MONASCA

1. Introduction:

Monasca is an open-source monitoring solution built by open-source technologies. It uses an OpenStack authentication service known as a keystone for authentication. It supports multi-tenancy and stores all metrics with tenant ID. Metrics defined using a set of (key, value) pairs called dimensions. Monasca provides several built-in plugins not only for OpenStack but for the tenant and host monitoring as well.

2. Features:

2.1. Monasca is a highly performant, scalable, reliable, and fault-tolerant that scales to service provider metrics levels of metrics throughput.

2.2. It can process 100s of thousands of metrics/sec.

2.3. It offers data retention periods of greater than a year with no data loss while still processing interactive queries.

2.4. Monasca offers Rest APIs for storing and querying metrics and historical information.

2.5. In Monasca, HTTP is the only protocol used which simplifies the overall design and also allows for a much richer way of describing the data via dimensions.

2.6. Real-time thresholding and alarming on metrics.

2.7. Compound alarms described using a simple expressive grammar composed of alarm sub-expressions and logical operators.

2.8. A monitoring agent that supports a number of built-in systems and service checks and also supports Nagios checks and statsd, and support for scraping endpoints as Prometheus does.

2.9. Monasca API can be integrated with other reporting, visualization, or billing systems, such as CloudKitty or Grafana.

3. Architecture:

3.1. **Monasca API:**

A RESTful API for monitoring which is primarily focused on retrieving metrics, statistics, alarm definitions, alarms, and notification methods.

3.2. **Message Queue:**

It is based on Kafka, a high-performance, distributed, fault-tolerant, and scalable message queue with durability built-in. It primarily receives published metrics from the Monitoring API, alarm state transition messages from the Threshold Engine, and log data from the Log API. The data is consumed by other components, such as the Persister, the Notification Engine, and the Log Persister.

3.3. **Persister:**

A Monasca component that consumes metrics and alarm state transitions from the Message Queue and stores them in the Metrics and Alarms Database (InfluxDB).

3.4. **Notification Engine:**

A Monasca component that consumes alarm state transition messages from the Message Queue and sends notifications for alarms, such as emails.

3.5. **Threshold Engine:**
A Monasca component that computes thresholds on metrics and publishes alarms to the Message Queue when they are triggered. The Threshold Engine is based on Apache Storm, a free and open distributed real-time computation system.

3.6. **Metrics and Alarms Database:**
An InfluxDB database used for storing metrics and alarm history.

3.7. **Config Database:**
A MariaDB database used for storing configuration information, alarm definitions, and notification methods.

3.8. **Log API:**
A RESTful API for log management. It gathers log data from the Log Agents and forwards it to the Message Queue.

3.9. **Log Transformer:**
A Logstash component that consumes the log data from the Message Queue, performs transformation and aggregation operations on the data and publishes the data that it creates back to the Message Queue.

3.10. **Log Metrics:**
A Monasca component that consumes log data from the Message Queue, filters the data according to the severity, and generates metrics for specific severities. The generated metrics are published to the Message Queue and can be further processed by the Threshold Engine like any other metrics.

3.11. **Log Persister:**
A Logstash component that consumes the transformed and aggregated log data from the Message Queue and stores it in the Log Database.

3.12. **Log Database:**
An Elasticsearch database for storing the log data.

3.13. **Metrics Agent:**
A Metrics Agent is required for retrieving metrics data from the host on which it runs and sending the metrics data to the Monitoring Service.

3.14. **Log Agent:**
A Log Agent is needed for collecting log data from the host on which it runs and forwarding the log data to the Monitoring Service for further processing. It can be installed on each virtual or physical server from which log data is to be retrieved.

4. Openstack Integration:

Monasca relies on keystone for running and there are requirements about which keystone configuration must exist.

4.1. The endpoint for the API must be registered in keystone as the 'monasca' service.

4.2. The API must have an admin token to use in verifying the keystone tokens it receives.

4.3. For each project which uses Monasca, two users must exist, one will be in the 'monasca-agent' role and be used by the monasca-agent's running on machines. The other should not be in that role and can be used logging into the UI, using the CLI, or for direct queries against the API.