Evolution of Telemetry

@ Bloomberg

GrafanaCon EU
March 1, 2018

Stig Sorensen
Head of Telemetry

Sean Hanson
Heart of Telemetry
Agenda

• Background on Bloomberg

• Evolving Monitoring
Bloomberg at a glance

- **Bloomberg Professional Service** (325,000+ subscribers)


- 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)
Deliver monitoring and alarming as a service for Bloomberg’s infrastructure and applications.
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**

### Metric Rule: One Rule to Ring them All

<table>
<thead>
<tr>
<th>Action</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit</td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td></td>
</tr>
<tr>
<td>Move</td>
<td></td>
</tr>
<tr>
<td>History</td>
<td></td>
</tr>
<tr>
<td>Exclusions</td>
<td></td>
</tr>
<tr>
<td>Splunk</td>
<td></td>
</tr>
<tr>
<td>DRQS</td>
<td></td>
</tr>
<tr>
<td>WSCH/CD/GR/ST/TH</td>
<td></td>
</tr>
<tr>
<td>Copy</td>
<td></td>
</tr>
<tr>
<td>Approve for WPs</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Metric Namespaces</th>
<th>os</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric</td>
<td>cpu.percent.idle.g</td>
</tr>
<tr>
<td>WP Generation</td>
<td>Not Approved</td>
</tr>
<tr>
<td></td>
<td>Use the Approve WP button in the action bar at the top</td>
</tr>
</tbody>
</table>

#### Default Environment

<table>
<thead>
<tr>
<th>Status</th>
<th>Enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine Groups</td>
<td>All</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DRQS OU Criteria</th>
<th>Comparison</th>
<th>Value</th>
<th>Occurrence</th>
</tr>
</thead>
<tbody>
<tr>
<td>cpu.percent.idle.g</td>
<td>&lt;</td>
<td>30.0</td>
<td>Occurs 29 times in 15 minutes</td>
</tr>
<tr>
<td>or cpu.percent.idle.g</td>
<td>&lt;</td>
<td>25.0</td>
<td>Occurs 20 times in 15 minutes</td>
</tr>
<tr>
<td>or cpu.percent.idle.g</td>
<td>&lt;</td>
<td>20.0</td>
<td>Occurs 10 times in 15 minutes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DRQS Update Settings</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Updates once after 1 hour, New tickets after 7 days.</td>
<td></td>
</tr>
</tbody>
</table>

### Alert Distribution

<table>
<thead>
<tr>
<th>Send Alert To</th>
<th>Parent Cluster / Cluster Owner for Event Source (View Details)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto Disable on</td>
<td>200 alerts in a 1 hour window</td>
</tr>
<tr>
<td>Ignore hosts when</td>
<td>No</td>
</tr>
<tr>
<td>Rtpcap Off</td>
<td></td>
</tr>
<tr>
<td>Different Alerts for Hosts or Clusters</td>
<td>Cluster</td>
</tr>
<tr>
<td>Different Alerts for a Tag Value</td>
<td>None</td>
</tr>
</tbody>
</table>
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 ->
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