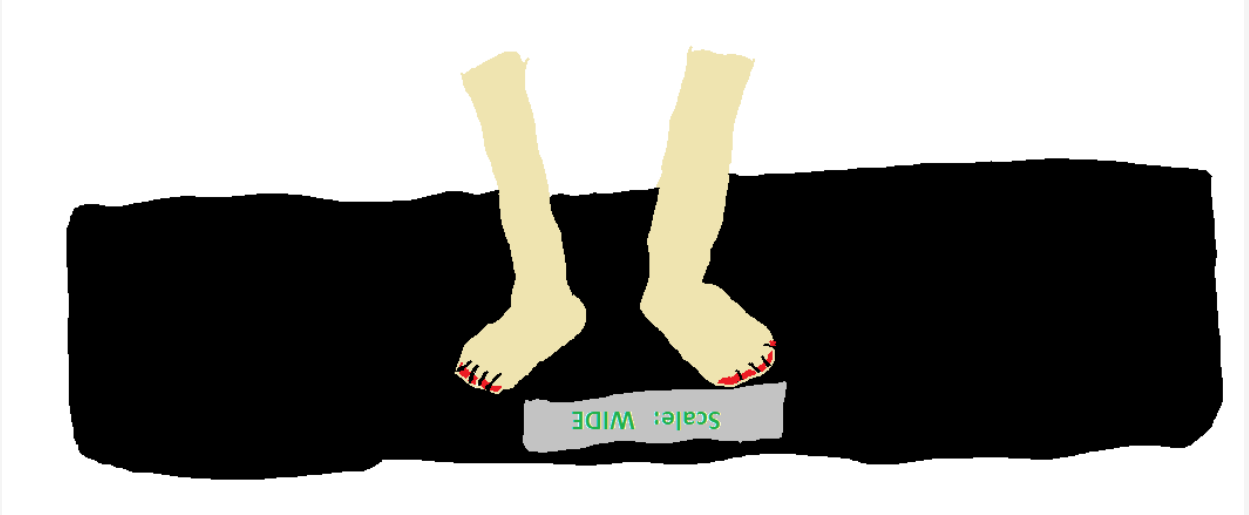
Self Service



Storing Time-series Data

1. Scales wide!
2. Supports periodic and non-periodic data
3. Efficient storage of long-term data
4. Flexible Query API

That's a scale ->



Bloomberg

Scale of Metrics Infrastructure

Currently:

- 2.5 million datapoints / sec
- 200 million time series
- Maximum cardinality of 500k

Goal:

- 20 million datapoints / sec
- 1 billion time series

New requirements list

1. Scales wide!...and efficiently
2. Supports periodic and non-periodic data
3. Efficient storage of long-term data...with rollups
4. Flexible Query API...with functions to derive metrics from others
5. Allows many tags for multi-dimensional data
6. Supports low-latency queries for “hot” data (young store, cache, etc.)
7. Configurable retention for different data
8. Metadata queries to help drive exploration/user interaction

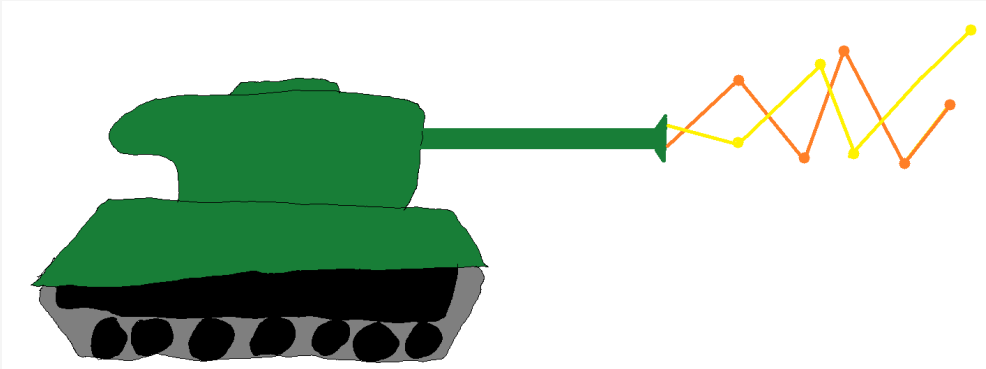
Evaluating Our Options

1. Tack on additional layers to our current system
2. Find an OS product that can be extended via contribution
3. Write our own!!

“Hey, the Grafana folks mentioned something that they built called ‘MetricTank’. It’s ‘beta’, but they say it’s reliable. We should check it out.”

-- Stig

My first thought →



How MetricTank stacked up

- Scales wide!...and efficiently ✓
- Supports periodic and non-periodic data ✓
- Efficient storage of long-term data...with rollups ✓
- Flexible Query API...with functions to derive metrics from others ✓
- Allows many tags for multi-dimensional data ✘
- Supports low-latency queries for “hot” data (young store, cache, etc.) ✓
- Configurable retention for different data ✓
- Metadata queries to help drive exploration/user interaction ⚠

How MetricTank stacked up

- Scales wide!...and efficiently ✓
- Supports periodic and non-periodic data ✓
- Efficient storage of long-term data...with rollups ✓
- Flexible Query API...with functions to derive metrics from others ✓
- Allows many tags for multi-dimensional data ✓
- Supports low-latency queries for “hot” data (young store, cache, etc.) ✓
- Configurable retention for different data ✓
- Metadata queries to help drive exploration/user interaction ⚠

How MetricTank stacked up

- Scales wide!...and efficiently ✓
- Supports periodic and non-periodic data ✓
- Efficient storage of long-term data...with rollups ✓
- Flexible Query API...with functions to derive metrics from others ✓
- Allows many tags for multi-dimensional data ✓
- Supports low-latency queries for “hot” data (young store, cache, etc.) ✓
- Configurable retention for different data ✓
- Metadata queries to help drive exploration/user interaction ✓

What's next?

- Scales wide!...and efficiently ✓
- Supports periodic and non-periodic data ✓
- Efficient storage of long-term data...with rollups ✓
- Flexible Query API...with functions to derive metrics from others ✓
- Allows many tags for multi-dimensional data ✓
- Supports low-latency queries for "hot" data (young store, cache, etc.) ✓
- Configurable retention for different data ✓
- Metadata queries to help drive exploration/user interaction ✓

And:

- Query auditing
- Extrinsic tags
- Pre-aggregation pipeline
- Offline rollups for longer intervals
- Backfilling / Updating data

Lessons learned

- Scaling wide is great…but watch out for hidden bottlenecks
- Always be on the lookout for new technologies
- Build a better Metrics system and users will beat a path to your door
- ALWAYS abstract integration points

THANKS

TechAtBloomberg.com

Bloomberg