Prometheus: A Next-Generation Monitoring System

Björn Rabenstein, Julius Volz
SoundCloud

SREcon Dublin
May 14, 2015
History
From a monolith...
...to microservices
Existing monitoring setup in 2012...
Service monitoring
Host monitoring
Alerting
Prometheus
Multi-dimensional data model

api_http_requests_total{method="GET", endpoint="/api/tracks", status="200"} 2034834

(like OpenTSDB)
Operational simplicity

$ go build
$ ./prometheus

(not like OpenTSDB)
Scalable data collection

Thousands of targets.
Hundreds of thousands of samples per second.
Millions of time series.
On a single monitoring server.
Running many servers is easy, too…
Pull, not push.
Powerful query language

topk(3, sum(rate(bazooka_instance_cpu_time_ns[5m])) by (app, proc))

sort_desc(sum(bazooka_instance_memory_limit_bytes - bazooka_instance_memory_usage_bytes) by (app, proc))
### Expression browser

```sql
sort_desc(sum(bazooka_instance_memory_limit_bytes - bazooka_instance_memory_usage_bytes) by (app, proc)) / 1024 / 1024 / 1024
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>{app=&quot;harsh-dagger&quot;,proc=&quot;api&quot;}</td>
<td>132.720862</td>
</tr>
<tr>
<td>{app=&quot;quality-locomotive&quot;,proc=&quot;web&quot;}</td>
<td>89.547081</td>
</tr>
<tr>
<td>{app=&quot;husky-long-oyster&quot;,proc=&quot;web&quot;}</td>
<td>68.982738</td>
</tr>
<tr>
<td>{app=&quot;vital-albatross&quot;,proc=&quot;api&quot;}</td>
<td>48.033772</td>
</tr>
<tr>
<td>{app=&quot;autopsy-gutsy&quot;,proc=&quot;widget&quot;}</td>
<td>47.410583</td>
</tr>
<tr>
<td>{app=&quot;western-python&quot;,proc=&quot;cruncher&quot;}</td>
<td>40.125926</td>
</tr>
<tr>
<td>{app=&quot;harsh-dagger&quot;,proc=&quot;api&quot;}</td>
<td>28.527714</td>
</tr>
<tr>
<td>{app=&quot;outstanding-dagger&quot;,proc=&quot;api&quot;}</td>
<td>26.119423</td>
</tr>
<tr>
<td>{app=&quot;gruesome-waterbird&quot;,proc=&quot;web&quot;}</td>
<td>17.666714</td>
</tr>
<tr>
<td>{app=&quot;gutsy-square&quot;,proc=&quot;public&quot;}</td>
<td>15.296242</td>
</tr>
<tr>
<td>{app=&quot;harsh-dagger&quot;,proc=&quot;web&quot;}</td>
<td>14.739327</td>
</tr>
<tr>
<td>{app=&quot;northern-electron&quot;,proc=&quot;api&quot;}</td>
<td>13.349815</td>
</tr>
</tbody>
</table>
Built-in graphing
## Alerts

<table>
<thead>
<tr>
<th>Alert Type</th>
<th>Alert Name</th>
<th>Labels</th>
<th>Active Since</th>
<th>Last Refreshed</th>
<th>Generated By</th>
<th>Alerting Rule</th>
<th>Silenced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Silence Alert</td>
<td>SomethingIsDown</td>
<td><em>job</em>: <code>code</code>, <em>service</em>: <code>foo</code></td>
<td>8m0.4s ago</td>
<td>400ms ago</td>
<td><code>http://localhost:9090/graph?%5B%7B%22expr%22%5D=%22...</code></td>
<td>ALERT SomethingIsDown IF (COUNT(up != 0)) BY (job) &gt; 1 FOR 1m WITH (service=&quot;foo&quot;); severity=&quot;page&quot;)</td>
<td>not silenced</td>
</tr>
<tr>
<td>Silence Alert</td>
<td>HighRPCRate</td>
<td><em>group</em>: <code>production</code>, <em>instance</em>: <code>localhost:8081</code></td>
<td>30.4s ago</td>
<td>400ms ago</td>
<td><code>http://localhost:9090/graph?%5B%7B%22expr%22%5D=%22...</code></td>
<td>ALERT HighRPCRate IF (rate(rpc_durations_microseconds_count[5m]) &gt; 12) FOR 15s WITH (severity=&quot;page&quot;)</td>
<td>not silenced</td>
</tr>
<tr>
<td>Silence Alert</td>
<td>HighRPCRate</td>
<td><em>group</em>: <code>production</code>, <em>instance</em>: <code>localhost:8081</code></td>
<td>30.4s ago</td>
<td>400ms ago</td>
<td><code>http://localhost:9090/graph?%5B%7B%22expr%22%5D=%22...</code></td>
<td>ALERT HighRPCRate IF (rate(rpc_durations_microseconds_count[5m]) &gt; 12) FOR 15s WITH (severity=&quot;page&quot;)</td>
<td>not silenced</td>
</tr>
<tr>
<td>Silence Alert</td>
<td>HighRPCRate</td>
<td><em>group</em>: <code>canary</code>, <em>instance</em>: <code>localhost:8082</code></td>
<td>30.4s ago</td>
<td>400ms ago</td>
<td><code>http://localhost:9090/graph?%5B%7B%22expr%22%5D=%22...</code></td>
<td>ALERT HighRPCRate IF (rate(rpc_durations_microseconds_count[5m]) &gt; 12) FOR 15s WITH (severity=&quot;page&quot;)</td>
<td>not silenced</td>
</tr>
</tbody>
</table>
Exporters

vs.

direct instrumentation
## Client libraries

<table>
<thead>
<tr>
<th>Official</th>
<th>Unofficial</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go</td>
<td>.NET / C#</td>
</tr>
<tr>
<td>Java (JVM)</td>
<td>Node.js</td>
</tr>
<tr>
<td>Ruby</td>
<td>Bash</td>
</tr>
<tr>
<td>Python</td>
<td></td>
</tr>
</tbody>
</table>
## Integrations

### Official exporters
- Node/system metrics exporter
- Graphite exporter
- Collectd exporter
- JMX exporter
- HAProxy exporter
- StatsD bridge
- AWS CloudWatch exporter
- Hystrix metrics publisher
- Mesos task exporter
- Consul exporter

### Unofficial exporters
- RethinkDB exporter
- Redis exporter
- scollector exporter
- MongoDB exporter
- Django exporter
- Google’s mtail log data extractor
- Minecraft exporter module
- Meteor JS web framework exporter

### Direct instrumentation
- cAdvisor
- Kubernetes
- Kubernetes-Mesos
- EtcD
- gokit
- go-metrics instrumentation library
- ...

...
Further reading

prometheus.io
developers.soundcloud.com/blog
boxever.com/tech-blog
Acknowledgements

Many contributors at SoundCloud…
...and by now so many more from everywhere.

Special thanks to:

Matt T. Proud (founding father)
Brian Brazil (Boxever)
Johannes “fish” Ziemke (Docker)