Monitoring Service Operator's Guide

SUSE OpenStack Cloud Monitoring

Monitoring Service Operator's Guide

SUSE OpenStack Cloud Monitoring

Publication Date: 09/08/2022

SUSE LLC
1800 South Novell Place
Provo, UT 84606
USA
https://documentation.suse.com

Contents

About this Manual

This manual describes how SUSE OpenStack Cloud operators can use SUSE OpenStack Cloud Monitoring to monitor OpenStack services and servers.

The manual is structured as follows:

Chapter	Description
Chapter 1, Introduction to SUSE OpenStack Cloud Monitoring	Introduces SUSE OpenStack Cloud Monitoring, its architecture, and users.
Chapter 2, Installation	For installation and configuration of SUSE OpenStack Cloud Monitoring, refer to the *Deployment Guide*, chapter *Deploying the OpenStack Services*, section *Deploying Monasca*.
Chapter 3, Operation and Maintenance	Describes the main operation and maintenance tasks for SUSE OpenStack Cloud Monitoring.
Chapter 4, Monitoring	Describes the basic tasks involved in monitoring services and servers.
Chapter 5, Log Management	Describes the basic tasks involved in managing the log data from the services and servers.
Appendix B, Glossary	Defines the central terms relevant for SUSE OpenStack Cloud Monitoring.

1 Intended Audience

This manual is intended for operators who install, operate, and maintain SUSE OpenStack Cloud Monitoring. It also describes how the operator can use SUSE OpenStack Cloud Monitoring for monitoring.

The manual assumes that you have expert knowledge of OpenStack and SUSE OpenStack Cloud Monitoring, especially the individual services comprising SUSE OpenStack Cloud Monitoring.

2 Notational Conventions

This manual uses the following notational conventions:

Add	Names of graphical user interface elements.
`init`	System names, for example command names and text that is entered from the keyboard.
`<variable>`	Variables for which values must be entered.
`[option]`	Optional items, for example optional command parameters.
`one | two`	Alternative entries.
`{one | two}`	Mandatory entries with alternatives.

3 Abbreviations

This manual uses the following abbreviations:

IaaS. Infrastructure as a Service

ICMP. Internet Control Message Protocol

OS. Operating System

OSS. Open Source Software

PaaS. Platform as a Service

SaaS. Software as a Service

4 Available Documentation

The following documentation on SUSE OpenStack Cloud Monitoring is available:

- *Overview:* A manual introducing SUSE OpenStack Cloud Monitoring. It is written for everybody interested in SUSE OpenStack Cloud Monitoring.
- *OpenStack Operator's Guide:* A manual for SUSE OpenStack Cloud operators describing how to prepare their OpenStack platform for SUSE OpenStack Cloud Monitoring. The manual also describes how the operators use SUSE OpenStack Cloud Monitoring for monitoring their OpenStack services.
- *Monitoring Service Operator's Guide:* A manual for system operators describing how to operate SUSE OpenStack Cloud Monitoring. The manual also describes how the operators use SUSE OpenStack Cloud Monitoring for monitoring their environment.

5 Related Web References

The following Web references provide information on open source offerings integrated with SUSE OpenStack Cloud Monitoring:

- OpenStack (http://docs.openstack.org/newton/): Documentation on OpenStack, the underlying platform technology.
- OpenStack Horizon (http://docs.openstack.org/developer/horizon/): Documentation on the OpenStack Horizon dashboard.
- Monasca Wiki (https://wiki.openstack.org/wiki/Monasca): Information on Monasca, the core of SUSE OpenStack Cloud Monitoring.

More detailed Web references provided in this manual are subject to change without notice.

6 Copyright

ware distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

For SUSE trademarks, see http://www.suse.com/company/legal/. All other third-party trademarks are the property of their respective owners. Trademark symbols (®, ™ etc.) denote trademarks of SUSE and its affiliates. Asterisks (*) denote third-party trademarks.

The OpenStack® Word Mark and OpenStack logo are registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation in the United States and other countries and are used with the OpenStack Foundation's permission.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither SUSE LLC, its affiliates, the authors nor the translators shall be held liable for possible errors or the consequences thereof.

7 Export Restrictions

Exportation/release of this document may require necessary procedures in accordance with the regulations of your resident country and/or US export control laws.

1 Introduction to SUSE OpenStack Cloud Monitoring

As more and more applications are deployed on cloud systems and cloud systems are growing in complexity, managing the cloud infrastructure is becoming increasingly difficult. SUSE OpenStack Cloud Monitoring helps mastering this challenge by providing a sophisticated Monitoring as a Service solution that is operated on top of OpenStack-based cloud computing platforms.

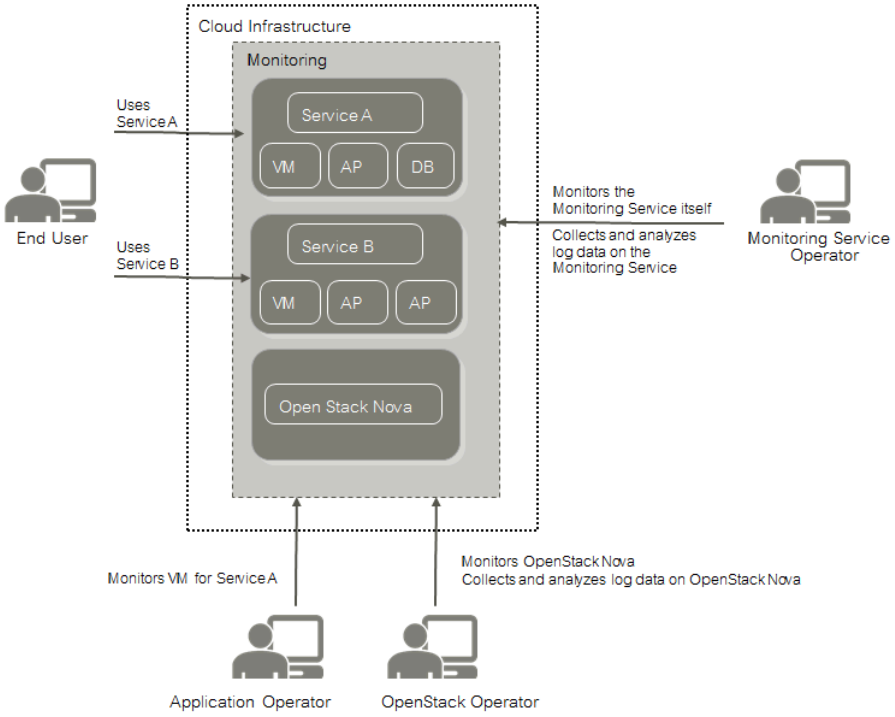
The component architecture of OpenStack provides for high flexibility, yet it increases the burden of system operation because multiple services must be handled. SUSE OpenStack Cloud Monitoring offers an integrated view of all services and assembles and presents related metrics and log data in one convenient access point. While being flexible and scalable to instantly reflect changes in the OpenStack platform, SUSE OpenStack Cloud Monitoring provides the ways and means required to ensure multi-tenancy, high availability, and data security.

SUSE OpenStack Cloud Monitoring covers all aspects of a Monitoring as a Service solution:

- Central management of monitoring and log data from medium and large-size OpenStack deployments.
- Storage of metrics and log data in a resilient way.
- Multi-tenancy architecture to ensure the secure isolation of metrics and log data.
- Horizontal and vertical scalability to support constantly evolving cloud infrastructures. When physical and virtual servers are scaled up or down to varying loads, the monitoring and log management solution can be adapted accordingly.

1.1 Basic Usage Scenario

The basic usage scenario of setting up and using the monitoring features of SUSE OpenStack Cloud Monitoring looks as follows:



An **application operator** acts as a service provider in the OpenStack environment. He books virtual machines to provide services to **end users** or to host services that he needs for his own development activities. SUSE OpenStack Cloud Monitoring helps application operators ensure that their services and the servers on which they are provided are configured and working as required.

The **OpenStack operator** is a special application operator who is responsible for administrating and maintaining the underlying OpenStack platform. The monitoring and log management services of SUSE OpenStack Cloud Monitoring enable him to ensure the availability and quality of the platform. He uses SUSE OpenStack Cloud Monitoring for:

- Monitoring physical and virtual servers, hypervisors, and OpenStack services.
- Monitoring middleware components, for example, database services.
- Retrieving and analyzing the log data of the OpenStack services and servers, the middleware components, and the operating system.

As the **Monitoring Service operator**, you are responsible for providing the monitoring and log management features to the application operators and the OpenStack operator. This enables the application operators and the OpenStack operator to focus on operation and the quality of their services and servers without having to carry out the tedious tasks implied by setting up and administrating their own monitoring software. You use the monitoring features yourself for ensuring the quality of SUSE OpenStack Cloud Monitoring.

The Monitoring Service Operator's Tasks

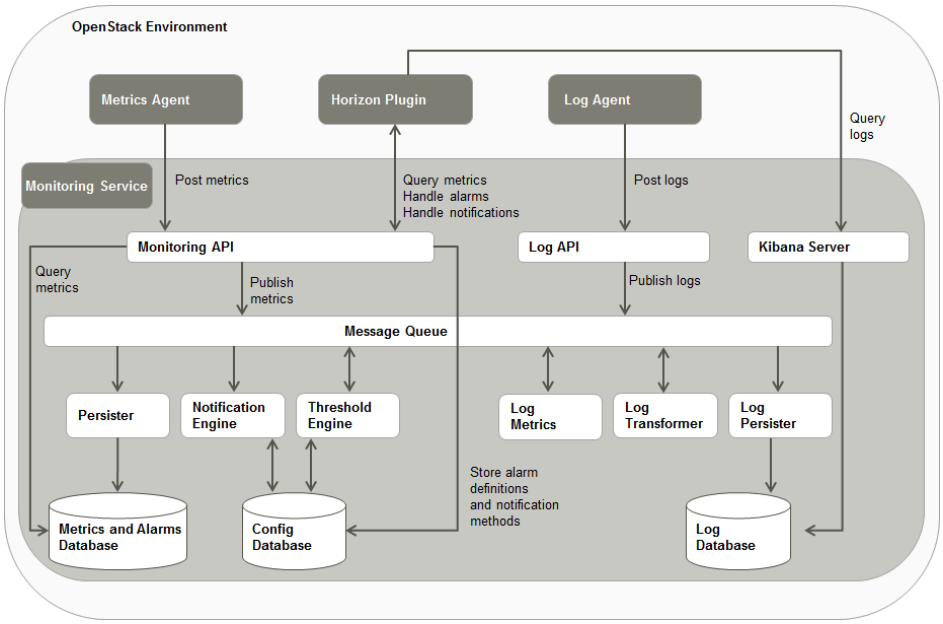
As the Monitoring Service operator, you have the following responsibilities:

- Deploying the Monitoring Service, thus providing the monitoring features to the application operators, and the monitoring and log management features to the OpenStack operator.
- Regular maintenance of the components and services SUSE OpenStack Cloud Monitoring consists of.
- Backup of the SUSE OpenStack Cloud Monitoring databases, configuration files, and customized dashboards.
- Monitoring of SUSE OpenStack Cloud Monitoring to ensure the SUSE OpenStack Cloud Monitoring quality. For monitoring, you use a graphical user interface that is seamlessly integrated into the cloud infrastructure. Based on OpenStack Horizon, the user interface visualizes the health and status of your cloud resources and enables user access to all monitoring and log management functionality.

The tasks of the Monitoring Service operator and the OpenStack Operator can jointly be performed by one system operator. In this case, also refer to the *OpenStack Operator's Guide*.

1.2 SUSE OpenStack Cloud Monitoring Architecture and Components

The following illustration provides an overview of the main components of SUSE OpenStack Cloud Monitoring and their interaction:



OpenStack

SUSE OpenStack Cloud Monitoring relies on OpenStack as technology for building cloud computing platforms for public and private clouds. OpenStack consists of a series of interrelated projects delivering various components for a cloud infrastructure solution and allowing for the deployment and management of Infrastructure as a Service (IaaS) platforms.

Monitoring Service

The Monitoring Service is the central SUSE OpenStack Cloud Monitoring component. It is responsible for receiving, persisting, and processing monitoring and log data, as well as providing the data to the users.

The Monitoring Service relies on Monasca, an open source Monitoring as a Service solution. It uses Monasca for high-speed metrics querying and integrates the Threshold Engine (streaming alarm engine) and the Notification Engine of Monasca.

The Monitoring Service consists of the following components:

Monitoring API

A RESTful API for monitoring. It is primarily focused on the following areas:

- Metrics: Store and query massive amounts of metrics in real time.
- Statistics: Provide statistics for metrics.
- Alarm Definitions: Create, update, query, and delete alarm definitions.
- Alarms: Query and delete the alarm history.
- Notification Methods: Create and delete notification methods and associate them with alarms. Users can be notified directly when alarms are triggered, for example, via email.

Message Queue

A component that 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. The Message Queue is also used to publish and consume other events in the system. It is based on Kafka, a high-performance, distributed, fault-tolerant, and scalable message queue with durability built-in. For administrating the Message Queue, SUSE OpenStack Cloud Monitoring uses Zookeeper, a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services.

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).

Notification Engine

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

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.

Metrics and Alarms Database

An InfluxDB database used for storing metrics and the alarm history.

Config Database

A MariaDB database used for storing configuration information, alarm definitions, and notification methods.

Log API

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

The SUSE OpenStack Cloud Monitoring log management is based on Logstash, a tool for receiving, processing, and publishing all kinds of logs. It provides a powerful pipeline for querying and analyzing logs. Elasticsearch is used as the back-end datastore, and Kibana as the front-end tool for retrieving and visualizing the log data.

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.

Log Metrics

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

Log Persister

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

Kibana Server

A Web browser-based analytics and search interface to the Log Database.

Log Database

An Elasticsearch database for storing the log data.

Horizon Plugin

SUSE OpenStack Cloud Monitoring comes with a plugin for the OpenStack Horizon dashboard. The plugin extends the main dashboard in OpenStack with a view for monitoring. This enables SUSE OpenStack Cloud Monitoring users to access the monitoring functions from a central Web-based graphical user interface. For details, refer to the OpenStack Horizon documentation (http://docs.openstack.org/developer/horizon/).

Based on OpenStack Horizon, the monitoring data is visualized on a comfortable and easy-to-use dashboard which fully integrates with the following applications:

Grafana (for metrics data)

An open source application for visualizing large-scale measurement data.

Kibana (for log data)

An open source analytics and visualization platform designed to work with Elasticsearch.

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. The agent supports metrics from a variety of sources as well as a number of built-in system and service checks.

A Metrics Agent can be installed on each virtual or physical server to be monitored.

The agent functionality is fully integrated into the source code base of the Monasca project. For details, refer to the Monasca Wiki (https://wiki.openstack.org/wiki/Monasca).

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.

The agent functionality is fully integrated into the source code base of the Monasca project. For details, refer to the Monasca Wiki (https://wiki.openstack.org/wiki/Monasca).

1.2.1 Agents and Services

Service Name	Description
`zookeeper`	Centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services.
`storm-nimbus`	Storm is a distributed real-time computation system for processing large volumes of high-velocity data. The Storm Nimbus daemon is responsible for distributing code around a cluster, assigning tasks to machines, and monitoring for failures.
`storm-supervisor`	The Storm supervisor listens for work assigned to its machine and starts and stops worker processes as necessary based on what Nimbus has assigned to it.
`mariadb`	MariaDB database service. SUSE OpenStack Cloud Monitoring stores configuration information in this database.
`kafka`	Message queue service.
`influxdb`	InfluxDB database service. SUSE OpenStack Cloud Monitoring stores metrics and alarms in this database.
`elasticsearch`	Elasticsearch database service. SUSE OpenStack Cloud Monitoring stores the log data in this database.
`memcached`	Memcached service. SUSE OpenStack Cloud Monitoring uses it for caching authentication and authorization information required for the communication between the Log API and OpenStack Keystone.
`openstack-monasca-notification`	Notification Engine.

Service Name	Description
`openstack-monasca-thresh`	Threshold Engine.
`openstack-monasca-log-transformer`	Log Transformer.
`apache2`	Log and monitoring API.
`openstack-monasca-persister`	Persister.
`openstack-monasca-agent`	Metrics Agent.
`kibana`	Kibana server.
`openstack-monasca-log-persister`	Log Persister.
`openstack-monasca-log-metrics`	Log Metrics.
`openstack-monasca-log-agent`	Log Agent.

1.3 User Management

SUSE OpenStack Cloud Monitoring is fully integrated with Keystone, the identity service which serves as the common authentication and authorization system in OpenStack.

The SUSE OpenStack Cloud Monitoring integration with Keystone requires any SUSE OpenStack Cloud Monitoring user, including the Monitoring Service operator, to be registered as an OpenStack user. All authentication and authorization in SUSE OpenStack Cloud Monitoring is done through Keystone. If a user requests monitoring data, for example, SUSE OpenStack Cloud Monitoring verifies that the user is a valid user in OpenStack and allowed to access the requested metrics.

SUSE OpenStack Cloud Monitoring users are created and administrated in OpenStack:

- Each user assumes a role in OpenStack to perform a specific set of operations. The OpenStack role specifies a set of rights and privileges.
- Each user is assigned to at least one project in OpenStack. A project is an organizational unit that defines a set of resources which can be accessed by the assigned users.
  Application operators in SUSE OpenStack Cloud Monitoring can monitor the set of resources that is defined for the projects to which they are assigned.

For details on user management, refer to the OpenStack documentation (http://docs.openstack.org/newton/).

1.4 Distribution Media

SUSE OpenStack Cloud Monitoring is shipped as part of SUSE OpenStack Cloud with a maintenance update. Note that you need to buy a separate subscription for SUSE OpenStack Cloud Monitoring to be entitled to get support for SUSE OpenStack Cloud Monitoring.

2 Installation

SUSE OpenStack Cloud Monitoring is automatically installed and configured if you deploy the Monasca barclamp. For details, see https://documentation.suse.com/soc/8/single-html/suse-openstack-cloud-deployment/#sec-depl-ostack-monasca.

3 Operation and Maintenance

Regular operation and maintenance includes:

- Configuring data retention for the InfluxDB database. This can be configured in the Monasca barclamp. For details, see the *SUSE OpenStack Cloud Deployment Guide*.
- Configuring data retention for the Elasticsearch database. This can be configured in the Monasca barclamp. For details, see the *SUSE OpenStack Cloud Deployment Guide*.
- Removing metrics data from the InfluxDB database.
- Removing log data from the Elasticsearch database.
- Handling log files of agents and services.
- Backup and recovery of databases, configuration files, and dashboards.

3.1 Removing Metrics Data

Metrics data is stored in the Metrics and Alarms InfluxDB Database. InfluxDB features an SQL-like query language for querying data and performing aggregations on that data.

The Metrics Agent configuration defines the metrics and types of measurement for which data is stored. For each measurement, a so-called series is written to the InfluxDB database. A series consists of a timestamp, the metrics, and the value measured.

Every series can be assigned key tags. In the case of SUSE OpenStack Cloud Monitoring, this is the `_tenant_id` tag. This tag identifies the OpenStack project for which the metrics data has been collected.

From time to time, you may want to delete outdated or unnecessary metrics data from the Metrics and Alarms Database, for example, to save space or remove data for metrics you are no longer interested in. To delete data, you use the InfluxDB command line interface, the interactive shell that is provided for the InfluxDB database.

Proceed as follows to delete metrics data from the database:

1. Create a backup of the database. For details, refer to *Section 3.4, "Backup and Recovery"*.
2. Determine the ID of the OpenStack project for the data to be deleted:
   Log in to the OpenStack dashboard and go to **Identity** > **Projects**. The `monasca` project initially provides all metrics data related to SUSE OpenStack Cloud Monitoring.

In the course of the productive operation of SUSE OpenStack Cloud Monitoring, additional projects may be created, for example, for application operators.
The **Project ID** field shows the relevant tenant ID.

3. Log in to the host where the Monitoring Service is installed.

4. Go to the directory where InfluxDB is installed:

```
cd /usr/bin
```

5. Connect to InfluxDB using the InfluxDB command line interface as follows:

```
./influx -host <host_ip>
```

Replace `<host_ip>` with the IP address of the machine on which SUSE OpenStack Cloud Monitoring is installed.
The output of this command is, for example, as follows:

```
Connected to http://localhost:8086 version 1.1.1
InfluxDB shell version: 1.1.1
```

6. Connect to the InfluxDB database of SUSE OpenStack Cloud Monitoring (`mon`):

```
> show databases
name: databases
name
----
mon
_internal

> use mon
Using database mon
```

7. Check the outdated or unnecessary data to be deleted.

   - You can view all measurements for a specific project as follows:

     ```
     SHOW MEASUREMENTS WHERE _tenant_id = '<project ID>'
     ```

   - You can view the series for a specific metrics and project, for example, as follows:

     ```
     SHOW SERIES FROM "cpu.user_perc" WHERE _tenant_id = '<project ID>'
     ```

8. Delete the desired data.

- When a project is no longer relevant or a specific tenant is no longer used, delete all series for the project as follows:

  ```
  DROP SERIES WHERE _tenant_id = '<project ID>'
  ```

  Example:

  ```
  DROP SERIES WHERE _tenant_id = '27620d7ee6e948e29172f1d0950bd6f4'
  ```

- When a metrics is no longer relevant for a project, delete all series for the specific project and metrics as follows:

  ```
  DROP SERIES FROM "<metrics>" WHERE _tenant_id = '<project ID>'
  ```

  Example:

  ```
  DROP SERIES FROM "cpu.user_perc" WHERE _tenant_id = '27620d7e'
  ```

9. Restart the `influxdb` service, for example, as follows:

   ```
   sudo systemctl restart influxdb
   ```

3.2 Removing Log Data

Log data is stored in the Elasticsearch database. Elasticsearch stores the data in indices. One index per day is created for every OpenStack project.

By default, the indices are stored in the following directory on the host where the Monitoring Service is installed:

`/var/data/elasticsearch/<cluster-name>/nodes/<node-name>`

Example:

`/var/data/elasticsearch/elasticsearch/nodes/0`

Note

If your system uses a different directory, look up the `path.data` parameter in the Elasticsearch configuration file, `/etc/elasticsearch/elasticsearch.yml`.

If you want to delete outdated or unnecessary log data from the Elasticsearch database, proceed as follows:

1. Make sure that `curl` is installed. If this is not the case, install the package with

   ```
   sudo zypper in curl
   ```

2. Create a backup of the Elasticsearch database. For details, refer to *Section 3.4, "Backup and Recovery"*.

3. Determine the ID of the OpenStack project for the data to be deleted:
   Log in to the OpenStack dashboard and go to **Identity** > **Projects**. The `monasca` project initially provides a ll metrics data related to SUSE OpenStack Cloud Monitoring.
   In the course of the productive operation of SUSE OpenStack Cloud Monitoring, additional projects may be created.
   The **Project ID** field shows the relevant ID.

4. Log in to the host where the Monitoring Service is installed.

5. Make sure that the data you want to delete exists by executing the following command:

   ```
   curl -XHEAD -i 'http://localhost:<port>/<projectID-date>'
   ```

   For example, if Elasticsearch is listening at port 9200 (default), the ID of the OpenStack project is `abc123`, and you want to check the index of 2015, July 1st, the command is as follows:

   ```
   curl -XHEAD -i 'http://localhost:9200/abc123-2015-07-01'
   ```

   If the HTTP response is `200`, the index exists; if the response is `404`, it does not exist.

6. Delete the index as follows:

   ```
   curl -XDELETE -i 'http://localhost:<port>/<projectID-date>'
   ```

   Example:

   ```
   curl -XDELETE -i 'http://localhost:9200/abc123-2015-07-01'
   ```

   This command either returns an error, such as `IndexMissingException`, or acknowledges the successful deletion of the index.

Note

Be aware that the `-XDELETE` command immediately deletes the index file!

Both, for `-XHEAD` and `-XDELETE`, you can use wildcards for processing several indices. For example, you can delete all indices of a specific project for the whole month of July, 2015:

```
curl -XDELETE -i 'http://localhost:9200/abc123-2015-07-*'
```

Note

Take extreme care when using wildcards for the deletion of indices. You could delete all existing indices with one single command!

3.3 Log File Handling

In case of trouble with the SUSE OpenStack Cloud Monitoring services, you can study their log files to find the reason. The log files are also useful if you need to contact your support organization. For storing the log files, the default installation uses the `/var/log` directory on the hosts where the agents or services are installed.

You can use `systemd`, a system and session manager for LINUX, and `journald`, a LINUX logging interface, for addressing dispersed log files.

The SUSE OpenStack Cloud Monitoring installer automatically puts all SUSE OpenStack Cloud Monitoring services under the control of `systemd`. `journald` provides a centralized management solution for the logging of all processes that are controlled by `systemd`. The logs are collected and managed in a so-called journal controlled by the `journald` daemon.

For details on the `systemd` and `journald` utilities, refer to the https://documentation.suse.com/sles/12-SP5/single-html/SLES-admin/#part-system.

3.4 Backup and Recovery

Typical tasks of the Monitoring Service operator are to make regular backups, particularly of the data created during operation.

At regular intervals, you should make a backup of all:

- Databases.
- Configuration files of the individual agents and services.
- Monitoring and log dashboards you have created and saved.

SUSE OpenStack Cloud Monitoring does not offer integrated backup and recovery mechanisms. Instead, use the mechanisms and procedures of the individual components.

3.4.1 Databases

You need to create regular backups of the following databases on the host where the Monitoring Service is installed:

- Elasticsearch database for historic log data.
- InfluxDB database for historic metrics data.
- MariaDB database for historic configuration information.

It is recommended that backup and restore operations for databases are carried out by experienced operators only.

Preparations

Before backing up and restoring a database, we recommend stopping the Monitoring API and the Log API on the `monasca-server` node, and check that all data is processed. This ensures that no data is written to a database during a backup and restore operation. After backing up and restoring a database, restart the APIs.

To stop the Monitoring API and the Log API, use the following command:

```
systemctl stop apache2
```

To check that all Kafka queues are empty, list the existing consumer groups and check the LAG column for each group. It should be 0. For example:

```
kafka-consumer-groups.sh --zookeeper 192.168.56.81:2181 --list
kafka-consumer-groups.sh --zookeeper 192.168.56.81:2181 --describe \
 --group 1_metrics | column -t -s ','
kafka-consumer-groups.sh --zookeeper 192.168.56.81:2181 --describe \
```

```
 --group transformer-logstash-consumer | column -t -s ','
kafka-consumer-groups.sh --zookeeper 192.168.56.81:2181 --describe \
 --group thresh-metric | column -t -s ','
```

To restart the Monitoring API and the Log API, use the following command:

```
systemctl start apache2
```

Elasticsearch Database

For backing up and restoring your Elasticsearch database, use the Snapshot and Restore module of Elasticsearch.

To create a backup of the database, proceed as follows:

1. Make sure that **`curl`** is installed, **`zypper in curl`**.
2. Log in to the host where the Monitoring Service is installed.
3. Create a snapshot repository. You need the Elasticsearch bind address for all commands. run **`grep network.bind_host /etc/elasticsearch/elasticsearch.yml`** to find the bind address, and replace *`IP`* in the following commands with this address. For example:

   ```
   curl -XPUT http://IP:9200/_snapshot/my_backup -d '{
     "type": "fs",
     "settings": {
          "location": "/mount/backup/elasticsearch1/my_backup",
          "compress": true
     }
   }'
   ```

   The example registers a shared file system repository (`"type": "fs"`) that uses the `/mount/backup/elasticsearch1` directory for storing snapshots.

Note

The directory for storing snapshots must be configured in the `elasticsearch/repo_dir` setting in the Monasca barclamp (see the Deployment Guide (https://documentation.suse.com/soc/8/single-html/suse-openstack-cloud-deployment/#sec-depl-ostack-monasca)↗). The directory must be manually mounted before creating the snapshot. The `elasticsearch` user must be specified as the owner of the directory.

`compress` is turned on to compress the metadata files.

4. Check whether the repository was created successfully:

```
curl -XGET http://IP:9200/_snapshot/my_backup
```

This example response shows a successfully created repository:

```
{
  "my_backup": {
    "type": "fs",
    "settings": {
      "compress": "true",
      "location": "/mount/backup/elasticsearch1/my_backup"
    }
  }
}
```

5. Create a snapshot of your database that contains all indices. A repository can contain multiple snapshots of the same database. The name of a snapshot must be unique within the snapshots created for your database, for example:

```
curl -XPUT http://IP:9200/_snapshot/my_backup/snapshot_1?wait_for_completion=true
```

The example creates a snapshot named `snapshot_1` for all indices in the `my_backup` repository.

To restore the database instance, proceed as follows:

1. Close all indices of your database, for example:

```
curl -XPOST http://IP:9200/_all/_close
```

2. Restore all indices from the snapshot you have created, for example:

```
curl -XPOST http://IP:9200/_snapshot/my_backup/snapshot_1/_restore
```

The example restores all indices from `snapshot_1` that is stored in the `my_backup` repository.

For additional information on backing up and restoring an Elasticsearch database, refer to the Elasticsearch documentation (https://www.elastic.co/guide/en/elasticsearch/reference/2.3/modules-snapshots.html).

InfluxDB Database

For backing up and restoring your InfluxDB database, you can use the InfluxDB shell. The shell is part of your InfluxDB distribution. If you installed InfluxDB via a package manager, the shell is, by default, installed in the `/usr/bin` directory.

To create a backup of the database, proceed as follows:

1. Log in to the InfluxDB database as a user who is allowed to run the `influxdb` service, for example:

   ```
   su influxdb -s /bin/bash
   ```

2. Back up the database, for example:

   ```
   influxd backup -database mon /mount/backup/mysnapshot
   ```

   Monasca is using `mon` as the name of the database The example creates the backup for the database in `/mount/backup/mysnapshot`.

Before restoring the database, make sure that all database processes are shut down. To restore the database, you can then proceed as follows:

1. If required, delete all files not included in the backup by dropping the database before you carry out the restore operation. A restore operation restores all files included in the backup. Files created or merged at a later point in time are not affected. For example:

   ```
   influx -host IP -execute 'drop database mon;'
   ```

   Replace *IP* with the IP address that the database is listening to. You can run **`influxd config`** and look up the IP address in the `[http]` section.

2. Stop the InfluxDB database service:

   ```
   systemctl stop influxdb
   ```

3. Log in to the InfluxDB database as a user who is allowed to run the `influxdb` service:

   ```
   su influxdb -s /bin/bash
   ```

4. Restore the metastore:

   ```
   influxd restore -metadir /var/opt/influxdb/meta /mount/backup/mysnapshot
   ```

5. Restore the database, for example:

```
influxd restore -database mon -datadir /var/opt/influxdb/data /mount/backup/
mysnapshot
```

   The example restores the backup from `/mount/backup/mysnapshot` to `/var/opt/influxdb/influxdb.conf`.

6. Ensure that the file permissions for the restored database are set correctly:

```
chown -R influxdb:influxdb /var/opt/influxdb
```

7. Start the InfluxDB database service:

```
systemctl start influxdb
```

For additional information on backing up and restoring an InfluxDB database, refer to the InfluxDB documentation (https://docs.influxdata.com/influxdb/v1.1/administration/backup_and_restore/).

MariaDB Database

For backing up and restoring your MariaDB database, you can use the **`mysqldump`** utility program. **`mysqldump`** performs a logical backup that produces a set of SQL statements. These statements can later be executed to restore the database.

To back up your MariaDB database, you must be the owner of the database or a user with superuser privileges, for example:

```
mysqldump -u root -p mon > dumpfile.sql
```

In addition to the name of the database, you have to specify the name and the location where **`mysqldump`** stores its output.

To restore your MariaDB database, proceed as follows:

1. Log in to the host where the Monitoring Service is installed as a user with root privileges.

2. Make sure that the **`mariadb`** service is running:

```
systemctl start mariadb
```

3. Log in to the database you have backed up as a user with root privileges, for example:

```
mysql -u root -p mon
```

4. Remove and then re-create the database:

```
DROP DATABASE mon;
CREATE DATABASE mon;
```

5. Exit mariadb:

```
\q
```

6. Restore the database, for example:

```
mysql -u root -p mon < dumpfile.sql
```

For additional information on backing up and restoring a MariaDB database with `mysqldump`, refer to the MariaDB documentation (https://mariadb.com/kb/en/mariadb/mysqldump/).

3.4.2 Configuration Files

Below you find a list of the configuration files of the agents and the individual services included in the Monitoring Service. Back up these files at least after you have installed and configured SUSE OpenStack Cloud Monitoring and after each change in the configuration.

```
/etc/influxdb/influxdb.conf
/etc/kafka/server.properties
/etc/my.cnf
/etc/my.cnf.d/client.cnf
/etc/my.cnf.d/mysql-clients.cnf
/etc/my.cnf.d/server.cnf
/etc/monasca/agent/agent.yaml
/etc/monasca/agent/conf.d/*
/etc/monasca/agent/supervisor.conf
/etc/monasca/api-config.conf
/etc/monasca/log-api-config.conf
/etc/monasca/log-api-config.ini
/etc/monasca-log-persister/monasca-log-persister.conf
/etc/monasca-log-transformer/monasca-log-transformer.conf
/etc/monasca-log-agent/agent.conf
/etc/monasca-notification/monasca-notification.yaml
/etc/monasca-persister/monasca-persister.yaml
```

```
/etc/monasca-thresh/thresh.yaml
/etc/elasticsearch/elasticsearch.yml
/etc/elasticsearch/logging.yml
/etc/kibana/kibana.yml
```

Recovery

If you need to recover the configuration of one or more agents or services, the recommended procedure is as follows:

1. If necessary, uninstall the agents or services, and install them again.
2. Stop the agents or services.
3. Copy the backup of your configuration files to the correct location according to the table above.
4. Start the agents or services again.

3.4.3 Dashboards

Kibana can persist customized log dashboard designs to the Elasticsearch database, and allows you to recall them. For details on saving, loading, and sharing log management dashboards, refer to the Kibana documentation (https://www.elastic.co/guide/en/kibana/4.5/dashboard.html#saving-dashboards).

Grafana allows you to export a monitoring dashboard to a JSON file, and to re-import it when necessary. For backing up and restoring the exported dashboards, use the standard mechanisms of your file system. For details on exporting monitoring dashboards, refer to the Getting Started (https://grafana.com/docs/guides/getting_started/) tutorial of Grafana.

4 Monitoring

SUSE OpenStack Cloud Monitoring offers various features which support you in proactively managing your cloud resources. A large number of metrics in combination with early warnings about problems and outages assists you in analyzing and troubleshooting any issue you encounter in your environment.

The monitoring features include:

- A monitoring overview which allows you to access all monitoring information.
- Metrics dashboards for visualizing your monitoring data.
- Alerting features for monitoring.

In the following sections, you will find information on the monitoring overview and the metrics dashboards as well as details on how to define and handle alarms and notifications.

Accessing SUSE OpenStack Cloud Monitoring

For accessing SUSE OpenStack Cloud Monitoring and performing monitoring tasks, you must have access to the OpenStack platform as a user with the *monasca-user* or *monasca-read-only-user* role in the *monasca* tenant.

Log in to OpenStack Horizon with your user name and password. The functions you can use in OpenStack Horizon depend on your access permissions. To access logs and metrics, switch to the *monasca* tenant in Horizon. This allows you to access all monitoring data for SUSE OpenStack Cloud Monitoring.

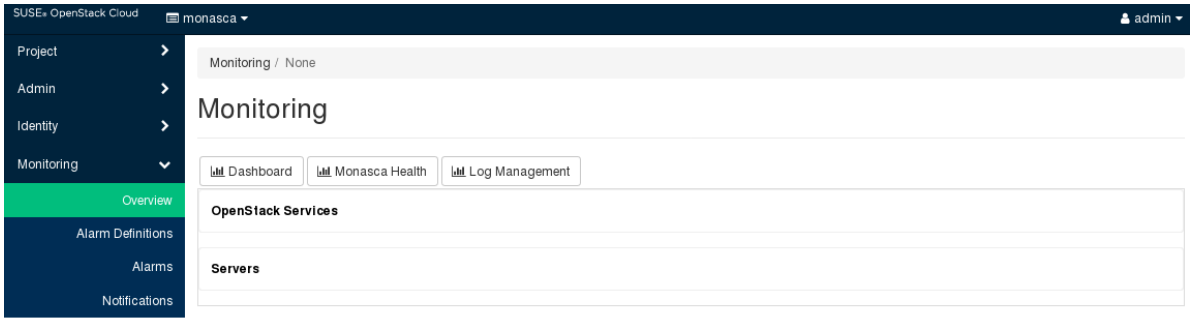


FIGURE 4.1: SUSE OPENSTACK CLOUD HORIZON DASHBOARD—MONITORING

4.1 Overview

SUSE OpenStack Cloud Monitoring provides one convenient access point to your monitoring data. Use *Monitoring* > *Overview* to keep track of your services and servers and quickly check their status. The overview also indicates any irregularities in the log data of the system components you are monitoring.

On the *Overview* page, you can:

- View the status of your services, servers, and log data at a glance. As soon as you have defined an alarm for a service, a server, or log data and metrics data has been received, there is status information displayed on the *Overview* page. Different colors are used for the different statuses.
  For details on the status information, refer to *Section 4.5, "Status of Services, Servers, and Log Data"*. For details on defining alarms, refer to *Section 4.3, "Defining Alarms"*.
- Access a preconfigured dashboard that visualizes your metrics data. Starting from this dashboard, you can create your own dashboards for visualizing your metrics data as required. For details, refer to *Section 4.2, "Working with Data Visualizations"*.
- Access the SUSE OpenStack Cloud Monitoring log management functionality. For details, refer to *Chapter 5, Log Management*.

4.2 Working with Data Visualizations

The user interface for monitoring your services, servers, and log data integrates with Grafana, an open source application for visualizing large-scale monitoring data. Use the options at the top border of the *Overview* page to access Grafana.

SUSE OpenStack Cloud Monitoring ships with preconfigured metrics dashboards. You can instantly use them for monitoring your environment. You can also use them as a starting point for building your own dashboards.

Preconfigured Metrics Dashboard for SUSE OpenStack Cloud Monitoring

As a Monitoring Service operator, you use the *Monasca Health* option on the *Overview* page to view the metrics data on the Monitoring Service. The OpenStack operator uses the *Dashboard* option to view the metrics data on the OpenStack services.

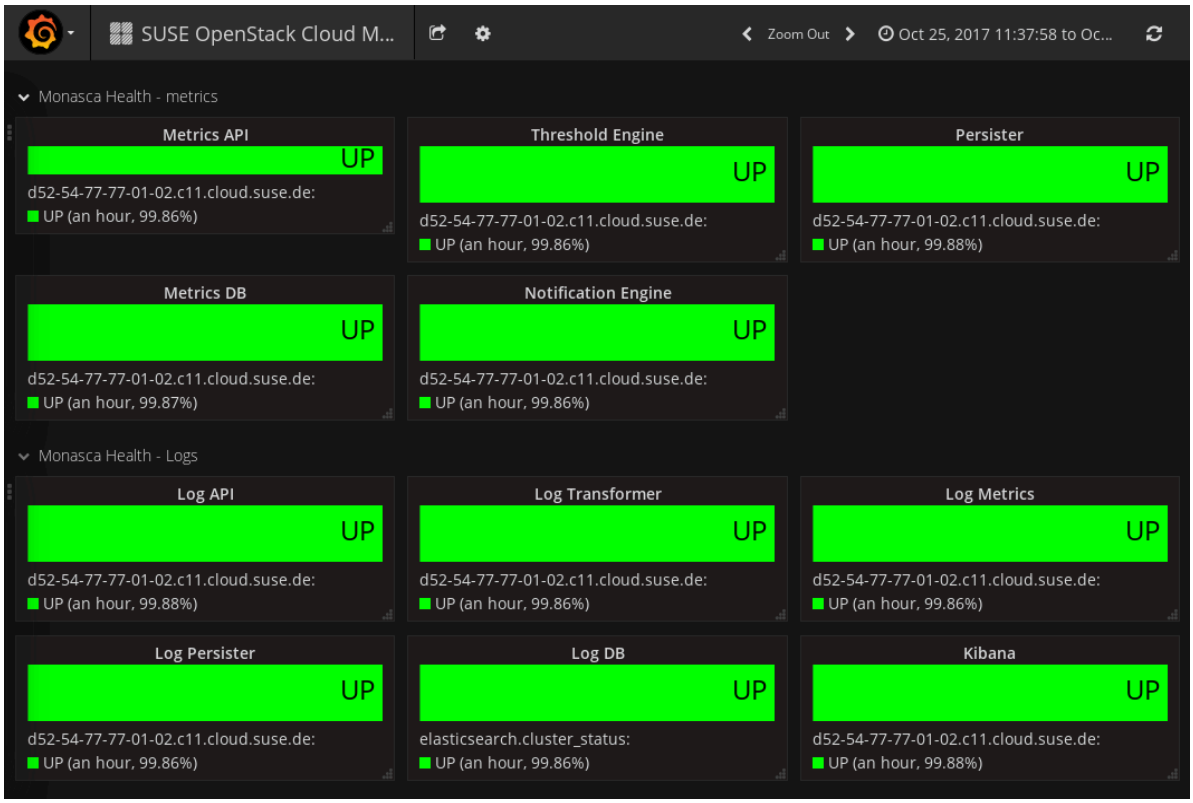


FIGURE 4.2: METRICS DASHBOARD—MONITORING SERVICE OPERATOR'S VIEW

The preconfigured dashboard shows the following data on the Monitoring Service:

- Status of the main SUSE OpenStack Cloud Monitoring components (`UP` or `DOWN`).
  In the upper part of the dashboard the components are grouped into metrics and log management components as well as common components that are used by metrics-based and log-based monitoring.

- Information on system resources.
  The dashboard shows metrics data on CPU usage: the percentage of time the CPU is used in total (`cpu.percent`), at user level (`cpu.user_perc`), and at system level (`cpu.system_perc`), as well as the percentage of time the CPU is idle when no I/O requests are in progress (`cpu.wait_perc`).
  The dashboard shows metrics data on memory usage: the number of megabytes of total memory (`mem.total_mb`), used memory (`mem.used_mb`), total swap memory (`mem.swap_total_mb`), and used swap memory (`mem.swap_used_mb`), as well as the number of megabytes used for the page cache (`mem.used_cache`).
  The dashboard visualizes metrics on the percentage of disk space that is being used on a device (`disk.space_used_perc`).

The dashboard shows metrics data on the SUSE OpenStack Cloud Monitoring system load over different periods (`load.avg_1_min`, `load.avg_5_min`, and `load.avg_15_min`).

- The network usage of SUSE OpenStack Cloud Monitoring.
  The dashboard shows the number of network bytes received and sent per second (`net.in_bytes_sec` and `net.out_bytes_sec`).

- Metrics data on each SUSE OpenStack Cloud Monitoring component. The metrics that are visualized differ slightly from component to component. The dashboard shows, for example, the percentage of CPU that is consumed by a component (`process.cpu_perc.value`), the amount of physical memory that is allocated to the component (`process.mem.rss_mbytes.value`), or the number of processes that exist with the corresponding component name (`process.pid_count.value`).

Building Dashboards

Each metrics dashboard is composed of one or more panels that are arranged in one or more rows. A row serves as a logical divider within a dashboard. It organizes your panels in groups. The panel is the basic building block for visualizing your metrics data.

For building dashboards, you have two options:

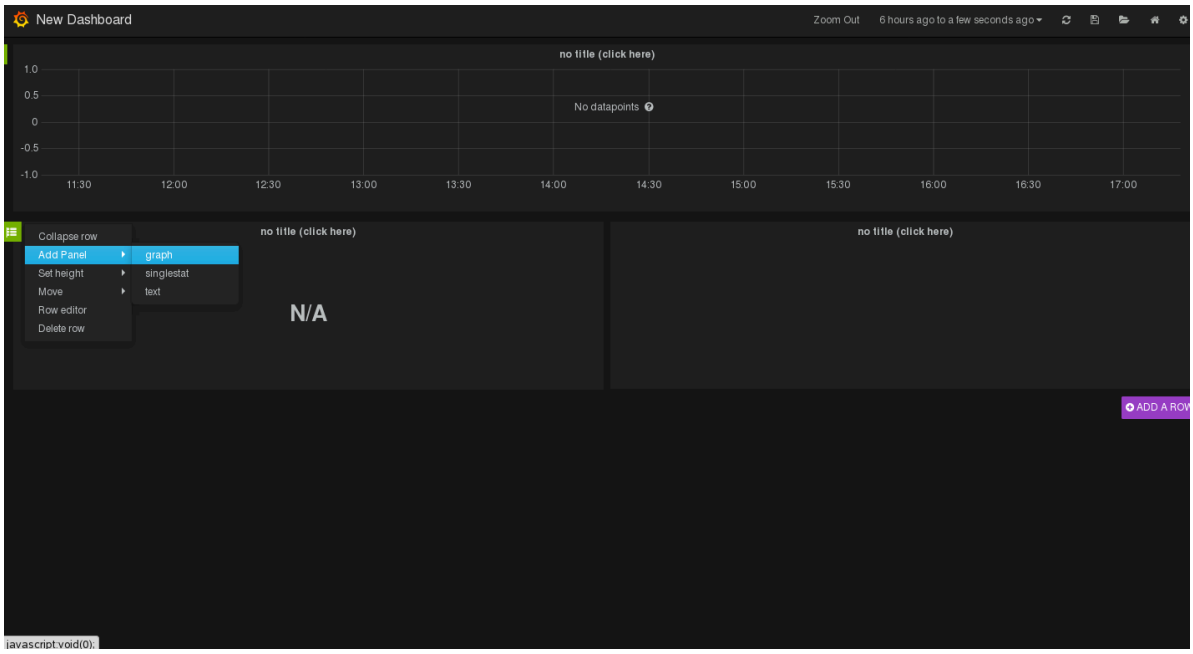
- Start from scratch and create a new dashboard.

- Take the dashboard that is shipped with SUSE OpenStack Cloud Monitoring as a starting point and customize it.

The following sections provide introductory information on dashboards, rows, and panels, and make you familiar with the first steps involved in building a dashboard. For additional information, you can also refer to the Grafana documentation (http://docs.grafana.org/).

Creating a Dashboard

To create a new dashboard, use the *Open* icon in the top right corner of your dashboard window. The option provides access to various features for administrating dashboards. Click *New* to create an empty dashboard that serves as a starting point for adding rows and panels.



On the left side of an empty dashboard, there is a green rectangle displayed. Hover over this rectangle to access a *Row* menu. To insert your first panel, you can use the options in the *Add Panel* submenu. See below for details on the available panel types.

As soon as you have inserted an empty panel, you can add additional rows. For this purpose, use the *Add Row* option on the right side of the dashboard.

Editing Rows

Features for editing rows can be accessed via the green rectangle that is displayed to the left of each row.

In addition to adding panels to a row, you can collapse or remove a row, move the position of the row within your dashboard, or set the row height. Row settings allows you, for example, to insert a row title or to hide the *Row* menu so that the row can no longer be edited.

Editing Panels

Grafana distinguishes between three panel types:

Panels of type *Graph* are used to visualize metrics data. A query editor is provided to define the data to be visualized. The editor allows you to combine multiple queries. This means that any number of metrics and data series can be visualized in one panel.

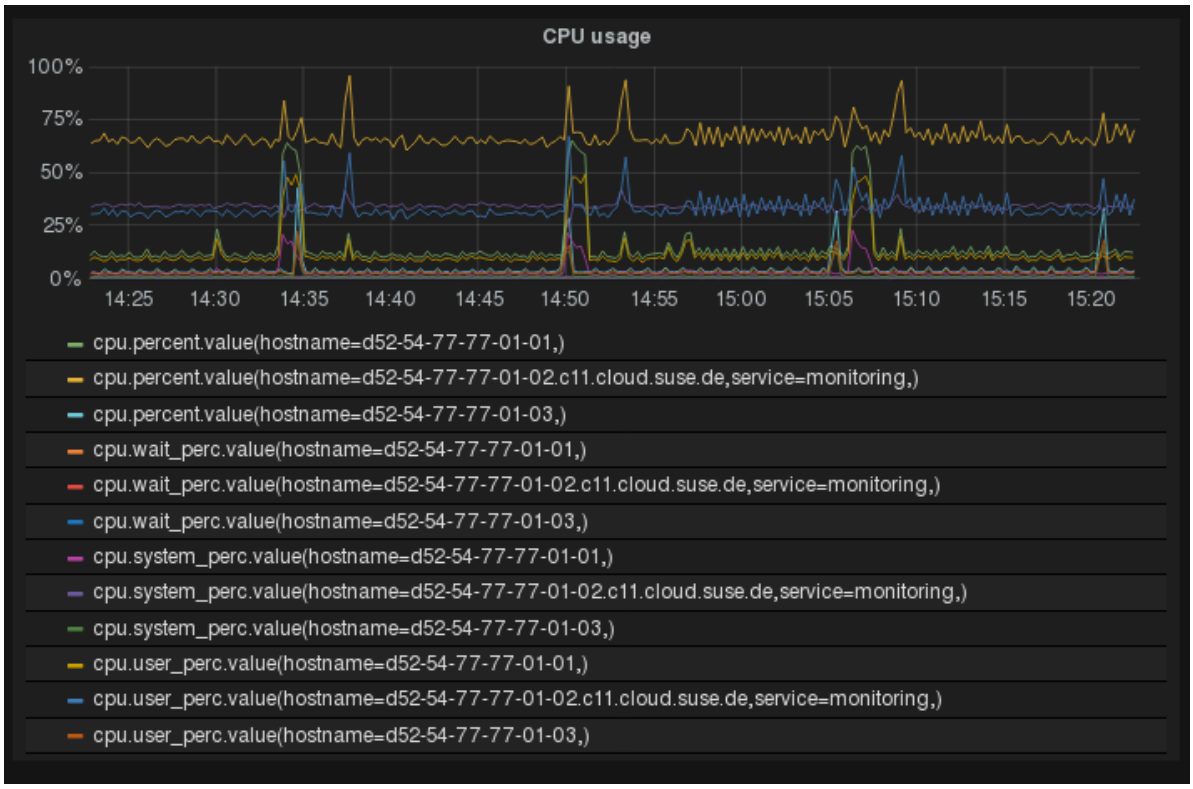


FIGURE 4.3: A GRAPH PANEL

Panels of type *Singlestat* are also used to visualize metrics data, yet they reduce a single query to a single number. The single number can be, for example, the minimum, maximum, average, or sum of values of the data series. The single number can be translated into a text value, if required.

Panels of type *Text* are used to insert static text. The text may, for example, provide information for the dashboard users. Text panels are not connected to any metrics data.

As soon as you have added a panel to your dashboard, you can access the options for editing the panel content. For this purpose, click the panel title and use *Edit*:

- For panels of type *Text*, a simple text editor is displayed for entering text. Plain text, HTML, and markdown format are supported.

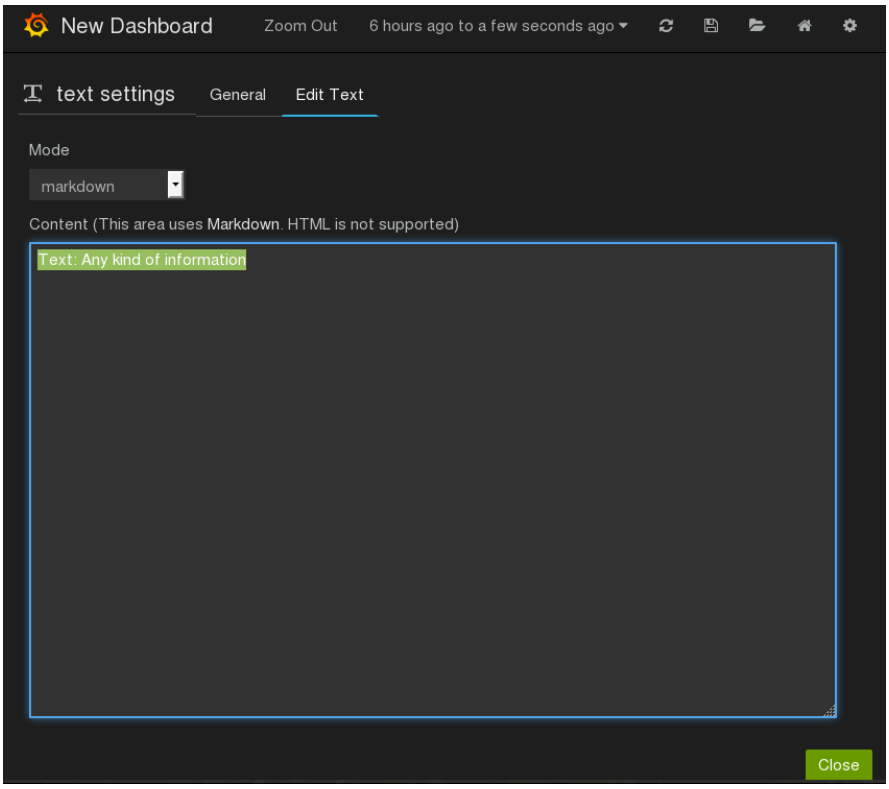


FIGURE 4.4: EDITING A TEXT PANEL

- For panels of type *Graph* and *Singlestat*, a query editor is displayed to define which data it to be shown. You can add multiple metrics, and apply functions to the metrics. The query results will be visualized in your panel in real time.

A large number of display and formatting features are provided to customize how the content is presented in a panel. Click the panel title to access the corresponding options. The menu that is displayed also allows you to duplicate or remove a panel. To change the size of a panel, click the + and - icons.

You can move panels on your dashboard by simply dragging and dropping them within and between rows.

By default, the time range for panels is controlled by dashboard settings. Use the time picker in the top right corner of your dashboard window to define relative or absolute time ranges. You can also set an auto-refresh interval, or manually refresh the data that is displayed.

Saving and Sharing Dashboards

SUSE OpenStack Cloud Monitoring allows you to save a metrics dashboard and export it to a JSON file. The JSON file can be edited, it can be shared with other users, and it can be imported to SUSE OpenStack Cloud Monitoring again.

To save a dashboard, use *Save* in the top right corner of your dashboard window. You can enter a name for the dashboard and simply save it to your browser's local storage. Use *Dashboard JSON* to directly view the corresponding JSON syntax, or use *Export dashboard* to download the JSON file. The JSON file can be forwarded to other users, if required. To import a JSON file, use *Open dashboard* in the top left corner of the dashboard window.

4.3 Defining Alarms

You have to define alarms to monitor your cloud resources. An alarm definition specifies the metrics to be collected and the threshold at which an alarm is to be triggered for a cloud resource. If the specified threshold is reached or exceeded, the alarm is triggered and notifications can be sent to inform users. By default, an alarm definition is evaluated every minute.

To handle a large variety of monitoring requirements, you can create either simple alarm definitions that refer to one metrics only, or compound alarm definitions that combine multiple metrics and allow you to track and process more complex events.

Example for a simple alarm definition that checks whether the system-level load of the CPU exceeds a threshold of 90 percent:

```
cpu.system_perc{hostname=monasca} > 90
```

Example for a simple alarm definition that checks the average time of the system-level load of the CPU over a period of 480 seconds. The alarm is triggered only if this average is greater than 95 percent:

```
avg(cpu.system_perc{hostname=monasca}, 120) > 95 times 4
```

Example for a compound alarm definition that evaluates two metrics. The alarm is triggered if either the system-level load of the CPU exceeds a threshold of 90 percent, or if the disk space that is used by the specified service exceeds a threshold of 90 percent:

```
avg(cpu.system_perc{hostname=monasca}) > 90 OR
max(disk.space_used_perc{service=monitoring}) > 90
```

To create, edit, and delete alarms, use *Monitoring* > *Alarm Definitions*.

The elements that define an alarm are grouped into *Details*, *Expression*, and *Notifications*. They are described in the following sections.

Details

For an alarm definition, you specify the following details:

- *Name*. Mandatory identifier of the alarm. The name must be unique within the project for which you define the alarm.
- *Description*. Optional. A short description that depicts the purpose of the alarm.
- *Severity*. The following severities for an alarm are supported: *Low* (default), *Medium*, *High*, or *Critical*.
  The severity affects the status information on the *Overview* page. If an alarm that is defined as *Critical* is triggered, the corresponding resource is displayed in a red box. If an alarm that is defined as *Low*, *Medium*, or *High* is triggered, the corresponding resource is displayed in a yellow box only.
  The severity level is subjective. Choose a level that is appropriate for prioritizing the alarms in your environment.

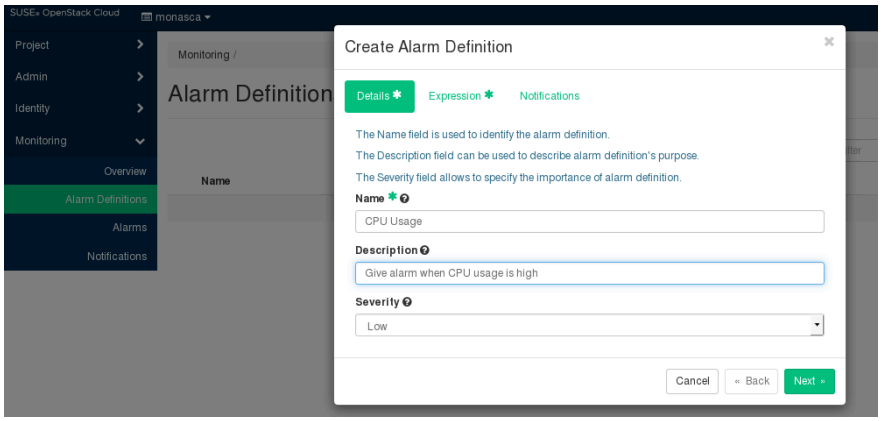
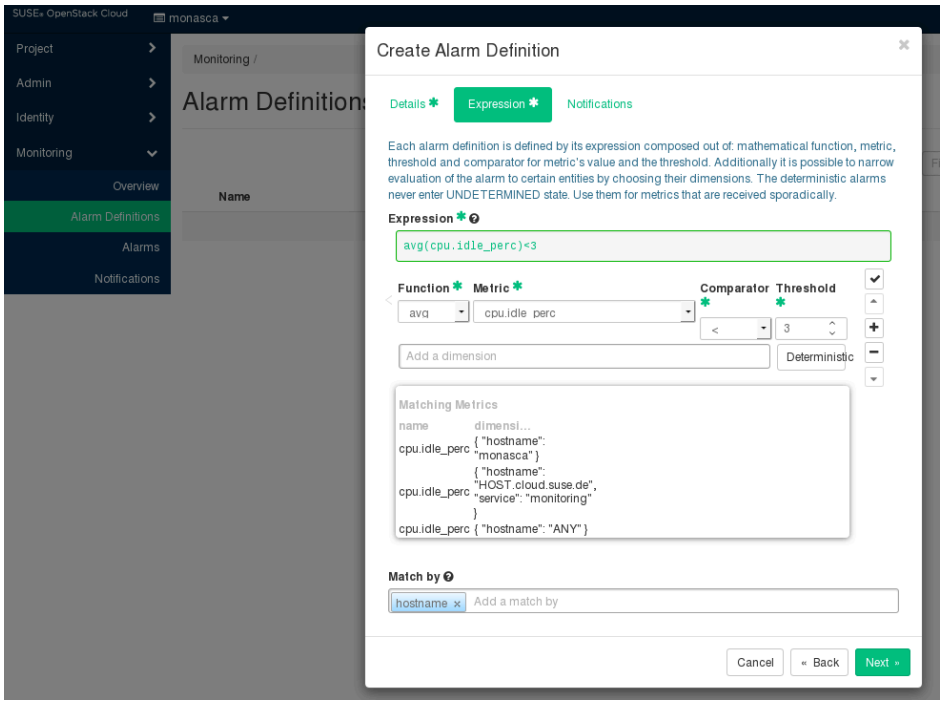


FIGURE 4.5: CREATING AN ALARM DEFINITION

Expression

The expression defines how to evaluate a metrics. The expression syntax is based on a simple expressive grammar. For details, refer to the Monasca API documentation (https://github.com/openstack/monasca-api/blob/stable/ocata/docs/monasca-api-spec.md).



To define an alarm expression, proceed as follows:

1. Select the metrics to be evaluated.

2. Select a statistical function for the metrics: `min` to monitor the minimum values, `max` to monitor the maximum values, `sum` to monitor the sum of the values, `count` for the monitored number, or `avg` for the arithmetic average.

3. Enter one or multiple dimensions in the *Add a dimension* field to further qualify the metrics. Dimensions filter the data to be monitored. They narrow down the evaluation to specific entities. Each dimension consists of a key/value pair that allows for a flexible and concise description of the data to be monitored, for example, region, availability zone, service tier, or resource ID.
   The dimensions available for the selected metrics are displayed in the *Matching Metrics* section. Type the name of the key you want to associate with the metrics in the *Add a dimension* field. You are offered a select list for adding the required key/value pair.

4. Enter the threshold value at which an alarm is to be triggered, and combine it with a relational operator `<`, `>`, `<=`, or `>=`.
   The unit of the threshold value is related to the metrics for which you define the threshold, for example, the unit is percentage for `cpu.idle_perc` or MB for `disk.total_used_space_mb`.

5. Switch on the *Deterministic* option if you evaluate a metrics for which data is received only sporadically. The option should be switched on, for example, for all log metrics. This ensures that the alarm status is `OK` and displayed as a green box on the *Overview* page although metrics data has not yet been received.
   Do not switch on the option if you evaluate a metrics for which data is received regularly. This ensures that you instantly notice, for example, that a host machine is offline and that there is no metrics data for the agent to collect. On the *Overview* page, the alarm status therefore changes from `OK` to `UNDETERMINED` and is displayed as a gray box.

6. Enter one or multiple dimensions in the *Match by* field if you want these dimensions to be taken into account for triggering alarms.
   Example: If you enter `hostname` as dimension, individual alarms will be created for each host machine on which metrics data is collected. The expression you have defined is not evaluated as a whole but individually for each host machine in your environment.
   If *Match by* is set to a dimension, the number of alarms depends on the number of dimension values on which metrics data is received. An empty *Match by* field results in exactly one alarm.
   To enter a dimension, you can simply type the name of the dimension in the *Match by* field. The dimensions you enter cannot be changed once the alarm definition is saved.

7. Build a compound alarm definition to combine multiple metrics in one expression. Using the logical operators `AND` or `OR`, any number of sub-expressions can be combined.
   Use the *Add* button to create a second expression, and choose either `AND` or `OR` as *Operator* to connect it to the one you have already defined. Proceed with the second expression as described in Step 1 to Step 6 above.
   The following options are provided for creating and organizing compound alarm definitions:

   - Create additional sub-expressions using the *Add* button.
   - Finish editing a sub-expression using the *Submit* button.
   - Delete a sub-expression using the *Remove* button.
   - Change the position of a sub-expression using the *Up* or *Down* button.

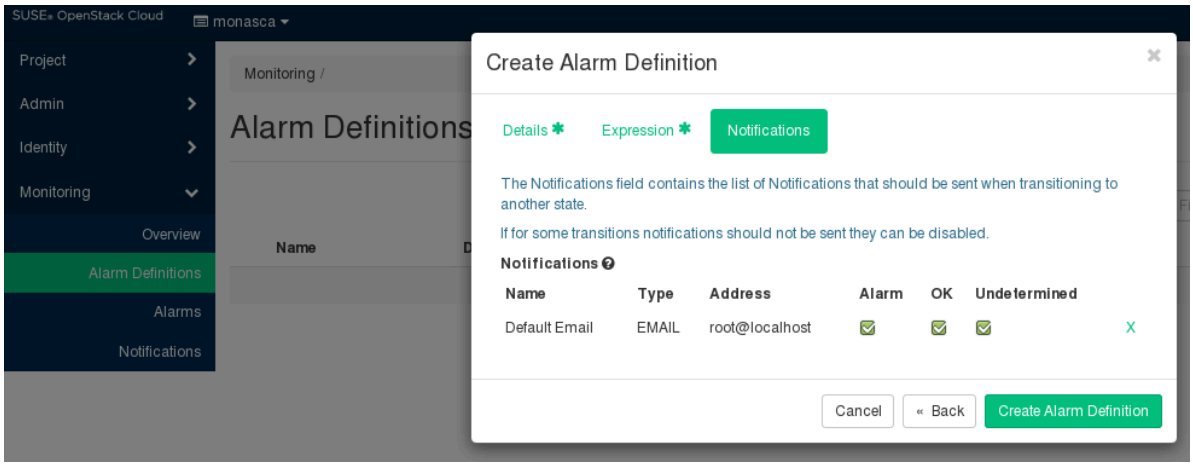
Note

You can also edit the expression syntax directly. For this purpose, save your alarm definition and update it using the *Edit Alarm Definition* option.

By default, an alarm definition is evaluated every minute. When updating the alarm definition, you can change this interval. For syntax details, refer to the Monasca API documentation on Alarm Definition Expressions (https://github.com/openstack/monasca-api/blob/stable/ocata/docs/monasca-api-spec.md#alarm-definition-expressions).

Notifications

You can enable notifications for an alarm definition. As soon as an alarm is triggered, the enabled notifications will be sent.



The *Notifications* tab allows you to select the notifications from the ones that are predefined in your environment. For a selected notification, you specify whether you want to send it for a status transition to *Alarm*, *OK*, and/or *Undetermined*.

For details on defining notifications, refer to *Section 4.4, "Defining Notifications"*. For details on alarm statuses, refer to *Section 4.5, "Status of Services, Servers, and Log Data"*.

4.4 Defining Notifications

Notifications define how users are informed when a threshold value defined for an alarm is reached or exceeded. In the alarm definition, you can assign one or multiple notifications.

For a notification, you specify the following elements:

- *Name*. A unique identifier of the notification. The name is offered for selection when defining an alarm.
- *Type*. `Email` is the notification method supported by SUSE OpenStack Cloud Monitoring. If you want to use `WebHook` or `PagerDuty`, contact your SUSE OpenStack Cloud Monitoring support for further information.
- *Address*. The email address to be notified when an alarm is triggered.

Note

Generic top-level domains such as business domain names are not supported in email addresses (for example, `user@xyz.company`).

To create, edit, and delete notifications, use *Monitoring* > *Notifications*.

4.5 Status of Services, Servers, and Log Data

An alarm definition for a service, server, or log data is evaluated over the interval specified in the alarm expression. The alarm definition is re-evaluated in each subsequent interval. The following alarm statuses are distinguished:

- `Alarm`. The alarm expression has evaluated to `true`. An alarm has been triggered for the cloud resource.
- `OK`. The alarm expression has evaluated to `false`. There is no need to trigger an alarm.
- `Undetermined`. No metrics data has been received within the defined interval.

As soon as you have defined an alarm for a cloud resource, there is status information displayed for it on the *Overview* page:

The color of the boxes in the three sections indicates the status:

- A green box for a service or server indicates that it is up and running. A green box for a log path indicates that a defined threshold for errors or warnings, for example, has not yet been reached or exceeded. There are alarms defined for the services, servers, or log paths, but no alarms have been triggered.
- A red box for a service, server, or log path indicates that there is a severe problem that needs to be checked. One or multiple alarms defined for a service, a server, or log data have been triggered.
- A yellow box indicates a problem. One or multiple alarms have already been triggered, yet, the severity of these alarms is low.
- A gray box indicates that alarms have been defined. Yet, metrics data has not been received.

The status information on the *Overview* page results from one or multiple alarms that have been defined for the corresponding resource. If multiple alarms are defined, the severity of the individual alarms controls the status color.

You can click a resource on the *Overview* page to display details on the related alarms. The details include the status of each alarm and the expression that is evaluated. For each alarm, you can drill down on the alarm history. To narrow down the problem, the history presents detailed information on the status transitions.

5 Log Management

For managing the log data of your services and the virtual and physical servers on which they are provisioned, SUSE OpenStack Cloud Monitoring integrates with Kibana, an open source analytics and visualization platform. SUSE OpenStack Cloud Monitoring uses Kibana as a front-end application to the log data held in the Elasticsearch database.

Kibana allows you to easily understand large data volumes. Based on the data that is stored in Elasticsearch indices, you can perform advanced data analysis and visualize your log data in a variety of charts, tables, or maps. Changes to the Elasticsearch indices are displayed in SUSE OpenStack Cloud Monitoring in real time.

The log management features of SUSE OpenStack Cloud Monitoring include:

- Features for searching, visualizing, and analyzing the log data.
- Alerting features for monitoring.

In the following sections, you will find information on the log management window where you search, visualize, and analyze your log data, as well as details on how to use the alerting features.

Accessing SUSE OpenStack Cloud Monitoring

For accessing SUSE OpenStack Cloud Monitoring and performing log management tasks, the following prerequisites must be fulfilled:

- You must have access to the OpenStack platform as a user with the `monasca-user` role.
- You must be assigned to the OpenStack project you want to monitor.

Log in to OpenStack Horizon with your user name and password. The functions you can use in OpenStack Horizon depend on your access permissions. To access logs and metrics, switch to the *monasca* tenant in Horizon.

The SUSE OpenStack Cloud Monitoring functionality is available on the *Monitoring* tab. It provides access to the log data of all projects to which you are assigned. The *Log Management* option at the top border of the *Overview* page displays the log management window where you can work on the log data.

5.1 Working with the Log Management Window

Index patterns determine which data from the underlying Elasticsearch database can be viewed and analyzed in SUSE OpenStack Cloud Monitoring's log management window. Index patterns are used to identify the Elasticsearch indices to run search and analytics against.

SUSE OpenStack Cloud Monitoring ships with a preconfigured index pattern which allows you to instantly view and analyze your log data when accessing the log management window for the first time. You can configure additional index patterns to view and analyze different data from different indices. For details, refer to *Section 5.2, "Configuring Index Patterns"*.

Search queries allow you to search the Elasticsearch indices for data that match your information requirements. The query results can be graphically represented in visualizations, and visualizations can be organized in dashboards.

The log management window provides features for:

- Querying log data.
- Visualizing query results.
- Combining visualizations in dashboards.
- Filtering query results in dashboards.
- Sharing dashboards.

The following sections provide an introduction to queries, visualizations, and dashboards. For additional details, refer to the Kibana documentation (https://www.elastic.co/guide/en/kibana/4.5/index.html).

Querying Log Data

For querying log data, you use the *Discover* page in the log management window. It is instantly displayed when you access the window. It shows the most recently collected log data:

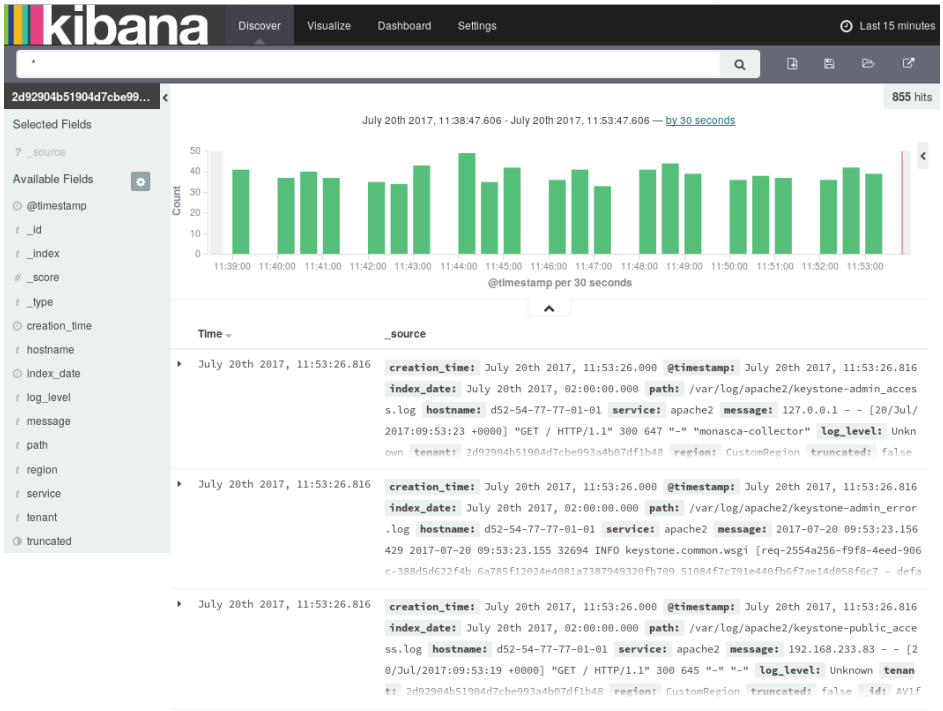


FIGURE 5.1: THE KIBANA DASHBOARD—DISCOVER PAGE

The *Discover* page allows you to access the log data in every index that matches the current index pattern. In addition to submitting queries, you can view, filter, and analyze the log data that is returned by your queries.

On the *Discover* page the following elements assist you in analyzing your log data:

- Below the main navigation bar at the top of the window, there is a **search box** for querying your log data. By submitting a query, you search all indices that match the current index pattern. The name of the current index pattern is displayed directly below the search box on the left side. You can select a different index pattern, if required. For details on configuring and selecting index patterns, refer to *Section 5.2, "Configuring Index Patterns"*.

For entering strings in the search box, use the Lucene query syntax. Kibana also supports the Elasticsearch Query DSL. For details, refer to the Elasticsearch Reference documentation (https://www.elastic.co/guide/en/elasticsearch/reference/2.3/query-dsl.html).

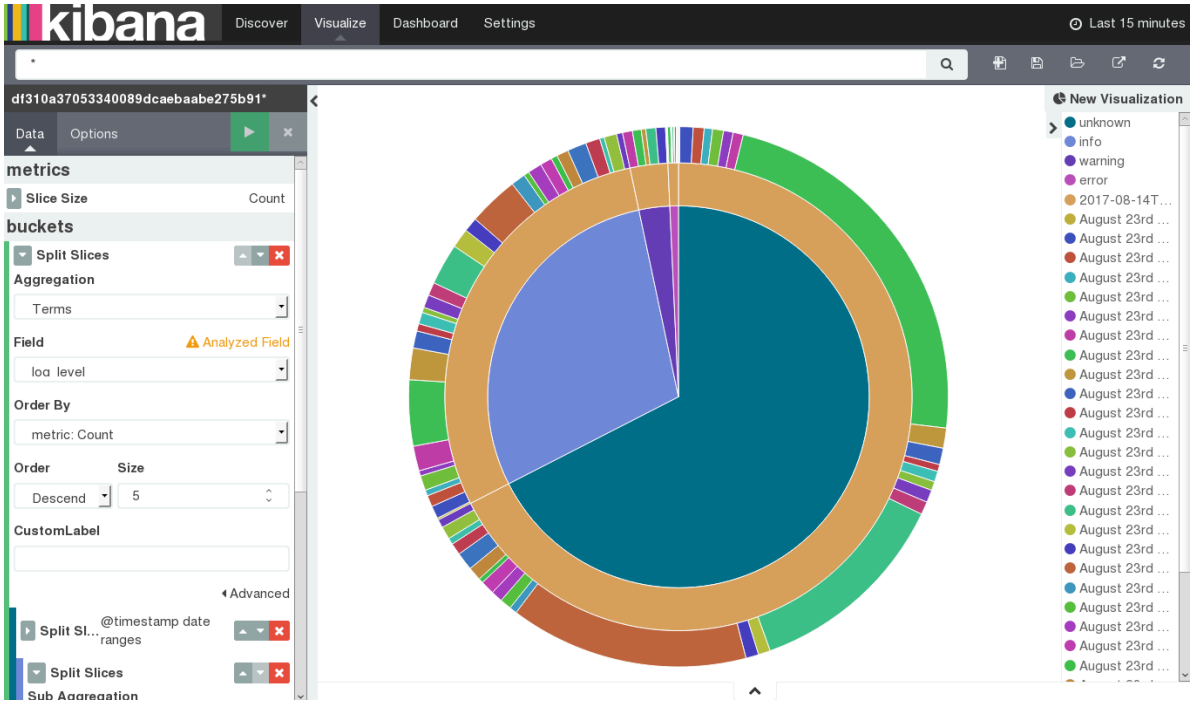
- Use the **clock icon** at the top right border of the log management window to define a time range for filtering the log data. By default, SUSE OpenStack Cloud Monitoring displays the log data collected during the last 15 minutes. You can deviate from this default. Multiple options are provided for defining relative or absolute time ranges. The time range you define is instantly applied to all log data.
- In the bottom right part of the *Discover* page, you can view the **log data** returned by your search queries. Depending on whether you have filtered the data by index fields, the log data is either restricted to these fields or entire records are displayed.
- On the left side of the *Discover* page below the search box, you see the **index fields** from the indices that match the current index pattern. You can select individual fields to modify which log data is displayed on the right side.
  Select a field from the *Available Fields* section for this purpose and use *Add*. To remove a field, select it in the *Selected Fields* section and use *Remove*.
  From the field list, you can expand a field by simply clicking it. This shows the most common values for the field. You can also set field values as filter, or you can exclude log data with specific field values.
- If a time field is configured for the current index pattern, the distribution of log entries over time is displayed in a **histogram** in the top right part of the *Discover* page.
  By default, the histogram shows the number of logs entries versus time, matched by the underlying query and time filter. You can click the bars in the histogram to narrow down the time filter.

Queries can be saved and re-used. They can also be shared with other users. For this purpose, use the options to the right of the search box at the top border of the log management window:

- To save a query, use *Save Search*. Saving a query means saving both the query syntax and the current index pattern.
- To load a query, use *Load Saved Search*. A saved query can be loaded and used by any OpenStack or Monitoring Service operator.
- To share a query with other users, use *Share Search*. The option displays a direct link to the query that you can forward. As a prerequisite for using a direct link, a user must have SUSE OpenStack Cloud Monitoring access.

Visualizing Query Results

SUSE OpenStack Cloud Monitoring supports you in building graphical representations of your query results. You can choose from different visualization types, for example, pie charts, data tables, line charts, or vertical bar charts. For visualizing your results, you use the *Visualize* page in the log management window:



To create a visualization, use *New Visualization* to the right of the search box at the top border of the window. You have to select a visualization type and the query to be used. You can either create a new query or load a query you have already saved.

Based on the visualization type and the query, you can proceed with designing the graphical representation in a visualization editor. Multiple design options and a preview function are provided for creating, modifying, and viewing the graphical representation.

You can save and re-use visualizations. You can also share them with other users. For this purpose, use the options to the right of the search box at the top border of the log management window:

- To save a visualization, use *Save Visualization*.
- To load a visualization, use *Load Saved Visualization*. A saved visualization can be loaded and used by any OpenStack or Monitoring Service operator.
- To share a visualization with other users, use *Share Visualization*. The option displays an HTML snippet that can be used to embed the visualization in a Web page. It also displays a direct link to the visualization that you can forward. As a prerequisite for using an embedded visualization or a direct link, a user must have SUSE OpenStack Cloud Monitoring access.

Combining Visualizations in Dashboards

For correlating related information or providing an overview, you can combine visualizations in dashboards. Use the *Dashboard* page in the log management window for this purpose:

To create a dashboard from scratch, you use *New Dashboard* to the right of the search box at the top border of the window. To add a visualization from a list of existing visualizations, use *Add Visualization*. You need at least one saved visualization to create a dashboard. In addition to adding visualizations, you can also place the tabular output of query results on your dashboards. Switch to the *Searches* tab when adding a visualization, and select a saved query. This adds the query result to your dashboard.

A visualization or query result is displayed in a container on your dashboard. Various options are provided for arranging containers:

- Move a container by clicking and dragging its title bar.
- Resize a container by dragging its bottom right corner.
- Remove a container using *Delete* in the top right corner of the container.

Using *Edit* in the top right corner of a container, you can switch to the *Visualize* or *Discover* page. This allows you to design the graphical representation or edit the query. To view the raw data behind a visualization, you can click the bar at the bottom of the container. This replaces your visualization by the underlying raw data. You can export the raw data, if required.

For each dashboard, you can configure a refresh interval to automatically refresh its content with the latest data. The current interval is displayed in the top right border of the log management window. Click the interval if you want to change it. You can define the interval in absolute or relative terms. Use *Auto-Refresh* next to the interval in the border of the log management window to instantly submit the underlying queries and refresh the dashboard content.

By default, dashboards are displayed with a light background. Using *Options* in the top right border of the log management window, you can switch to a dark color scheme.

Filtering Query Results in Dashboards

By submitting a query on the data displayed in a dashboard, you can filter out specific sets of data that you want to aggregate while not changing the logic of the individual visualizations.

Use the search box below the main navigation bar at the top of the log management window for entering a query on the whole dashboard. If a visualization is already based on a saved query, both queries apply.

Sharing Dashboards

Dashboards can be saved and re-used. They can also be shared with other users. For this purpose, use the options to the right of the search box at the top border of the log management window:

- To save a dashboard, use *Save Dashboard*. By default, saving a dashboard also saves the time filter that is defined at the time of saving. You can disable this default by clearing the *Store time with dashboard* option. Disabling the default means that the time filter is set to the currently selected time each time the dashboard is loaded.
- To load a dashboard, use *Load Saved Dashboard*. A saved dashboard can be loaded and used by any OpenStack or Monitoring Service operator.
- To share a dashboard with other users, use *Share Dashboard*. The option displays an HTML snippet that can be used to embed the dashboard in a Web page. It also displays a direct link to the dashboard that you can forward. As a prerequisite for using an embedded dashboard or a direct link, a user must have SUSE OpenStack Cloud Monitoring access.

5.2 Configuring Index Patterns

SUSE OpenStack Cloud Monitoring enables the dynamic mapping of fields. After configuring an index pattern, the indices that match the pattern are automatically scanned to display the list of index fields. This guarantees that the fields are correctly visualized in the dashboard.

SUSE OpenStack Cloud Monitoring ships with a preconfigured index pattern that allows you to instantly explore your Elasticsearch indices when accessing the dashboard for the first time. You can create additional patterns to view and analyze specific sets of data. One or multiple patterns can be created per project. When you create additional patterns, you have to set one of them as the default.

To configure an additional index pattern, use *Settings* > *Indices*. Click the index pattern that is displayed in the *Index Patterns* field on the left, and use the *Add New* option.

Indices that match the pattern you define must exist in the Elasticsearch database, and they must contain data. For an index pattern, you specify the following elements:

- *Index contains time-based events*. It is recommended that this option is selected. This improves search performance by enabling searches only on those indices that contain data on time-based events.

- *Use event times to create index names*. It is recommended that this option is selected. This improves search performance by enabling searches only on those indices that contain data in the time range you specify.

- *Index pattern interval*. Select `Daily` as index pattern interval. Daily intervals are supported by the Monitoring Service.

- *Index name or pattern*. The pattern allows you to define dynamic index names. Static text in a pattern is denoted using brackets. Replace the predefined pattern (`[logstash-]*` or `[logstash-]YYYY.MM.DD`) as follows:
  Replace `logstash-` by the project ID of the OpenStack project whose log data is to be visualized in the dashboard.
  Replace `*` or `YYYY.MM.DD` by `YYYY-MM-DD` as naming pattern. This naming pattern is supported by the Monitoring Service.
  Example: `[557aff4bf007473d84069aca202a1633-]YYYY-MM-DD`

- *Time-field name*. Select `@timestamp` as time-field name. `@timestamp` matches the `YYYY-MM-DD` naming pattern.

The default index pattern is automatically loaded when you access the log management window. It is marked with an asterisk in front of the pattern name in the *Index Patterns* field at the top left corner of the *Settings* page. Select the pattern you want to set as the default from the *Index Patterns* field. The content of the log management window is instantly updated.

5.3 Monitoring Log Data

SUSE OpenStack Cloud Monitoring provides alerting features for monitoring your log data. Specific log metrics support you in checking the severity of the entries in your log files. Log metrics are handled like any other metrics in SUSE OpenStack Cloud Monitoring. They complete the log management features and support you in analyzing and troubleshooting any issue that you encounter in your log data.

Using the log metrics for monitoring corresponds to using any other metrics:

- Use *Monitoring* > *Alarm Definitions* to create, edit, and delete alarms for log data.
- Use *Monitoring* > *Notifications* to create, edit, and delete notifications for alarms.
- Use *Monitoring* > *Overview* to check whether there are any irregularities in your log data. As soon as you have defined an alarm for your log data and metrics data has been received, there is status information displayed on the *Overview* page.

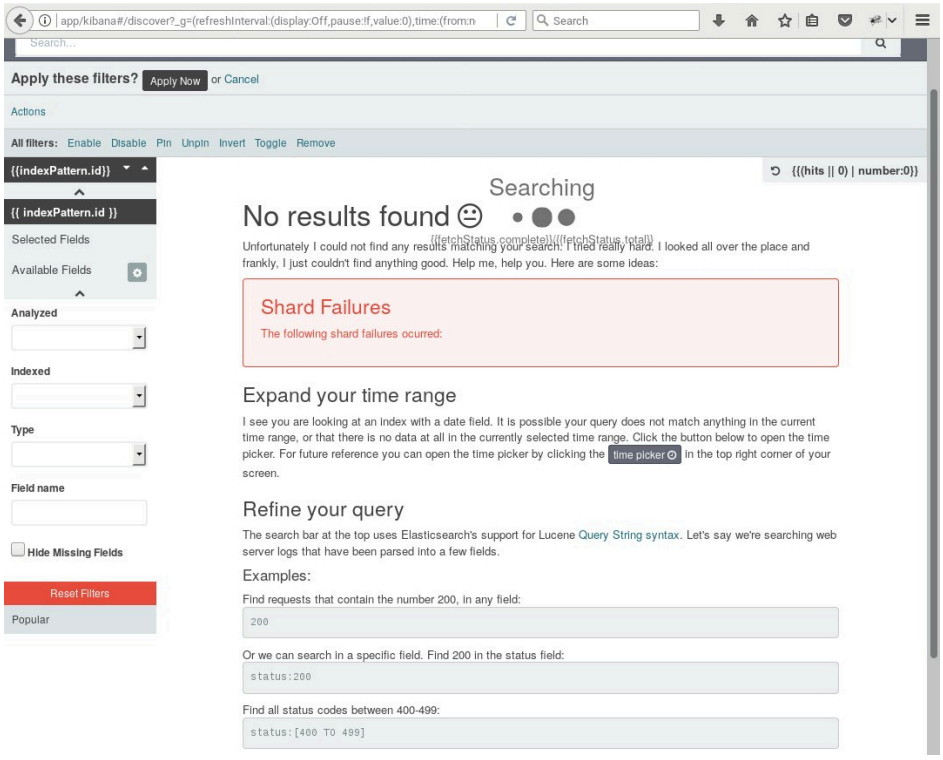
For details on using the log metrics, refer to *Chapter 4, Monitoring*.

5.4 Troubleshooting

Here are some common errors, and how to resolve them.

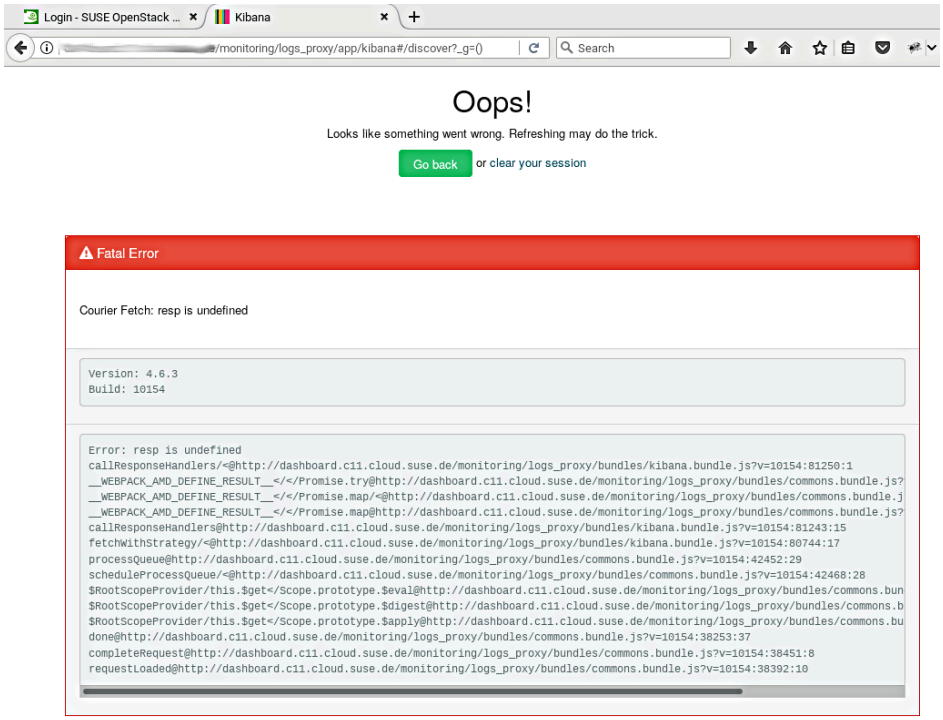
Shard Failures

This problem may occur the first time the Log Management Window is opened. It occurs because the Elasticsearch index for the Monasca project does not yet exist. A browser reload should fix the problem in most cases.



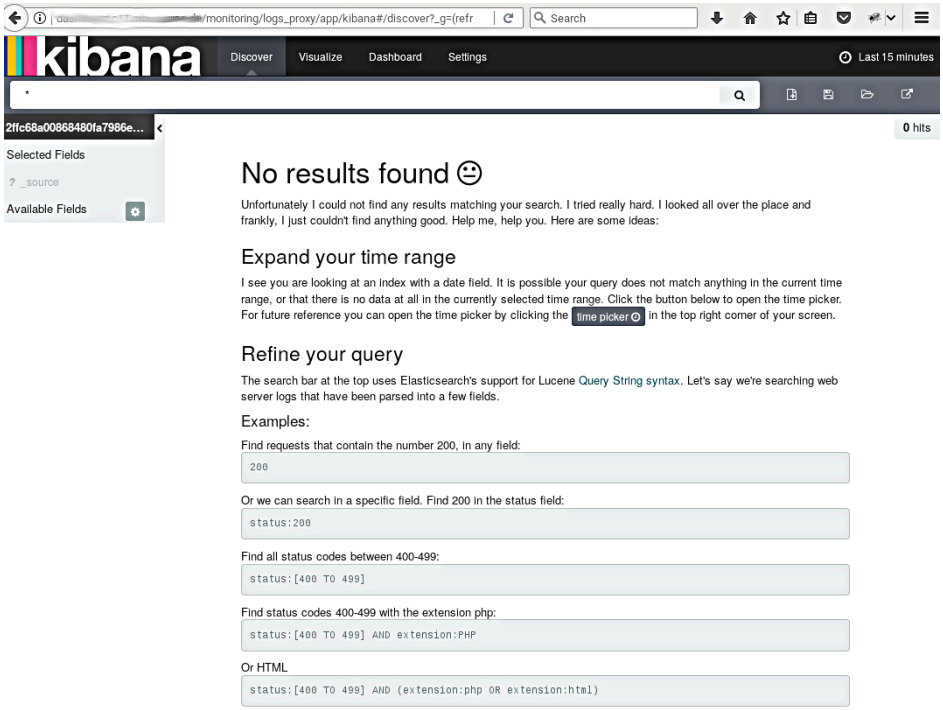
Oops! Looks like something went wrong

This problem happens when you were previously logged into Horizon and the session expired, leaving a stale session state that confuses Kibana. In this case, re-login to Horizon and then reload Kibana to fix the problem.



No results found

This may be due to an overly narrow time window (that is, no logs were sent in the last 15 minutes, which is the default time window). Try increasing the time window with the drop-down box in the screen's top right corner. If the problem persists verify that log agents are running properly; make sure the `openstack-monasca-log-agent` service is running and check `/var/log/monasca-log-agent/agent.log` for errors on the agent nodes.


/monitoring/logs_proxy/app/kibana#/discover?_g=(refr
Search
kibana
Discover
Visualize
Dashboard
Settings
Last 15 minutes
*
2ffc68a00868480fa7986e...
0 hits
Selected Fields
? _source
Available Fields
No results found
Unfortunately I could not find any results matching your search. I tried really hard. I looked all over the place and frankly, I just couldn't find anything good. Help me, help you. Here are some ideas:
Expand your time range
I see you are looking at an index with a date field. It is possible your query does not match anything in the current time range, or that there is no data at all in the currently selected time range. Click the button below to open the time picker. For future reference you can open the time picker by clicking the time picker in the top right corner of your screen.
Refine your query
The search bar at the top uses Elasticsearch's support for Lucene Query String syntax. Let's say we're searching web server logs that have been parsed into a few fields.
Examples:
Find requests that contain the number 200, in any field:
200
Or we can search in a specific field. Find 200 in the status field:
status:200
Find all status codes between 400-499:
status:[400 TO 499]
Find status codes 400-499 with the extension php:
status:[400 TO 499] AND extension:PHP
Or HTML
status:[400 TO 499] AND (extension:php OR extension:html)

A Supported Metrics

The sections below describe the metrics supported by SUSE OpenStack Cloud Monitoring:

- Standard metrics for general monitoring of servers and networks.
- Additional metrics for monitoring specific servers and services.

A.1 Standard Metrics

SUSE OpenStack Cloud Monitoring supports the following standard metrics for monitoring servers and networks. These metrics usually do not require specific settings. The metrics are grouped by metrics types. Each metrics type references a set of related metrics.

cpu.yaml

Metrics on CPU usage, e.g. the percentage of time the CPU is idle when no I/O requests are in progress, or the percentage of time the CPU is used at system level or user level.

disk.yaml

Metrics on disk space, e.g. the percentage of disk space that is used on a device, or the total amount of disk space aggregated across all the disks on a particular node.

load.yaml

Metrics on the average system load over different periods (e.g. 1 minute, 5 minutes, or 15 minutes).

memory.yaml

Metrics on memory usage, e.g. the number of megabytes of total memory or free memory, or the percentage of free swap memory.

network.yaml

Metrics on the network, e.g. the number of network bytes received or sent per second, or the number of network errors on incoming or outgoing network traffic per second.

A.2 Additional Metrics

In addition to the standard metrics, SUSE OpenStack Cloud Monitoring automatically adds the following additional metrics to the monasca agent configuration on the Monitoring Node.

elastic.yaml

Elastic checks gather metrics for Elasticsearch databases, such as the Log Database of SUSE OpenStack Cloud Monitoring. The configuration file must specify the URL for HTTP requests. If basic authentication is used, for example, `elasticsearch-http-basic`, the configuration file must also specify the user name and password for every instance that requires authentication.

The agent installer automatically creates the `elastic.yaml` configuration file in the `/etc/monasca/agent/conf.d/` directory. If there is an Elasticsearch database instance installed on the machine where the agent is installed, you have to specify the configuration information in the `elastic.yaml` file after the installation.

Example configuration:

```
init_config: null
instances:
- dimensions:
    component: elasticsearch
    service: monitoring
  url: http://localhost:9200
  username: username
  password: password
```

http_check.yaml

HTTP endpoint checks perform up/down checks on HTTP endpoints. Based on a list of URLs, the agent sends an HTTP request and reports success or failure to the Monitoring Service.

Crowbar configures HTTP checks for the following services on the Monitoring Node:

- monasca-api
- monasca-log-api
- Kibana
- Monasca Persister

http_metrics.yaml

HTTP metrics checks retrieve metrics from any URL that returns a JSON response. Based on a list of URLs, the agent can dispatch an HTTP request and parse the desired metrics from the JSON response.

Crowbar configures a HTTP metrics check for the monasca-persister service running on the Monitoring Node.

kafka_consumer.yaml

Kafka consumer checks gather metrics related to services consuming Kafka topics, such as the Persister or Notification Engine of SUSE OpenStack Cloud Monitoring.

Crowbar configures Kafka consumer checks for the Kafka service running on the Monitoring Node.

kibana.yaml

Kibana checks gather various statistics such as load, heap size and usage, and various request performance statistics from a Kibana server. The checks retrieve these statistics from Kibana's status endpoint.

Crowbar configures Kibana checks for the Kibana service running on the Monitoring Node.

mysql.yaml

MySQL checks gather various statistics such as buffer sizes and usage, reads/writes, network connections, etc. from a MySQL database server. MariaDB is also supported. The metrics are drawn from the server status variables of MySQL.

Crowbar configures the MySQL checks for the MariaDB database service running on the Monitoring Node.

ntp.yaml

Network Time Protocol checks monitor the time offset between an NTP server and the machine the agent is running on.

Crowbar configures the NTP check to monitor the offset between the Monitoring Node and the NTP server running on the Administration Server.

postfix.yaml

Postfix checks monitor the queue sizes (incoming, active, deferred) on a Postfix mail server.

Crowbar configures Postfix checks for the Postfix service running on the Monitoring Node.

zk.yaml

ZooKeeper checks gather metrics on nodes and connections covered by ZooKeeper, a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services. The check parses the result of the ZooKeeper `stat` admin command.

Crowbar configures ZooKeeper checks for the ZooKeeper instance running on the Monitoring Node.

B Glossary

Application Operator

A person with limited access to cloud resources in OpenStack. An application operator provides services to end users or hosts services for his own development activities.

Dimension

A key/value pair that allows for a flexible and concise description of the data to be monitored, for example, region, availability zone, service tier, or resource ID. Each dimension describes a specific characteristic of the metrics to be monitored.

In SUSE OpenStack Cloud Monitoring, metrics are uniquely identified by a name and a set of dimensions. Dimensions can serve as a filter for the monitoring data.

Elasticsearch

An open source application that provides a highly scalable full-text search and analytics engine. SUSE OpenStack Cloud Monitoring uses Elasticsearch as the underlying technology for storing, searching, and analyzing large volumes of log data.

Grafana

An open source application for visualizing large-scale measurement data. SUSE OpenStack Cloud Monitoring integrates with Grafana for visualizing the monitoring data.

Infrastructure as a Service (IaaS)

The delivery of computer infrastructure (typically a platform virtualization environment) as a service.

InfluxDB

An open source time-series database that supports high write loads and large data set storage. SUSE OpenStack Cloud Monitoring uses InfluxDB as the underlying technology for storing metrics and the alarm history.

Kibana

An open source analytics and visualization platform designed to work with Elasticsearch. SUSE OpenStack Cloud Monitoring integrates with Kibana for visualizing the log data.

Logstash

An open source application that provides a data collection engine with pipelining capabilities. SUSE OpenStack Cloud Monitoring integrates with Logstash for collecting, processing, and outputting logs.

MariaDB

An open source relational database that provides an SQL-compliant interface for accessing data. SUSE OpenStack Cloud Monitoring uses MariaDB as the underlying technology for storing configuration information, alarm definitions, and notification methods.

Metrics

Self-describing data structures that allow for a flexible and concise description of the data to be monitored. Metrics values represent the actual monitoring data that is collected and presented in SUSE OpenStack Cloud Monitoring.

Monasca

An open source Monitoring as a Service solution that integrates with OpenStack. It forms the core of SUSE OpenStack Cloud Monitoring.

Monitoring Service Operator

A person responsible for maintaining and administrating SUSE OpenStack Cloud Monitoring.

OpenStack Operator

A person responsible for maintaining and administrating OpenStack, the underlying platform technology of SUSE OpenStack Cloud Monitoring.

Platform as a Service (PaaS)

The delivery of a computing platform and solution stack as a service.

Software as a Service (SaaS)

A model of software deployment where a provider licenses an application to customers for use as a service on demand.