Paris OpenStack® Summit
This is your cloud
Monasca Deep Dive
Monitoring-as-a-Service (at Scale)
Roland Hochmuth, Sandy Walsh, Tong Li/ November 5, 2014
Agenda

Problem Statement
What is Monasca?
• Architecture
• Metrics
• Events/StackTach.v3
• Anomaly Detection

Current Status
Performance

Next Steps
Demo

Q&A
Problem Statement

**Monitoring-as-a-Service:** Lacking multi-tenant model

**Performance, scalability and data retention**

**Multiple uses of the data:** SLA calculations, business analytics, RCA, …

**Elasticity and dynamic run-time configurability**
- Metrics and Alarm management
- Spammy alerts and alert fatigue.

**Real-time event stream processing**

**Extensibility:** Integrate with other systems via API or internally

**Multiple Systems:** Internal/operational monitoring and external/customer-facing monitoring are separate systems. Health/Status different from metrics.

**Cryptic data:** Force fit metric/event names results in an impedance mismatch
What is Monasca?

**Monitoring-as-a-Service solution based on a first-class REST API**

**Highly-performant, scalable, fault-tolerant and capable of big data retention**

**Metrics storage/retrieval/statistics and alarm/thresholding engine**

**Notification system**

**Real-time event stream processing**

**Open-source and built-on open-source technologies such as:**
- Kafka: Performant, scalable, fault-tolerant, durable message queue. Used by LinkedIn, Twitter, …
- Apache Storm:
- Time-series databases: InfluxDB supported today. Elastic-search in progress.

**Consolidates multiple monitoring systems into a single solution**
- Used for both operational and customer facing monitoring.

**Extensible based on micro-services message bus architecture**
Metrics
REST API

**Metrics:** Create, query and get statistics for metrics

```json
{
    name: cpu.user_perc,
    dimensions: {
        hostname: hostname.domain.com,
        region: uswest,
        zone: 1,
        service: compute
    }
}
```

**Alarm Definitions**
- Alarm definitions are templates that are used to automatically create alarms based on matching metric names and dimensions
- Simple compound expression grammar: `avg(cpu.user_perc{}) > 85 or avg(disk_read_ops{device=vda}, 120) > 1000`
- Actions associated with alarms for state transitions to ALARM, OK and UNDETERMINED
- Severity (LOW, MEDIUM, HIGH, CRITICAL).

**Alarms:** Query and Delete alarms and query alarm state history

**Notification Methods:** e.g. Email address. Associated with alarm definitions
Monasca Agent

**Python monitoring agent**

**System metrics (cpu, memory, …)**

**Service metrics**
- RabbitMQ, MySQL, Kafka, and many others

**Application metrics**
- Built-in statsd daemon
- Python Monasca Statsd library

**VM metrics**

**Active checks**
- HTTP status checks and response times
- System up/down checks (ping and ssh)

**Runs any Nagios plugin**

**Extensible/Pluggable: Additional services can be easily added**
Horizon Dashboard
- Overview/Top-level drill-down
- Create/Read/Update/Delete alarm definitions using an expression builder
- Read/Delete alarms and alarm history
- Create/Read/Update/Delete notification methods

Grafana Dashboard (http://grafana.org/)
- Provides visualization of metrics
Events

StackTach.v3
Monasca Anomaly Engine implements real-time streaming anomaly detection

Two algorithms:
- Numenta Platform for Intelligent Computing (NuPIC) used by Grok
  - An open-source Python/C++ implementation of Hierarchical Temporal Memory
- Kolmogorov-Smirnov (K-S) Two Sample Test

Anomaly Engine
- Consumes metrics from the Kafka metrics topic
- Calculates predicted value and anomaly score (probability of an anomaly)
- Publishes calculated values as metrics to the Kafka metrics topic

Alarms can be created for Anomaly scores
Anomaly Detection (cpu.user_perc)
Current Status

Monasca and Stacktach.v3 is open-sourced in StackForge
Not an OpenStack incubated project, but we are targeting incubation
Metrics, Alarm Definitions, Alarms and Notification Methods completely
supported/functional and ready for production deployment

Who is working on it?

• HP
• RackSpace
• IBM

Who is deploying it?

• HP: Public Cloud and Helion distribution
• Time Warner Cable (TWC)
• Workday
Performance (Metrics inserts/sec)

Test Deployment (HP R&D Cluster):
- Three HP Proliant SL390s G7 servers
- InfluxDB cluster

Performance:
- Total end-to-end performance including storage in InfluxDB: Approximately 25K to 30K metrics/second.
- monasca-api: > 50K metrics/sec per single API server.
- monasca-api > 150K metrics/sec for a three node cluster with a load-balancing VIP.

If you need more database performance?
- Vertica is supported. Scales to hundreds of thousands of metrics per second.
Next Steps

Events/StackTach.v3 integration is in progress
Anomaly detection is in progress
Formalize micro-services architecture
  • Define message formats
  • Define how services are published and registered

Python port is in progress:
  • All components Python except for API and Threshold Engine
    • API is 75% ported to Python. Note, Java API is 100% functional
    • Threshold Engine is the only remaining Java component
Call to Action

Looking for contributors
• Monasca Service, StackTach.v3, Events, Anomaly Detection
• Monasca Agent: Help extend with additional services. E.g. Sensor data
• Help Integrate, Deploy, Test and Performance benchmarking

More info:
• Launchpad: https://launchpad.net/monasca
• Wiki: https://wiki.openstack.org/wiki/Monasca
• IRC: #openstack-monasca

Monasca development environment:
• monasca-vagrant (https://github.com/stackforge/monasca-vagrant): A turn-key development environment that installs Monasca and a Devstack VM
• Newly upgraded to use Ansible
Demo Recap

What did we just show?
1. An OpenStack Notification is sent to the Monasca Events API
2. The API publishes the notification to the Kafka raw events topic
3. The Transform Engine consumes, transforms (using StackTach Distiller) and publishes the event to the transformed event topic
4. The Events Engine consumes, adds to the StackTach Winchester Pipeline.
5. If the notification is a “compute.instance.create.end” event the Winchester pipeline handler fires, calculates the delta, and publishes to the Kafka metrics topic
6. The metric can then be alarmed on in the metrics pipeline or visualized
Metrics Sequence

1. POST metric
   - Monasca Metrics API

1.1 consume metrics
1.1.1 store metric
1.1.1 store alarm

1.2 consume metric
1.2.1 aggregate metric
1.2.2 publish metric
1.3 consume metric
1.3.1 threshold metric
1.3.2 publish alarm
1.4 consume alarm
1.4.1 send notification
   - SMS, Email, Heat, SMTP, Twilio, ...

Message Queue <<Kafka>>
- "metrics" topic
- "alarm-state-transitions" topic

Threshold Engine

Transform Engine

Persisters
- storage drivers

1.5 consume alarms
1.5.1 store alarm

Metrics and Alarms Database <<InfluxDB and Vertica>>
Events Sequence

1.1 consume raw event
1.1.1 transform raw event
1.2 consume raw event
1.2.1 store event
1.3 consume transformed event
2. publish metric
2.1 publish metric
2. fire event
Monitoring Overview (All Services Healthy)
Monitoring Overview (nova-api down)
### List Alarm Definitions

![Image of OpenStack Dashboard with Alarm Definitions]

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Notifications Enabled</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>API Status</td>
<td>API Status</td>
<td>True</td>
<td>Edit Alarm Definition</td>
</tr>
<tr>
<td>Disk Space</td>
<td>Disk space utilization percentage</td>
<td>True</td>
<td>Edit Alarm Definition</td>
</tr>
<tr>
<td>Compute Instance Create Time</td>
<td>Compute Instance Create Time</td>
<td>True</td>
<td>Edit Alarm Definition</td>
</tr>
<tr>
<td>CPU User Percent</td>
<td>CPU User Percent</td>
<td>True</td>
<td>Edit Alarm Definition</td>
</tr>
</tbody>
</table>

Displaying 4 items
Create Alarm Definition (Disk Space)

Description:
The Name field is used to identify the notification method. The Expression field which if true, triggers a notification to be sent. See Alarm Expressions for how to write an expression. The Alarm Actions field contains the list of Notification that should be sent when transitioning to an ALARM state.

Name *
Disk Space Utilization

Expression *
avg disk_space_utilization_perc > 85

Matching Metrics

Apply function to metrics
individually
Edit Alarm Definition (API Status)

![Edit Alarm Definition](image)

**Description:**
The Name field is used to identify the notification method.

The Expression field which is true, triggers a notification to be sent. See Alarm Expressions for how to write an expression.

The Alarm Actions field contains the list of Notification that should be sent when transitioning to an ALARM state.

Name
- API Status

Expression
- `avg(http_status) > 0`

Apply function to metrics
- individually
- as a group

Description
- API Status

Severity
- Critical

Notifications Enabled

Notifications
- Name: [Name]

+ Add more
List All Alarms

<table>
<thead>
<tr>
<th>Status</th>
<th>Metric Name</th>
<th>Metric Dimensions</th>
<th>Definition</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔</td>
<td>http_status</td>
<td>url=<a href="http://localhost:8081/healthcheck,hostname=mini-mon,service=monitoring">http://localhost:8081/healthcheck,hostname=mini-mon,service=monitoring</a></td>
<td>API Status</td>
<td>Graph Metric</td>
</tr>
<tr>
<td></td>
<td>http_status</td>
<td>url=<a href="http://localhost:8774/v2.0,hostname=devstack,service=nova_api">http://localhost:8774/v2.0,hostname=devstack,service=nova_api</a></td>
<td>API Status</td>
<td>Graph Metric</td>
</tr>
<tr>
<td></td>
<td>http_status</td>
<td>url=<a href="http://localhost:8776/v2.0,hostname=devstack,service=cinder_api">http://localhost:8776/v2.0,hostname=devstack,service=cinder_api</a></td>
<td>API Status</td>
<td>Graph Metric</td>
</tr>
<tr>
<td></td>
<td>http_status</td>
<td>url=<a href="http://localhost:8080/healthcheck,hostname=devstack,service=swift_api">http://localhost:8080/healthcheck,hostname=devstack,service=swift_api</a></td>
<td>API Status</td>
<td>Graph Metric</td>
</tr>
<tr>
<td></td>
<td>http_status</td>
<td>url=<a href="http://localhost:9292,hostname=devstack,service=glance_api">http://localhost:9292,hostname=devstack,service=glance_api</a></td>
<td>API Status</td>
<td>Graph Metric</td>
</tr>
<tr>
<td></td>
<td>disk_space_utilization_perc</td>
<td>device=v-csr-2,hostname=devstack,service=monitoring</td>
<td>Disk Space</td>
<td>Graph Metric</td>
</tr>
<tr>
<td></td>
<td>disk_space_utilization_perc</td>
<td>device=udev,hostname=devstack,service=monitoring</td>
<td>Disk Space</td>
<td>Graph Metric</td>
</tr>
</tbody>
</table>
### List Alarm History

#### Alarm History

<table>
<thead>
<tr>
<th>Name</th>
<th>Old State</th>
<th>New State</th>
<th>Timestamp</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>036cb86b-ad9d-424c-abfc-8541f4a8d466</td>
<td>ALARM</td>
<td>OK</td>
<td>2014-11-04T17:48:30.000Z</td>
<td>The alarm threshold(s) have not been exceeded</td>
</tr>
<tr>
<td>036cb86b-ad9d-424c-abfc-8541f4a8d466</td>
<td>OK</td>
<td>ALARM</td>
<td>2014-11-04T17:44:30.000Z</td>
<td>Thresholds were exceeded for the sub-alarms: [avg(http_status) &gt; 0.0]</td>
</tr>
<tr>
<td>036cb86b-ad9d-424c-abfc-8541f4a8d466</td>
<td>ALARM</td>
<td>OK</td>
<td>2014-11-04T17:42:30.000Z</td>
<td>The alarm threshold(s) have not been exceeded</td>
</tr>
<tr>
<td>036cb86b-ad9d-424c-abfc-8541f4a8d466</td>
<td>OK</td>
<td>ALARM</td>
<td>2014-11-04T17:40:30.000Z</td>
<td>Thresholds were exceeded for the sub-alarms: [avg(http_status) &gt; 0.0]</td>
</tr>
<tr>
<td>036cb86b-ad9d-424c-abfc-8541f4a8d466</td>
<td>ALARM</td>
<td>OK</td>
<td>2014-11-04T17:38:30.000Z</td>
<td>The alarm threshold(s) have not been exceeded</td>
</tr>
<tr>
<td>036cb86b-ad9d-424c-abfc-8541f4a8d466</td>
<td>OK</td>
<td>ALARM</td>
<td>2014-11-04T17:36:30.000Z</td>
<td>Thresholds were exceeded for the sub-alarms: [avg(http_status) &gt; 0.0]</td>
</tr>
<tr>
<td>036cb86b-ad9d-424c-abfc-8541f4a8d466</td>
<td>ALARM</td>
<td>OK</td>
<td>2014-11-04T17:19:30.000Z</td>
<td>The alarm threshold(s) have not been exceeded</td>
</tr>
<tr>
<td>036cb86b-ad9d-424c-abfc-8541f4a8d466</td>
<td>OK</td>
<td>ALARM</td>
<td>2014-11-04T16:50:30.000Z</td>
<td>Thresholds were exceeded for the sub-alarms: [avg(http_status) &gt; 0.0]</td>
</tr>
<tr>
<td>036cb86b-ad9d-424c-abfc-8541f4a8d466</td>
<td>UNDETERMINED</td>
<td>OK</td>
<td>2014-11-04T08:07:45.000Z</td>
<td>The alarm threshold(s) have not been exceeded</td>
</tr>
</tbody>
</table>
Show Alarm Definition
List Notifications

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Address</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roland Hochmuth</td>
<td>EMAIL</td>
<td><a href="mailto:roland.hochmuth@hp.com">roland.hochmuth@hp.com</a></td>
<td>Edit Notification</td>
</tr>
<tr>
<td>Sandy Walsh</td>
<td>EMAIL</td>
<td><a href="mailto:sandy.walsh@rackspace.com">sandy.walsh@rackspace.com</a></td>
<td>Edit Notification</td>
</tr>
<tr>
<td>Tong Li</td>
<td>EMAIL</td>
<td><a href="mailto:litong01@us.ibm.com">litong01@us.ibm.com</a></td>
<td>Edit Notification</td>
</tr>
</tbody>
</table>

Displaying 3 items
Grafana

Openstack Dashboard

Service API Health

CPU

Disk

Database

Network

Monitoring - OpenStack Dashboard

Grafana - OpenStack Dashboard

© Copyright 2014 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice.
Grafana (compute_instance_create_time)