The StackLight Collector Plugin for Fuel Documentation

Release 1.1.0

Mirantis Inc.

Mar 28, 2017

Contents

1 Overview 1

2 Installing StackLight Collector plugin for Fuel 10

3 Configuring StackLight Collector plugin for Fuel 12

4 Configuring alarms 19

5 Appendix 40

CHAPTER 1

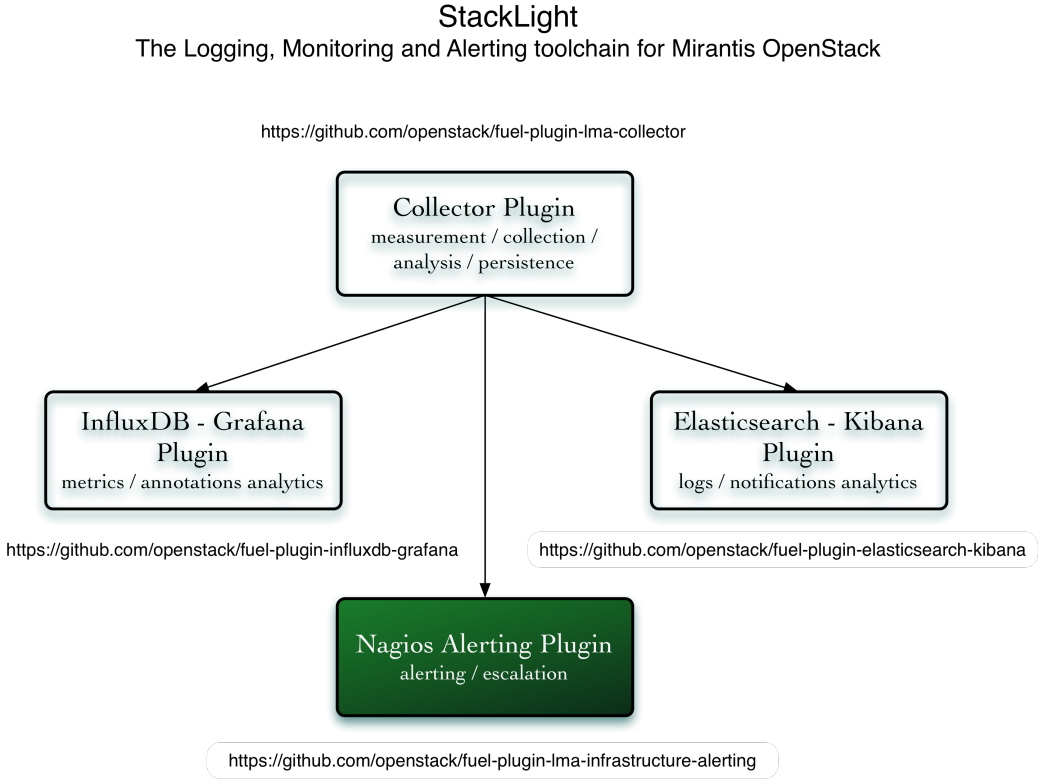
Overview

Introduction

The **StackLight Collector Plugin for Fuel** is used to install and configure several software components that are used to collect and process all the data that is relevant to provide deep operational insights about your OpenStack environment. These finely integrated components are collectively referred to as the **StackLight Collector**, or just **the Collector**.

Note: The Collector has evolved over time, so the term *collector* is a little bit of a misnomer since it is more of a *smart monitoring agent* than a mere data *collector*.

The Collector is the key component of the so-called Logging, Monitoring, and Alerting toolchain of Mirantis OpenStack, also known as StackLight.



The Collector is installed on every node of your OpenStack environment. Each Collector is individually responsible for supporting all the monitoring functions of your OpenStack environment for both the operating system and the services running on the node. The Collector running on the *primary controller* (the controller which owns the management VIP) is called the **Aggregator** since it performs additional aggregation and correlation functions. The Aggregator is the central point of convergence for all the faults and anomalies detected at the node level. The fundamental role of the Aggregator is to issue an opinion about the health status of your OpenStack environment at the cluster level. As such, the Collector may be viewed as a monitoring agent for cloud infrastructure clusters.

The main building blocks of the Collector are as follows:

- The **collectd** daemon, which comes bundled with a collection of monitoring plugins. Some of them are standard collectd plugins while others are purpose-built plugins written in Python to perform various OpenStack services checks.
- **Heka**, a Golang data-processing multifunctional tool by Mozilla. Heka supports a number of standard input and output plugins that allows to ingest data from a variety of sources including collectd, log files, and RabbitMQ, as well as to persist the operational data to external back-end servers like Elasticsearch, InfluxDB, and Nagios for search and further processing.
- **A collection of Heka plugins** written in Lua, which perform the actual data processing such as running metrics transformations, running alarms, and logs parsing.

Note: An important function of the Collector is to normalize the operational data into an internal Heka message structure representation that can be ingested into the Heka's stream-processing pipeline. The stream-processing pipeline uses matching policies to route the Heka messages to the Lua plugins that perform the actual data-computation functions.

The following Lua plugins were developed for the Collector:

- **decoder plugins** sanitize and normalize the ingested data.
- **filter plugins** process the data.
- **encoder plugins** serialize the data that is sent to the back-end servers.

The following are the types of data sent by the Collector (and the Aggregator) to the back-end servers:

- The logs and the notifications, which are referred to as events sent to Elasticsearch for indexing.
- The metric's time-series sent to InfluxDB.
- The annotations sent to InfluxDB.
- The OpenStack environment clusters health status sent as *passive checks* to Nagios.

Note: The annotations are like notification messages that are exposed in Grafana. They contain information about the anomalies and faults that have been detected by the Collector. Annotations basically contain the same information as the *passive checks* sent to Nagios. In addition, they may contain hints on what can be the root cause of a problem.

Requirements

The StackLight Collector plugin 1.1.0 has the following requirements:

Requirement	Version/Comment
Mirantis OpenStack	8.0, 9.x
A running Elasticsearch server (for log analytics)	1.7.4 or higher, the RESTful API must be enabled over port 9200
A running InfluxDB server (for metric analytics)	0.10.0 or higher, the RESTful API must be enabled over port 8086
A running Nagios server (for infrastructure alerting)	3.5 or higher, the command CGI must be enabled

Prerequisites

Prior to installing the StackLight Collector plugin for Fuel, you may want to install the back-end services the *collector* uses to store the data. These back-end services include the following:

- Elasticsearch
- InfluxDB
- Nagios

There are two installation options:

1. Install the back-end services automatically within a Fuel environment using the following Fuel plugins:
    - StackLight Elasticsearch-Kibana plugin
    - StackLight InfluxDB-Grafana plugin
    - StackLight Infrastructure Alerting plugin
2. Install the back-end services manually outside of a Fuel environment. In this case, the installation must comply with the *requirements* of the StackLight Collector plugin.

Limitations

The StackLight Collector plugin 1.1.0 has the following limitations:

- The plugin is not compatible with an OpenStack environment deployed with nova-network.
- When you re-execute tasks on deployed nodes using the Fuel CLI, the *collectd* processes will be restarted on these nodes during the post-deployment phase. See bug #1570850.

Release notes

Version 1.1.0

Version 1.0.0

The StackLight Collector plugin 1.0.0 for Fuel contains the following updates:

New alarms:

- Monitor RabbitMQ based on Pacemaker point-of-view
- Monitor all partitions and OSD disk(s)
- Horizon HTTP 5xx errors
- Keystone slow response times
- HDD errors
- SWAP percent usage
- Network packet drops
- Local OpenStack API checks
- Local checks for services: Apache, Memcached, MySQL, RabbitMQ, Pacemaker
- Monitor Nova resource utilization per aggregate (virtual CPUs, memory and disk)

Alarm enhancements:

- Added the `group by` attribute support for alarm rules
- Added support for `pattern matching` to filter metric dimensions

Bug fixes:

- Fixed the concurrent execution of logrotate. See #1455104.
- Implemented the capability for the Elasticsearch bulk size to increase when required. See #1617211.
- Implemented the capability to use RabbitMQ management API in place of the **`rabbitmqctl`** command.
- Enforce timezone setting in log processing. See #1633074.
- Improve the resilience of the log_collector. See #1643280.
- Support Oslo messaging v2 notifications See #1648479.

Version 0.10.0

Additionally to the bug fixes, the StackLight Collector plugin 0.10.0 for Fuel contains the following updates:

- Separated the processing pipeline for logs and metrics.

  Prior to StackLight version 0.10.0, there was one instance of the *hekad* process running to process both the logs and the metrics. Starting with StackLight version 0.10.0, the processing of the logs and notifications is separated from the processing of the metrics in two different *hekad* instances. This allows for better performance and control of the flow when the maximum buffer size on disk has reached a limit. With the *hekad* instance processing the metrics, the buffering policy mandates to drop the metrics when the maximum buffer size is reached. With the *hekad* instance processing the logs, the buffering policy mandates to block the entire processing pipeline.

This helps to avoid losing logs (and notifications) when the Elasticsearch server is inaccessible for a long period of time. As a result, the StackLight collector has now two processes running on the node:

  - One for the *log_collector* service
  - One for the *metric_collector* service

- The metrics derived from logs are now aggregated by the *log_collector* service.

  To avoid flooding the *metric_collector* with bursts of metrics derived from logs, the *log_collector* service sends metrics by bulk to the *metric_collector* service. An example of aggregated metric derived from logs is the openstack_<service>_http_response_time_stats.

- Added a diagnostic tool.

  A diagnostic tool is now available to help diagnose issues. The diagnostic tool checks that the toolchain is properly installed and configured across the entire LMA toolchain. For more information, see *Diagnostic tool*.

Version 0.9.0

The StackLight Collector plugin 0.9.0 for Fuel contains the following updates:

- Upgraded to Heka *0.10.0*.
- Added the capability to collect libvirt metrics on compute nodes.
- Added the capability to detect spikes of errors in the OpenStack services logs.
- Added the capability to report OpenStack workers status per node.
- Added support for multi-environment deployments.
- Added support for Sahara logs and notifications.
- Bug fixes:
  - Added the capability to reconnect to the local RabbitMQ instance if the connection has been lost. See #1503251.
  - Enabled buffering for Elasticsearch, InfluxDB, Nagios and TCP outputs to reduce congestion in the Heka pipeline. See #1488717, #1557388.
  - Fixed the status for Nova when Midonet is used. See #1531541.
  - Fixed the status for Neutron when Contrail is used. See #1546017.
  - Increased the maximum number of file descriptors. See #1543289.
  - The spawning of several hekad processes is now avoided. See #1561109.
  - Removed the monitoring of individual queues of RabbitMQ. See #1549721.
  - Added the capability to rotate hekad logs every 30 minutes if necessary. See #1561603.

Version 0.8.0

The StackLight Collector plugin 0.8.0 for Fuel contains the following updates:

- Added support for alerting in two different modes:
  - Email notifications
  - Integration with Nagios
- Upgraded to InfluxDB 0.9.5.

- Upgraded to Grafana 2.5.
- Management of the LMA collector service by Pacemaker on the controller nodes for improved reliability.
- Monitoring of the LMA toolchain components (self-monitoring).
- Added support for configurable alarm rules in the Collector.

Version 0.7.0

The initial release of the StackLight Collector plugin. This is a beta version.

Licenses

Third-party components

Name	Project website	License
Heka	https://github.com/mozilla-services/heka	Mozilla Public License
collectd	http://collectd.org/	GPLv2
Collectd::CPU	http://collectd.org/	GPLv2
Collectd::Disk	http://collectd.org/	GPLv2
Collectd::Df	http://collectd.org/	GPLv2
Collectd::Interface	http://collectd.org/	GPLv2
Collectd::Load	http://collectd.org/	GPLv2
Collectd::Memory	http://collectd.org/	GPLv2
Collectd::Processes	http://collectd.org/	GPLv2 or later
Collectd::Swap	http://collectd.org/	GPLv2
Collectd::User	http://collectd.org/	GPLv2
Collectd::LogFile	http://collectd.org/	GPLv2
Collectd::User	http://collectd.org/	GPLv2
Collectd::WriteHttp	http://collectd.org/	GPLv2
Collectd::MySQL	http://collectd.org/	GPLv2
Collectd::DBI	http://collectd.org/	GPLv2
Collectd::Apache	http://collectd.org/	GPLv2
Collectd::Python	http://collectd.org/	MIT
Collectd::Python::RabbitMQ	http://collectd.org/	Apache v2
Collectd::Python::HAProxy	http://collectd.org/	Permissive

Puppet modules

Name	Project website	License
puppet-collectd	https://github.com/puppet-community/puppet-collectd	Apache v2
puppetlabs-apache	https://github.com/puppetlabs/puppetlabs-apache	Apache v2
puppetlabs-stdlib	https://github.com/puppetlabs/puppetlabs-stdlib	Apache v2
puppetlabs-inifile	https://github.com/puppetlabs/puppetlabs-inifile	Apache v2
puppetlabs-concat	https://github.com/puppetlabs/puppetlabs-concat	Apache v2
puppetlabs-firewall	https://github.com/puppetlabs/puppetlabs-firewall	Apache v2
openstack-cinder	https://github.com/openstack/puppet-cinder	Apache v2
openstack-glance	https://github.com/openstack/puppet-glance	Apache v2
openstack-heat	https://github.com/openstack/puppet-heat	Apache v2
openstack-keystone	https://github.com/openstack/puppet-keystone	Apache v2
openstack-neutron	https://github.com/openstack/puppet-neutron	Apache v2
openstack-nova	https://github.com/openstack/puppet-nova	Apache v2
openstack-openstacklib	https://github.com/openstack/puppet-openstacklib	Apache v2

References

- The StackLight Collector plugin project at GitHub
- The StackLight Elasticsearch-Kibana plugin project at GitHub
- The StackLight InfluxDB-Grafana plugin project at GitHub
- The StackLight Infrastructure Alerting plugin project at GitHub
- The official Kibana documentation
- The official Elasticsearch documentation
- The official InfluxDB documentation
- The official Grafana documentation
- The official Nagios documentation

CHAPTER 2

Installing StackLight Collector plugin for Fuel

Install using the RPM file of the Fuel plugins catalog

To install the StackLight Collector Fuel plugin using the RPM file of the Fuel plugins catalog:

1. Go to the Fuel plugins catalog.
2. From the *Filter* drop-down menu, select the Mirantis OpenStack version you are using and the *Monitoring* category.
3. Download the RPM file.
4. Copy the RPM file to the Fuel Master node:

```
[root@home ~]# scp lma_collector-1.1-1.1.0-1.noarch.rpm \
root@<Fuel Master node IP address>:
```

5. Install the plugin using the Fuel Plugins CLI:

```
[root@fuel ~]# fuel plugins --install lma_collector-1.1-1.1.0-1.noarch.rpm
```

6. Verify that the plugin is installed correctly:

```
[root@fuel ~]# fuel plugins --list
id | name                 | version | package_version
---|----------------------|---------|----------------
1  | lma_collector        | 1.1.0   | 4.0.0
```

Install from source

Alternatively, you may want to build the plugin RPM file from source if, for example, you want to test the latest features of the master branch or customize the plugin.

Note: Running a Fuel plugin that you built yourself is at your own risk and will not be supported.

To install the StackLight Collector Plugin from source, first prepare an environment to build the RPM file. The recommended approach is to build the RPM file directly onto the Fuel Master node so that you will not have to copy that file later on.

To prepare an environment and build the plugin:

1. Install the standard Linux development tools:

```
[root@home ~] yum install createrepo rpm rpm-build dpkg-devel
```

2. Install the Fuel Plugin Builder. To do that, you should first get pip:

```
[root@home ~] easy_install pip
```

3. Then install the Fuel Plugin Builder (the *fpb* command line) with *pip*:

```
[root@home ~] pip install fuel-plugin-builder
```

Note: You may also need to build the Fuel Plugin Builder if the package version of the plugin is higher than the package version supported by the Fuel Plugin Builder you get from `pypi`. For instructions on how to build the Fuel Plugin Builder, see the *Install Fuel Plugin Builder* section of the Fuel Plugin SDK Guide.

4. Clone the plugin repository:

```
[root@home ~] git clone https://github.com/openstack/fuel-plugin-lma-collector.git
```

5. Verify that the plugin is valid:

```
[root@home ~] fpb --check ./fuel-plugin-lma-collector
```

6. Build the plugin:

```
[root@home ~] fpb --build ./fuel-plugin-lma-collector
```

To install the plugin:

1. Once you have created the RPM file, install the plugin:

```
[root@fuel ~] fuel plugins --install ./fuel-plugin-lma-collector/*.noarch.rpm
```

2. Verify that the plugin is installed correctly:

```
[root@fuel ~]# fuel plugins --list
id | name                 | version | package_version
---|----------------------|---------|----------------
1  | lma_collector        | 1.1.0   | 4.0.0
```

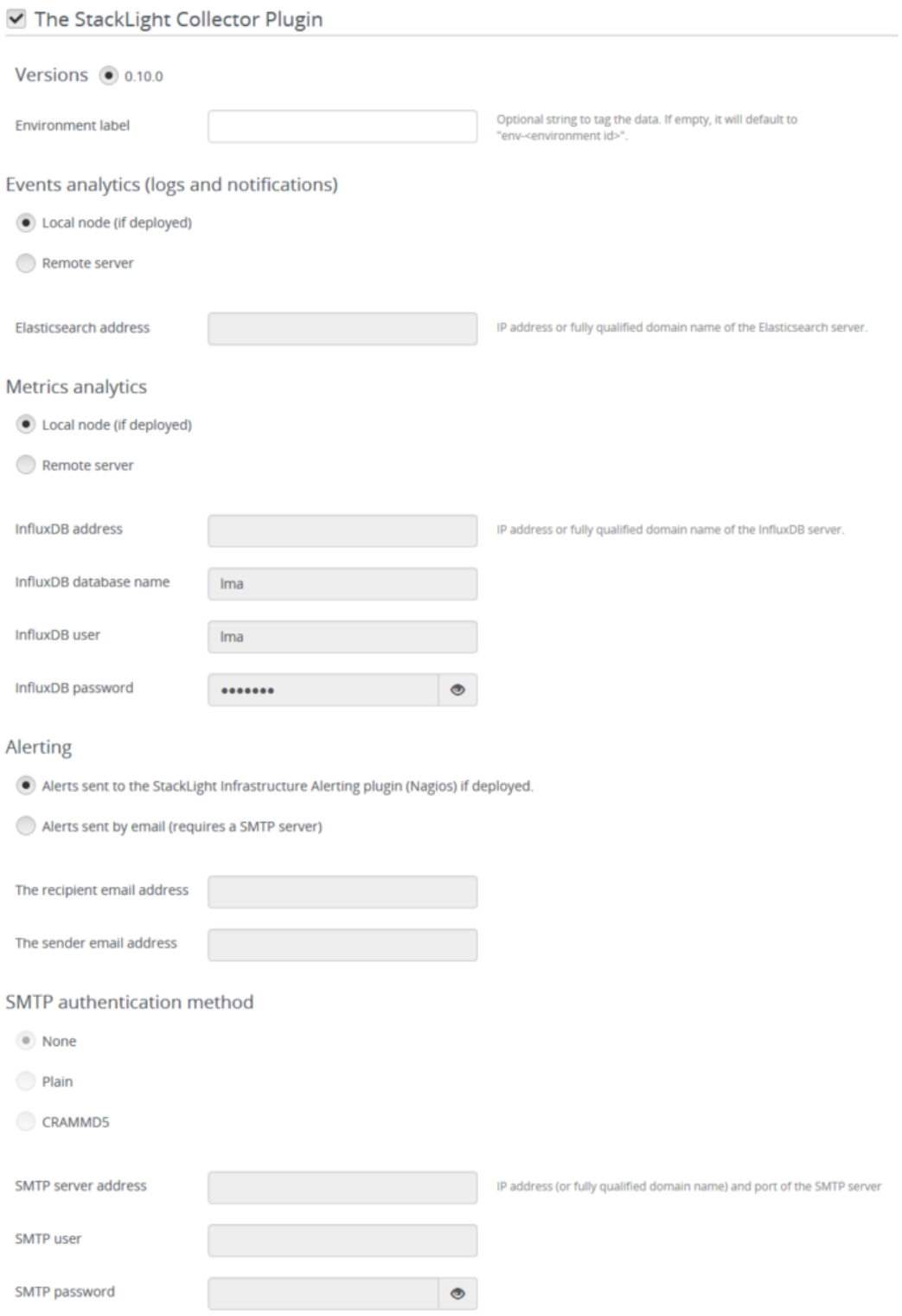
CHAPTER 3

Configuring StackLight Collector plugin for Fuel

Plugin configuration

To configure the StackLight Collector plugin:

1. Create a new environment as described in Create a new OpenStack environment.
2. In the Fuel web UI, click the *Settings* tab and select the *Other* category.
3. Scroll down through the settings until you find the StackLight Collector Plugin section. You should see a page like this:



4. Select *The Logging, Monitoring and Alerting (LMA) Collector Plugin* and fill in the required fields as indicated below.

   (a) Optional. Provide an *Environment Label* of your choice to tag your data.

   (b) In the *Events Analytics* section, select *Local node* if you plan to use the Elasticsearch-Kibana Plugin in the

environment. Otherwise, select *Remote server* and specify the fully qualified name or the IP address of an external Elasticsearch server.

(c) In the *Metrics Analytics* section, select *Local node* if you plan to use the InfluxDB-Grafana Plugin in the environment. Otherwise, select *Remote server* and specify the fully qualified name or the IP address of an external InfluxDB server. Then, specify the InfluxDB database name you want to use, a username and password that have read and write access permissions.

(d) In the *Alerting* section, select *Alerts sent by email* if you want to receive alerts sent by email from the Collector. Otherwise, select *Alerts sent to a local cluster* if you plan to use the Infrastructure Alerting Plugin (Nagios) in the environment. Alternatively, select *Alerts sent to a remote Nagios server*.

(e) For *Alerts sent by email*, specify the SMTP authentication method you want to use. Then, specify the SMTP server fully qualified name or IP address, the SMTP username and password to have the permissions to send emails.

5. Configure your environment as described in Configure your environment.

Note: By default, StackLight is configured to use the *management network* of the so-called *default node network group* created by Fuel. While this default setup may be appropriate for small deployments or evaluation purposes, it is recommended that you not use the default *management network* for StackLight. Instead, create a dedicated network when configuring your environment. This will improve the overall performance of both OpenStack and StackLight and facilitate the access to the Kibana and Grafana analytics.

---

6. Deploy your environment as described in Deploy an OpenStack environment.

Note: The StackLight Collector Plugin is a *hot-pluggable* plugin. Therefore, it is possible to install and deploy the *collector* in an environment that is already deployed. After the installation of the StackLight Collector Plugin, define the settings of the plugin and run the commands shown below from the *Fuel master node* for every node of your deployment starting with *the controller node(s)*:

```
[root@nailgun ~]# fuel nodes --env <env_id> --node <node_id> --tasks hiera␣
↪install-ocf-script
```

Once the task has completed for the node, run the following command:

```
[root@nailgun ~]# fuel nodes --env <env_id> --node <node_id> --start post_
↪deployment_start
```

Plugin verification

Once the OpenStack environment is ready, verify that both the *collectd* and *hekad* processes are running on the OpenStack nodes:

```
[root@node-1 ~]# pidof hekad
5568
5569
[root@node-1 ~]# pidof collectd
5684
```

Note: Starting with StackLight version 0.10, there are two *hekad* processes running instead of one. One is used to collect and process the logs and the notifications, the other one is used to process the metrics.

Troubleshooting

If you see no data in the Kibana and/or Grafana dashboards, follow the instructions below to troubleshoot the issue:

1. Verify that the *collector* services are up and running:
   - On the controller nodes:

     ```
     [root@node-1 ~]# crm resource status metric_collector
     [root@node-1 ~]# crm resource status log_collector
     ```

   - On non-controller nodes:

     ```
     [root@node-2 ~]# status log_collector
     [root@node-2 ~]# status metric_collector
     ```

2. If a *collector* is down, restart it:
   - On the controller nodes:

     ```
     [root@node-1 ~]# crm resource start log_collector
     [root@node-1 ~]# crm resource start metric_collector
     ```

   - On non-controller nodes:

     ```
     [root@node-2 ~]# start log_collector
     [root@node-2 ~]# start metric_collector
     ```

3. Look for errors in the log file of the *collectors* located at `/var/log/log_collector.log` and `/var/log/metric_collector.log`.
4. Look for errors in the log file of *collectd* located at `/var/log/collectd.log`.
5. Verify that the nodes are able to connect to the Elasticsearch server on port 9200.
6. Verify that the nodes are able to connect to the InfluxDB server on port 8086.

Diagnostic tool

The StackLight Collector Plugin installs a **global diagnostic tool** on the Fuel Master node. The global diagnostic tool checks that StackLight is configured and running properly across the entire LMA toolchain for all the nodes that are ready in your OpenStack environment:

```
[root@nailgun ~]# /var/www/nailgun/plugins/lma_collector-<version>/contrib/tools/
↪diagnostic.sh
Running lma_diagnostic tool on all available nodes (this can take several minutes)
The diagnostic archive is here: /var/lma_diagnostics.2016-06-10_11-23-1465557820.tgz
```

Note: A global diagnostic can take several minutes.

All the results are consolidated in the `/var/lma_diagnostics.[date +%Y-%m-%d_%H-%M-%s].tgz` archive.

Instead of running a global diagnostic, you may want to run the diagnostic on individual nodes. Based on the role of the node, the tool determines what checks should be executed. For example:

```
root@node-3:~# hiera roles
["controller"]

root@node-3:~# lma_diagnostics

2016-06-10-11-08-04 INFO node-3.test.domain.local role ["controller"]
2016-06-10-11-08-04 INFO ** LMA Collector
2016-06-10-11-08-04 INFO 2 process(es) 'hekad -config' found
2016-06-10-11-08-04 INFO 1 process(es) hekad is/are listening on port 4352
2016-06-10-11-08-04 INFO 1 process(es) hekad is/are listening on port 8325
2016-06-10-11-08-05 INFO 1 process(es) hekad is/are listening on port 5567
2016-06-10-11-08-05 INFO 1 process(es) hekad is/are listening on port 4353
[...]
```

In the example above, the diagnostic tool reports that two *hekad* processes are running on *node-3*, which is the expected outcome. In the case when one *hekad* process is not running, the diagnostic tool reports an error. For example:

```
root@node-3:~# lma_diagnostics
2016-06-10-11-11-48 INFO node-3.test.domain.local role ["controller"]
2016-06-10-11-11-48 INFO ** LMA Collector
2016-06-10-11-11-48 ERROR 1 'hekad -config' processes found, 2 expected!
2016-06-10-11-11-48 ERROR 'hekad' process does not LISTEN on port: 4352
[...]
```

In the example above, the diagnostic tool reported two errors:

1. There is only one *hekad* process running instead of two.
2. No *hekad* process is listening on port 4352.

These examples describe only one type of checks performed by the diagnostic tool, but there are many others.

On the OpenStack nodes, the diagnostic results are stored in `/var/lma_diagnostics/diagnostics.log`.

Note: A successful LMA toolchain diagnostic should be free of errors.

Advanced configuration

Due to a current limitation in Fuel, when a node is removed from an OpenStack environment through the Fuel web UI or CLI, the services that were running on that node are not automatically removed from the database. Therefore, StackLight reports these services as failed. To resolve this issue, remove these services manually.

To reconfigure the StackLight Collector after removing a node:

1. From a controller node, list the services that are reported failed. In the example below, it is `node-7`.

```
root@node-6:~# source ./openrc
root@node-6:~# neutron agent-list
+--------------+------------------+------------------+-------------------+------
↪-+
| id           | agent_type       | host             | availability_zone |␣
↪alive |
+--------------+------------------+------------------+-------------------+------
↪-+
| 08a69bad-... | Metadata agent   | node-8.domain.tld |                   | :-) ␣
↪ |
| 11b6dca6-... | Metadata agent   | node-7.domain.tld |                   | xxx ␣
↪ |
| 22ea82e3-... | DHCP agent       | node-6.domain.tld | nova              | :-) ␣
↪ |
| 2d82849e-... | L3 agent         | node-6.domain.tld | nova              | :-) ␣
↪ |
| 3221ec18-... | Open vSwitch agent| node-6.domain.tld |                   | :-) ␣
↪ |
| 84bfd240-... | Open vSwitch agent| node-7.domain.tld |                   | xxx ␣
↪ |
| 9452e8f0-... | Open vSwitch agent| node-9.domain.tld |                   | :-) ␣
↪ |
| 97136b09-... | Open vSwitch agent| node-8.domain.tld |                   | :-) ␣
↪ |
| c198bc94-... | DHCP agent       | node-7.domain.tld | nova              | xxx ␣
↪ |
| c76c4ed4-... | L3 agent         | node-7.domain.tld | nova              | xxx ␣
↪ |
| d0fd8bb5-... | L3 agent         | node-8.domain.tld | nova              | :-) ␣
↪ |
| d21f9cea-... | DHCP agent       | node-8.domain.tld | nova              | :-) ␣
↪ |
| f6f871b7-... | Metadata agent   | node-6.domain.tld |                   | :-) ␣
↪ |
+--------------+------------------+------------------+-------------------+------
↪-+
root@node-6:~# nova service-list
+--+----------------+----------------+---------+--------+-------+----------------
↪-+
|Id|Binary          |Host            | Zone    | Status | State |   Updated_at  ␣
↪ |
+--+----------------+----------------+---------+--------+-------+----------------
↪-+
|1 |nova-consoleauth|node-6.domain.tld| internal| enabled| up    | 2016-07-
↪19T11:43|
|4 |nova-scheduler  |node-6.domain.tld| internal| enabled| up    | 2016-07-
↪19T11:43|
|7 |nova-cert       |node-6.domain.tld| internal| enabled| up    | 2016-07-
↪19T11:43|
```

```
|10|nova-conductor  |node-6.domain.tld| internal| enabled| up    | 2016-07-
↪19T11:42|
|22|nova-cert       |node-7.domain.tld| internal| enabled| down  | 2016-07-
↪19T11:43|
|25|nova-consoleauth|node-7.domain.tld| internal| enabled| down  | 2016-07-
↪19T11:43|
|28|nova-scheduler  |node-7.domain.tld| internal| enabled| down  | 2016-07-
↪19T11:43|
|31|nova-cert       |node-8.domain.tld| internal| enabled| up    | 2016-07-
↪19T11:43|
|34|nova-consoleauth|node-8.domain.tld| internal| enabled| up    | 2016-07-
↪19T11:43|
|37|nova-conductor  |node-7.domain.tld| internal| enabled| down  | 2016-07-
↪19T11:42|
|43|nova-scheduler  |node-8.domain.tld| internal| enabled| up    | 2016-07-
↪19T11:43|
|49|nova-conductor  |node-8.domain.tld| internal| enabled| up    | 2016-07-
↪19T11:42|
|64|nova-compute    |node-9.domain.tld| nova    | enabled| up    | 2016-07-
↪19T11:42|
+--+----------------+-----------------+---------+--------+-------+----------------
↪-+
root@node-6:~# cinder service-list
+----------------+--------------------------+----+-------+-----+----------------
↪-+
|    Binary      |            Host          |Zone| Status|State|   Updated_at ␣
↪ |
+----------------+--------------------------+----+-------+-----+----------------
↪-+
|cinder-backup   |       node-9.domain.tld  |nova|enabled|up   |2016-07-
↪19T11:44|
|cinder-scheduler|       node-6.domain.tld  |nova|enabled|up   |2016-07-
↪19T11:43|
|cinder-scheduler|       node-7.domain.tld  |nova|enabled|down |2016-07-
↪19T11:43|
|cinder-scheduler|       node-8.domain.tld  |nova|enabled|up   |2016-07-
↪19T11:44|
|cinder-volume   |node-9.domain.tld@LVM-backend|nova|enabled|up   |2016-07-
↪19T11:44|
+----------------+--------------------------+----+-------+-----+----------------
↪-+
```

2. Remove the services and/or agents that are reported failed on that node:

```
root@node-6:~# nova service-delete <id of service to delete>
root@node-6:~# cinder service-disable <hostname> <binary>
root@node-6:~# neutron agent-delete <id of agent to delete>
```

3. Restart the Collector on all the controller nodes:

```
[root@node-1 ~]# crm resource restart log_collector
[root@node-1 ~]# crm resource restart metric_collector
```

CHAPTER 4

Configuring alarms

Overview

The process of running alarms in StackLight is not centralized, as it is often the case in more conventional monitoring systems, but distributed across all the StackLight Collectors.

Each Collector is individually responsible for monitoring the resources and services that are deployed on the node and for reporting any anomaly or fault it has detected to the Aggregator.

The anomaly and fault detection logic in StackLight is designed more like an *expert system* in that the Collector and the Aggregator use artifacts we can refer to as *facts* and *rules*.

The *facts* are the operational data ingested in the StackLight's stream-processing pipeline. The *rules* are either alarm rules or aggregation rules. They are declaratively defined in YAML files that can be modified. Those rules are turned into a collection of Lua plugins that are executed by the Collector and the Aggregator. They are generated dynamically using the Puppet modules of the StackLight Collector Plugin.

The following are the two types of Lua plugins related to the processing of alarms:

- The **AFD plugin** – Anomaly and Fault Detection plugin
- The **GSE plugin** – Global Status Evaluation plugin

These plugins create special types of metrics, as follows:

- The **AFD metric**, which contains information about the health status of a node or service in the OpenStack environment. The AFD metrics are sent on a regular basis to the Aggregator where they are further processed by the GSE plugins.
- The **GSE metric**, which contains information about the health status of a cluster in the OpenStack environment. A cluster is a logical grouping of nodes or services. We call them node clusters and service clusters hereafter. A service cluster can be anything like a cluster of API endpoints or a cluster of workers. A cluster of nodes is a grouping of nodes that have the same role. For example, *compute* or *storage*.

Note: The AFD and GSE metrics are new types of metrics introduced in StackLight version 0.8. They contain detailed information about the fault and anomalies detected by StackLight. For more information about the message

structure of these metrics, refer to Metrics section of the Developer Guide.

The following figure shows the StackLight stream-processing pipeline workflow:

The AFD and GSE plugins

The current version of StackLight contains the following three types of GSE plugins:

- The **Service Cluster GSE Plugin**, which receives AFD metrics for services from the AFD plugins.
- The **Node Cluster GSE Plugin**, which receives AFD metrics for nodes from the AFD plugins.
- The **Global Cluster GSE Plugin**, which receives GSE metrics from the GSE plugins above. It aggregates and correlates the GSE metrics to issue a global health status for the top-level clusters like Nova, MySQL, and others.

The health status exposed in the GSE metrics is as follows:

- `Down`: One or several primary functions of a cluster has failed or is failing. For example, the API service for Nova or Cinder is not accessible.
- `Critical`: One or several primary functions of a cluster are severely degraded. The quality of service delivered to the end user is severely impacted.
- `Warning`: One or several primary functions of the cluster are slightly degraded. The quality of service delivered to the end user is slightly impacted.
- `Unknown`: There is not enough data to infer the actual health status of the cluster.
- `Okay`: None of the above was found to be true.

The AFD and GSE persisters

The AFD and GSE metrics are also consumed by other types of Lua plugins called **persisters**:

- The **InfluxDB persister** transforms the GSE metrics into InfluxDB data points and Grafana annotations. They are used in Grafana to graph the health status of the OpenStack clusters.
- The **Elasticsearch persister** transforms the AFD metrics into events that are indexed in Elasticsearch. Using Kibana, these events can be searched to display a fault or an anomaly that occurred in the environment (not yet implemented).
- The **Nagios persister** transforms the GSE and AFD metrics into passive checks that are sent to Nagios for alerting and escalation.

New persisters can be easily created to feed other systems with the operational insight contained in the AFD and GSE metrics.

Alarms configuration

StackLight comes with a predefined set of alarm rules. We have tried to make these rules as comprehensive and relevant as possible, but your mileage may vary depending on the specifics of your OpenStack environment and monitoring requirements. Therefore, it is possible to modify those predefined rules and create new ones. To do so, modify the `/etc/hiera/override/alarming.yaml` file and apply the *Puppet manifest* that will dynamically generate Lua plugins, known as the AFD Plugins, which are the actuators of the alarm rules. But before you proceed, verify that understand the structure of that file.

Alarm structure

An alarm rule is defined declaratively using the YAML syntax. For example:

```
name: 'fs-warning'
description: 'Filesystem free space is low'
severity: 'warning'
enabled: 'true'
trigger:
  rules:
    - metric: fs_space_percent_free
      group_by: ['fs']
      relational_operator: '<'
      fields:
        fs: "=~ ceph/%d+$"
      threshold: 5
      window: 60
      periods: 0
      function: min
```

Where

name:
: Type: unicode
: The name of the alarm definition

description:
: Type: unicode
: A description of the alarm definition for humans

severity:
: Type: Enum(0 (down), 1 (critical) , 2 (warning))
: The severity of the alarm

enabled:
: Type: Enum('true' | 'false')
: The alarm is enabled or disabled

relational_operator:
: Type: Enum('lt' | '<' | 'gt' | '>' | 'lte' | '<=' | 'gte' | '>=')
: The comparison against the alarm threshold

rules
: Type: list
: List of rules to execute

logical_operator

Type: Enum('and' | '&&' | 'or' | '||')
The conjunction relation for the alarm rules

metric
Type: unicode
The name of the metric

value
Type: unicode
The value of the metric

group_by
Type: list
The list of fields to group by. For example, the alarm definition sample given above would apply the rule for each of the file system mount points associated with the `fs_space_percent_free` metric.

fields
Type: list
List of field name/value pairs, also known as dimensions, used to select a particular device for the metric, such as a network interface name or file system mount point. If the value is not provided, then the rule applies to all the aggregated values for the specified field name. In the example above, the rule applies to the metrics having an **fs** dimension that matches the pattern **"=~ ceph/%d+$"**. See *Dimension pattern matching*.

window
Type: integer
The in-memory time-series analysis window in seconds

periods
Type: integer
The number of prior time-series analysis window to compare the window with
(this is not implemented yet).

function
Type: enum('last' | 'min' | 'max' | 'sum' | 'count' | 'avg' | 'median' | 'mode' | 'roc' | 'mww' | 'mww_nonparametric')
Where:
last:
returns the last value of all the values
min:
returns the minimum of all the values
max:
returns the maximum of all the values
sum:

returns the sum of all the values

count:

returns the number of metric observations

avg:

returns the arithmetic mean of all the values

median:

returns the middle value of all the values (not implemented yet)

mode:

returns the value that occurs most often in all the values

(not implemented yet)

roc:

The 'roc' function detects a significant rate of change when comparing current metrics values with historical data. To achieve this, it computes the average of the values in the current window and the average of the values in the window before the current window and compares the difference against the standard deviation of the historical window. The function returns `true` if the difference exceeds the standard deviation multiplied by the 'threshold' value. This function uses the rate of change algorithm already available in the anomaly detection module of Heka. It can only be applied to normal distributions. With an alarm rule using the 'roc' function, the 'window' parameter specifies the duration in seconds of the current window, and the 'periods' parameter specifies the number of windows used for the historical data. You need at least one period and the 'periods' parameter must not be zero. If you choose a period of 'p', the function will compute the rate of change using a historical data window of ('p' * window) seconds. For example, if you specify the following in the alarm rule:

window = 60
periods = 3
threshold = 1.5

the function will store in a circular buffer the value of the metrics received during the last 300 seconds (5 minutes) where:

Current window (CW) = 60 sec
Previous window (PW) = 60 sec
Historical window (HW) = 180 sec

and apply the following formula:

abs(avg(CW) - avg(PW)) > std(HW) * 1.5 ? true : false

mww:

returns the result (true, false) of the Mann-Whitney-Wilcoxon test function of Heka that can be used only with normal distributions (not implemented yet)

mww-nonparametric:

returns the result (true, false) of the Mann-Whitney-Wilcoxon test function of Heka that can be used with non-normal distributions (not implemented yet)

diff:

returns the difference between the last value and the first value of all the values

threshold

Type: float
The threshold of the alarm rule

Dimension pattern matching

The alarming framework allows specifying alarms against metrics with a filtering mechanism called **dimension pattern matching**. For details, see the specification.

The pattern matching is evaluated against the field/dimension specified by the alarm rule:

```
rules:
  - metric: foo_metric
    fields:
      my_dimension: <PATTERN MATCHING EXPRESSION>
```

where the pattern matching expression has the following format:

```
EXP ::=  "<relation operator> string"
```

Expressions can be combined with logical operator(s):

```
EXP (<logical_operator> EXP, ..)
```

Where:

- Logical operators:
  - OR: **||**
  - AND: **&&**
- Relational operators:
  - Strings and numbers:
    - Equality: **==**
    - Not equal: **!=**
  - Strings only (for syntax, see Lua pattern matching):
    - Match: **=~**
    - Negated match: **!~**
  - Numbers only:
    - Greater than: **>**
    - Greater than equals: **>=**
    - Less than: **<**
    - Less than equals: **<=**

Example:

Value	Pattern	Matching
10	10	True
10	== 10	True
10	!= 42	True
10	> 42	False
10	>= 10	True
foo	== foo	True
foo	== bar	False
/var/log	=~ /log$	True
/data	!~ ^/data$	False
/var/log/data	!~ /data$	False
/var/log/data	!~ ^/data$	True

Modify or create an alarm

To modify or create an alarm, edit the `/etc/hiera/override/alarming.yaml` file. This file has the following sections:

1. The `alarms` section contains a global list of alarms that are executed by the Collectors. These alarms are global to the LMA toolchain and should be kept identical on all nodes of the OpenStack environment. The following is another example of the definition of an alarm:

```
alarms:
  - name: 'cpu-critical-controller'
    description: 'CPU critical on controller'
    severity: 'critical'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
         - metric: cpu_idle
           relational_operator: '<='
           threshold: 5
           window: 120
           periods: 0
           function: avg
         - metric: cpu_wait
           relational_operator: '>='
           threshold: 35
           window: 120
           periods: 0
           function: avg
```

   This alarm is called 'cpu-critical-controller'. It says that CPU activity is critical (severity: 'critical') if any of the rules in the alarm definition evaluate to true.

   The rule says that the alarm will evaluate to 'true' if the value of the metric `cpu_idle` has been in average (function: avg), below or equal (relational_operator: <=) to 5 for the last 2 minutes (window: 120).

   OR (logical_operator: 'or')

   If the value of the metric **cpu_wait** has been in average (function: avg), superior or equal (relational_operator: >=) to 35 for the last 2 minutes (window: 120)

   Note that these metrics are expressed in percentage.

   What alarms are executed on which node depends on the mapping between the alarm definition and the definition of a cluster as described in the following sections.

2. The `node_cluster_roles` section defines the mapping between the internal definition of a cluster of nodes and one or several Fuel roles. For example:

```
node_cluster_roles:
  controller: ['primary-controller', 'controller']
  compute: ['compute']
  storage: ['cinder', 'ceph-osd']
  [ ... ]
```

   Creates a mapping between the 'primary-controller' and 'controller' Fuel roles, and the internal definition of a cluster of nodes called 'controller'. Likewise, the internal definition of a cluster of nodes called 'storage' is mapped to the 'cinder' and 'ceph-osd' Fuel roles. The internal definition of a cluster of nodes is used to assign the alarms to the relevant category of nodes. This mapping is also used to configure the **passive checks** in Nagios. Therefore, it is critically important to keep exactly the same copy of `/etc/hiera/override/alarming.yaml` across all nodes of the OpenStack environment including the node(s) where Nagios is installed.

3. The `service_cluster_roles` section defines the mapping between the internal definition of a cluster of services and one or several Fuel roles. For example:

```
service_cluster_roles:
  rabbitmq: ['primary-controller', 'controller']
  nova-api: ['primary-controller', 'controller']
  elasticsearch: ['primary-elasticsearch_kibana', 'elasticsearch_kibana']
  [ ... ]
```

   Creates a mapping between the 'primary-controller' and 'controller' Fuel roles, and the internal definition of a cluster of services called 'rabbitmq'. Likewise, the internal definition of a cluster of services called 'elasticsearch' is mapped to the 'primary-elasticsearch_kibana' and 'elasticsearch_kibana' Fuel roles. As for the clusters of nodes, the internal definition of a cluster of services is used to assign the alarms to the relevant category of services.

4. The `node_cluster_alarms` section defines the mapping between the internal definition of a cluster of nodes and the alarms that are assigned to that category of nodes. For example:

```
node_cluster_alarms:
   controller-nodes:
       apply_to_node: controller
       alerting: enabled
       members:
           cpu:
               alarms: ['cpu-critical-controller', 'cpu-warning-controller']
           root-fs:
               alarms: ['root-fs-critical', 'root-fs-warning']
           log-fs:
               alarms: ['log-fs-critical', 'log-fs-warning']
           hdd-errors:
               alerting: enabled_with_notification
               alarms: ['hdd-errors-critical']
```

   Creates four alarm groups for the cluster of controller nodes:

   - The *cpu* alarm group is mapped to two alarms defined in the `alarms` section known as the 'cpu-critical-controller' and 'cpu-warning-controller' alarms. These alarms monitor the CPU on the controller nodes. The order matters here since the first alarm that evaluates to 'true' stops the evaluation. Therefore, it is important to start the list with the most critical alarms.
   - The *root-fs* alarm group is mapped to two alarms defined in the `alarms` section known as the 'root-fs-critical' and 'root-fs-warning' alarms. These alarms monitor the root file system on the controller nodes.

- The *log-fs* alarm group is mapped to two alarms defined in the `alarms` section known as the 'log-fs-critical' and 'log-fs-warning' alarms. These alarms monitor the file system where the logs are created on the controller nodes.
- The *hdd-errors* alarm group is mapped to the 'hdd-errors-critical' alarm defined in the `alarms` section. This alarm monitors the `kern.log` log entries containing critical IO errors detected by the kernel. The *hdd-error* alarm obtains the *enabled_with_notification* alerting attribute, meaning that the operator will be notified if any of the controller nodes encounters a disk failure. Other alarms do not trigger notification per node but at an aggregated cluster level.

Note: An *alarm group* is a mere implementation artifact (although it has functional value) that is primarily used to distribute the alarms evaluation workload across several Lua plugins. Since the Lua plugins runtime is sandboxed within Heka, it is preferable to run smaller sets of alarms in different plugins rather than a large set of alarms in a single plugin. This is to avoid having alarms evaluation plugins shut down by Heka. Furthermore, the alarm groups are used to identify what is called a *source*. A *source* is a tuple in which we associate a cluster with an alarm group. For example, the tuple ['controller', 'cpu'] is a *source*. It associates a 'controller' cluster with the 'cpu' alarm group. The tuple ['controller', 'root-fs'] is another *source* example. The *source* is used by the GSE Plugins to remember the AFD metrics it has received. If a GSE Plugin stops receiving AFD metrics it used to get, then the GSE Plugin infers that the health status of the cluster associated with the source is *Unknown*.

This is evaluated every *ticker-interval*. By default, the *ticker interval* for the GSE Plugins is set to 10 seconds.

Aggregation and correlation configuration

StackLight comes with a predefined set of aggregation rules and correlation policies. However, you can create new aggregation rules and correlation policies or modify the existing ones. To do so, modify the `/etc/hiera/override/gse_filters.yaml` file and apply the *Puppet manifest* that will generate Lua plugins known as the GSE Plugins, which are the actuators of these aggregation rules and correlation policies. But before you proceed, verify that you understand the structure of that file.

Note: As for `/etc/hiera/override/alarming.yaml`, it is critically important to keep exactly the same copy of `/etc/hiera/override/gse_filters.yaml` across all the nodes of the OpenStack environment including the node(s) where Nagios is installed.

The aggregation rules and correlation policies are defined in the `/etc/hiera/override/gse_filters.yaml` configuration file.

This file has the following sections:

1. The `gse_policies` section contains the *health status correlation policies* that apply to the node clusters and service clusters.
2. The `gse_cluster_service` section contains the *aggregation rules* for the service clusters. These aggregation rules are actuated by the Service Cluster GSE Plugin that runs on the Aggregator.
3. The `gse_cluster_node` section contains the *aggregation rules* for the node clusters. These aggregation rules are actuated by the Node Cluster GSE Plugin that runs on the Aggregator.
4. The `gse_cluster_global` section contains the *aggregation rules* for the so-called top-level clusters. A global cluster is a kind of logical construct of node clusters and service clusters. These aggregation rules are actuated by the Global Cluster GSE Plugin that runs on the Aggregator.

Health status policies

The correlation logic implemented by the GSE plugins is policy-based. The policies define how the GSE plugins infer the health status of a cluster.

By default, there are two policies:

- The **highest_severity** policy defines that the cluster's status depends on the member with the highest severity, typically used for a cluster of services.
- The **majority_of_members** policy defines that the cluster is healthy as long as (N+1)/2 members of the cluster are healthy. This is typically used for clusters managed by Pacemaker.

A policy consists of a list of rules that are evaluated against the current status of the cluster's members. When one of the rules matches, the cluster's status gets the value associated with the rule and the evaluation stops. The last rule of the list is usually a catch-all rule that defines the default status if none of the previous rules matches.

The following example shows the policy rule definition:

```
# The following rule definition reads as: "the cluster's status is critical
# if more than 50% of its members are either down or critical"
- status: critical
  trigger:
    logical_operator: or
    rules:
      - function: percent
        arguments: [ down, critical ]
        relational_operator: '>'
        threshold: 50
```

Where

status:
: Type: Enum(down, critical, warning, okay, unknown)
: The cluster's status if the condition is met

logical_operator
: Type: Enum('and' | '&&' | 'or' | '||')
: The conjunction relation for the condition rules

rules
: Type: list
: List of condition rules to execute

function
: Type: enum('count' | 'percent')
: Where:
: count:
: returns the *number of members* that match the passed value(s).
: percent:
: returns the *percentage of members* that match the passed value(s).

arguments:
: Type: list of status values
: List of status values passed to the function

relational_operator:
: Type: Enum('lt' | '<' | 'gt' | '>' | 'lte' | '<=' | 'gte' | '>=')
: The comparison against the threshold

threshold
: Type: float
: The threshold value

Consider the policy called *highest_severity*:

```
gse_policies:

  highest_severity:
    - status: down
      trigger:
        logical_operator: or
        rules:
          - function: count
            arguments: [ down ]
            relational_operator: '>'
            threshold: 0
    - status: critical
      trigger:
        logical_operator: or
        rules:
          - function: count
            arguments: [ critical ]
            relational_operator: '>'
            threshold: 0
    - status: warning
      trigger:
        logical_operator: or
        rules:
          - function: count
            arguments: [ warning ]
            relational_operator: '>'
            threshold: 0
    - status: okay
      trigger:
        logical_operator: or
        rules:
          - function: count
            arguments: [ okay ]
            relational_operator: '>'
            threshold: 0
    - status: unknown
```

The policy definition reads as follows:

- The status of the cluster is `Down` if the status of at least one cluster's member is `Down`.

- Otherwise, the status of the cluster is `Critical` if the status of at least one cluster's member is `Critical`.
- Otherwise, the status of the cluster is `Warning` if the status of at least one cluster's member is `Warning`.
- Otherwise, the status of the cluster is `Okay` if the status of at least one cluster's entity is *Okay*.
- Otherwise, the status of the cluster is `Unknown`.

Service cluster aggregation rules

The service cluster aggregation rules are used to designate the members of a service cluster along with the AFD metrics that must be taken into account to derive a health status for the service cluster. The following is an example of the service cluster aggregation rules:

```
gse_cluster_service:
  input_message_types:
    - afd_service_metric
  aggregator_flag: true
  cluster_field: service
  member_field: source
  output_message_type: gse_service_cluster_metric
  output_metric_name: cluster_service_status
  interval: 10
  warm_up_period: 20
  alerting: enabled_with_notification
  clusters:
    nova-api:
      policy: highest_severity
      group_by: member
      members:
        - backends
        - endpoint
        - http_errors
```

Where

input_message_types
: Type: list
  The type(s) of AFD metric messages consumed by the GSE plugin.

aggregator_flag
: Type: boolean
  Whether or not the input messages are received from the upstream collectors. This is true for the Service and Node Cluster plugins and false for the Global Cluster plugin.

cluster_field
: Type: unicode
  The field in the input message used by the GSE plugin to associate the AFD metrics to the clusters.

member_field
: Type: unicode

The field in the input message used by the GSE plugin to identify the cluster members.

output_message_type
: Type: unicode
The type of metric messages emitted by the GSE plugin.

output_metric_name
: Type: unicode
The Fields[name] value of the metric messages that the GSE plugin emits.

interval
: Type: integer
The interval (in seconds) at which the GSE plugin emits its metric messages.

warm_up_period
: Type: integer
The number of seconds after a (re)start that the GSE plugin will wait before emitting its metric messages.

alerting
: Type: string (one of 'disabled', 'enabled' or 'enabled_with_notification').
The alerting configuration of the service clusters.

clusters
: Type: list
The list of service clusters that the plugin handles. See *Service cluster definition* for details.

Service cluster definition

The following example shows the service clusters definition:

```
gse_cluster_service:
  [...]
  clusters:
    nova-api:
      members:
        - backends
        - endpoint
        - http_errors
      group_by: member
      policy: highest_severity
```

Where

members
: Type: list

The list of cluster members. The AFD messages that are associated with the cluster when the `cluster_field` value is equal to the cluster name and the `member_field` value is in this list.

group_by
: Type: Enum(member, hostname)
This parameter defines how the incoming AFD metrics are aggregated.

  member:
  : aggregation by member, irrespective of the host that emitted the AFD metric. This setting is typically used for AFD metrics that are not host-centric.

  hostname:
  : aggregation by hostname then by member.
  This setting is typically used for AFD metrics that are host-centric, such as those working on the file system or CPU usage metrics.

policy:
: Type: unicode
The policy to use for computing the service cluster status. See *Health status policies* for details.

A closer look into the example above defines that the Service Cluster GSE plugin resulting from those rules will emit a *gse_service_cluster_metric* message every 10 seconds to report the current status of the *nova-api* cluster. This status is computed using the *afd_service_metric* metric for which Fields[service] is 'nova-api' and Fields[source] is one of 'backends', 'endpoint', or 'http_errors'. The 'nova-api' cluster's status is computed using the 'highest_severity' policy, which means that it will be equal to the 'worst' status across all members.

Node cluster aggregation rules

The node cluster aggregation rules are used to designate the members of a node cluster along with the AFD metrics that must be taken into account to derive a health status for the node cluster. The following is an example of the node cluster aggregation rules:

```
gse_cluster_node:
  input_message_types:
    - afd_node_metric
  aggregator_flag: true
  # the field in the input messages to identify the cluster
  cluster_field: node_role
  # the field in the input messages to identify the cluster's member
  member_field: source
  output_message_type: gse_node_cluster_metric
  output_metric_name: cluster_node_status
  interval: 10
  warm_up_period: 80
  alerting: enabled_with_notification
  clusters:
    controller:
      policy: majority_of_members
```

```
      group_by: hostname
      members:
        - cpu
        - root-fs
        - log-fs
```

Where

input_message_types
: Type: list
The type(s) of AFD metric messages consumed by the GSE plugin.

aggregator_flag
: Type: boolean
Whether or not the input messages are received from the upstream collectors. This is true for the Service and Node Cluster plugins and false for the Global Cluster plugin.

cluster_field
: Type: unicode
The field in the input message used by the GSE plugin to associate the AFD metrics to the clusters.

member_field
: Type: unicode
The field in the input message used by the GSE plugin to identify the cluster members.

output_message_type
: Type: unicode
The type of metric messages emitted by the GSE plugin.

output_metric_name
: Type: unicode
The Fields[name] value of the metric messages that the GSE plugin emits.

interval
: Type: integer
The interval (in seconds) at which the GSE plugin emits its metric messages.

warm_up_period
: Type: integer
The number of seconds after a (re)start that the GSE plugin will wait before emitting its metric messages.

alerting

Type: string (one of 'disabled', 'enabled' or 'enabled_with_notification').
The alerting configuration of the node clusters.

clusters
: Type: list
The list of node clusters that the plugin handles. See *Node cluster definition* for details.

Node cluster definition

The following example shows the node clusters definition:

```
gse_cluster_node:
  [...]
  clusters:
    controller:
      policy: majority_of_members
      group_by: hostname
      members:
        - cpu
        - root-fs
        - log-fs
```

Where

members
: Type: list
The list of cluster members. The AFD messages are associated to the cluster when the `cluster_field` value is equal to the cluster name and the `member_field` value is in this list.

group_by
: Type: Enum(member, hostname)
This parameter defines how the incoming AFD metrics are aggregated.

member:
: aggregation by member, irrespective of the host that emitted the AFD metric. This setting is typically used for AFD metrics that are not host-centric.

hostname:
: aggregation by hostname then by member.
This setting is typically used for AFD metrics that are host-centric, such as those working on the file system or CPU usage metrics.

policy:
: Type: unicode
The policy to use for computing the node cluster status. See *Health status policies* for details.

A closer look into the example above defines that the Node Cluster GSE plugin resulting from those rules will emit a *gse_node_cluster_metric* message every 10 seconds to report the current status of the *controller* cluster. This status is computed using the *afd_node_metric* metric for which Fields[node_role] is 'controller' and Fields[source] is one of 'cpu', 'root-fs' or 'log-fs'. The 'controller' cluster's status is computed using the 'majority_of_members' policy which means that it will be equal to the 'majority' status across all members.

Top-level cluster aggregation rules

The top-level aggregation rules aggregate GSE metrics from the Service Cluster GSE Plugin and the Node Cluster GSE Plugin. This is the last aggregation stage that issues health status for the top-level clusters. A top-level cluster is a logical construct of service and node clustering. By default, we define that the health status of Nova, as a top-level cluster, depends on the health status of several service clusters related to Nova and the health status of the 'controller' and 'compute' node clusters. But it can be anything. For example, you can define a 'control-plane' top-level cluster that would exclude the health status of the 'compute' node cluster if required. The top-level cluster aggregation rules are used to designate the node clusters and service clusters members of a top-level cluster along with the GSE metrics that must be taken into account to derive a health status for the top-level cluster. The following is an example of a top-level cluster aggregation rules:

```
gse_cluster_global:
  input_message_types:
    - gse_service_cluster_metric
    - gse_node_cluster_metric
  aggregator_flag: false
  # the field in the input messages to identify the cluster's member
  member_field: cluster_name
  output_message_type: gse_cluster_metric
  output_metric_name: cluster_status
  interval: 10
  warm_up_period: 30
  clusters:
    nova:
      policy: highest_severity
      group_by: member
      members:
        - nova-logs
        - nova-api
        - nova-metadata-api
        - nova-scheduler
        - nova-compute
        - nova-conductor
        - nova-cert
        - nova-consoleauth
        - nova-novncproxy-websocket
        - controller
        - compute
      hints:
        - cinder
        - glance
        - keystone
        - neutron
        - mysql
        - rabbitmq
```

Where

input_message_types
: Type: list
The type(s) of GSE metric messages consumed by the GSE plugin.

aggregator_flag
: Type: boolean This is always false for the Global Cluster plugin.

member_field
: Type: unicode
The field in the input message used by the GSE plugin to identify the cluster members.

output_message_type
: Type: unicode
The type of metric messages emitted by the GSE plugin.

output_metric_name
: Type: unicode
The Fields[name] value of the metric messages that the GSE plugin emits.

interval
: Type: integer
The interval (in seconds) at which the GSE plugin emits its metric messages.

warm_up_period
: Type: integer
The number of seconds after a (re)start that the GSE plugin will wait before emitting its metric messages.

clusters
: Type: list
The list of node clusters and service clusters that the plugin handles. See *Top-level cluster definition* for details.

Top-level cluster definition

The following example shows the top-level clusters definition:

```
gse_cluster_global:
  [...]
  clusters:
    nova:
      policy: highest_severity
      group_by: member
      members:
        - nova-logs
        - nova-api
```

```
        - nova-metadata-api
        - nova-scheduler
        - nova-compute
        - nova-conductor
        - nova-cert
        - nova-consoleauth
        - nova-novncproxy-websocket
        - controller
        - compute
      hints:
        - cinder
        - glance
        - keystone
        - neutron
        - mysql
        - rabbitmq
```

Where

members
: Type: list
The list of cluster members.
The GSE messages are associated to the cluster when the `member_field` value (`cluster_name`), is on this list.

hints
: Type: list
The list of clusters that are indirectly associated with the top-level cluster. The GSE messages are indirectly associated to the cluster when the `member_field` value (`cluster_name`) is on this list. This means that they are not used to derive the health status of the top-level cluster but as 'hints' for root cause analysis.

group_by
: Type: Enum(member, hostname)
This parameter defines how the incoming GSE metrics are aggregated.
In the case of the top-level cluster definition, it is always by member.

policy:
: Type: unicode
The policy to use for computing the top-level cluster status. See *Health status policies* for details.

Apply your configuration changes

Once you have edited and saved your changes in `/etc/hiera/override/alarming.yaml` and / or `/etc/hiera/override/gse_filters.yaml`, apply the following Puppet manifest on all the nodes of your Open-

Stack environment **including the node(s) where Nagios is installed** for the changes to take effect:

```
# puppet apply --modulepath=/etc/fuel/plugins/lma_collector-<version>/puppet/modules:\
    /etc/puppet/modules \
    /etc/fuel/plugins/lma_collector-<version>/puppet/manifests/configure_afd_filters.
↪pp
```

If you have also deployed *lma_infrastructure_alerting" plugin, Nagios must be reconfigured as well by running the following commands on all the nodes with the *lma_infrastructure_alerting* role:

```
# rm -f /etc/nagios3/conf.d/lma_* && puppet apply \
     --modulepath=/etc/fuel/plugins/lma_infrastructure_alerting-<version>/puppet/
↪modules:\
    /etc/puppet/modules \
    /etc/fuel/plugins/lma_infrastructure_alerting-<version>/puppet/manifests/nagios.pp
```

CHAPTER 5

Appendix

List of metrics

The following is a list of metrics that are emitted by the StackLight Collector. The metrics are listed by category, then by metric name.

System

CPU

Metrics have a `cpu_number` field that contains the CPU number to which the metric applies.

- `cpu_idle`, the percentage of CPU time spent in the idle task.
- `cpu_interrupt`, the percentage of CPU time spent servicing interrupts.
- `cpu_nice`, the percentage of CPU time spent in user mode with low priority (nice).
- `cpu_softirq`, the percentage of CPU time spent servicing soft interrupts.
- `cpu_steal`, the percentage of CPU time spent in other operating systems.
- `cpu_system`, the percentage of CPU time spent in system mode.
- `cpu_user`, the percentage of CPU time spent in user mode.
- `cpu_wait`, the percentage of CPU time spent waiting for I/O operations to complete.

Disk

Metrics have a `device` field that contains the disk device number the metric applies to. For example, 'sda', 'sdb', and others.

- `disk_merged_read`, the number of read operations per second that could be merged with already queued operations.

- `disk_merged_write`, the number of write operations per second that could be merged with already queued operations.
- `disk_octets_read`, the number of octets (bytes) read per second.
- `disk_octets_write`, the number of octets (bytes) written per second.
- `disk_ops_read`, the number of read operations per second.
- `disk_ops_write`, the number of write operations per second.
- `disk_time_read`, the average time for a read operation to complete in the last interval.
- `disk_time_write`, the average time for a write operation to complete in the last interval.

File system

Metrics have a `fs` field that contains the partition's mount point to which the metric applies. For example, '/', '/var/lib', and others.

- `fs_inodes_free`, the number of free inodes on the file system.
- `fs_inodes_percent_free`, the percentage of free inodes on the file system.
- `fs_inodes_percent_reserved`, the percentage of reserved inodes.
- `fs_inodes_percent_used`, the percentage of used inodes.
- `fs_inodes_reserved`, the number of reserved inodes.
- `fs_inodes_used`, the number of used inodes.
- `fs_space_free`, the number of free bytes.
- `fs_space_percent_free`, the percentage of free bytes.
- `fs_space_percent_reserved`, the percentage of reserved bytes.
- `fs_space_percent_used`, the percentage of used bytes.
- `fs_space_reserved`, the number of reserved bytes.
- `fs_space_used`, the number of used bytes.

System load

- `load_longterm`, the system load average over the last 15 minutes.
- `load_midterm`, the system load average over the last 5 minutes.
- `load_shortterm`, the system load average over the last minute.

Memory

- `memory_buffered`, the amount of buffered memory in bytes.
- `memory_cached`, the amount of cached memory in bytes.
- `memory_free`, the amount of free memory in bytes.
- `memory_used`, the amount of used memory in bytes.

Network

Metrics have an `interface` field that contains the interface name the metric applies to. For example, 'eth0', 'eth1', and others.

- `if_collisions`, the number of collisions per second per interface.
- `if_dropped_rx`, the number of dropped packets per second when receiving from the interface.
- `if_dropped_tx`, the number of dropped packets per second when transmitting from the interface.
- `if_errors_rx`, the number of errors per second detected when receiving from the interface.
- `if_errors_rx_crc`, the number of received frames with wrong CRC (cyclic redundancy check) per second.
- `if_errors_rx_fifo`, the number of received frames dropped per second due to FIFO buffer overflows.
- `if_errors_rx_frame`, the number of received frames with invalid frame checksum (FCS).
- `if_errors_rx_length`, the number of received frames with a length that doesn't comply with the Ethernet specification.
- `if_errors_rx_missed`, the number of missed packets when receiving from the interface.
- `if_errors_rx_over`, the number of received frames per second that were dropped due to an hardware port receive buffer overflow.
- `if_errors_tx`, the number of errors per second detected when transmitting from the interface.
- `if_errors_tx_aborted`, the number of aborted frames per second when transmitting from the interface
- `if_errors_tx_carrier`, the number of times per second the interface has lost its link connection to the switch.
- `if_errors_tx_fifo`, the number of transmitted frames per second dropped due to FIFO buffer overflows.
- `if_errors_tx_heartbeat`, the number of heartbeat errors per second.
- `if_errors_tx_window`, the number of late collisions per second when transmitting from the interface.
- `if_multicast`, the number of multicast packets per second per interface.
- `if_octets_rx`, the number of octets (bytes) received per second by the interface.
- `if_octets_tx`, the number of octets (bytes) transmitted per second by the interface.
- `if_packets_rx`, the number of packets received per second by the interface.
- `if_packets_tx`, the number of packets transmitted per second by the interface.

Processes

- `processes_count`, the number of processes in a given state. The metric has a `state` field (one of 'blocked', 'paging', 'running', 'sleeping', 'stopped' or 'zombies').
- `processes_fork_rate`, the number of processes forked per second.

Swap

- `swap_cached`, the amount of cached memory (in bytes) that is in the swap.
- `swap_free`, the amount of free memory (in bytes) that is in the swap.
- `swap_io_in`, the number of swap bytes written per second.

- swap_io_out, the number of swap bytes read per second.
- swap_used, the amount of used memory (in bytes) that is in the swap.
- swap_percent_used, the amount of used memory (in percentages) that is in the swap.

Users

- logged_users, the number of users currently logged in.

Apache

- apache_bytes, the number of bytes per second transmitted by the server.
- apache_connections, the current number of active connections.
- apache_idle_workers, the current number of idle workers.
- apache_requests, the number of requests processed per second.
- apache_workers_closing, the number of workers in closing state.
- apache_workers_dnslookup, the number of workers in DNS lookup state.
- apache_workers_finishing, the number of workers in finishing state.
- apache_workers_idle_cleanup, the number of workers in idle cleanup state.
- apache_workers_keepalive, the number of workers in keepalive state.
- apache_workers_logging, the number of workers in logging state.
- apache_workers_open, the number of workers in open state.
- apache_workers_reading, the number of workers in reading state.
- apache_workers_sending, the number of workers in sending state.
- apache_workers_starting, the number of workers in starting state.
- apache_workers_waiting, the number of workers in waiting state.

MySQL

Commands

mysql_commands, the number of times per second a given statement has been executed. The metric has a statement field that contains the statement to which it applies. The values can be as follows:

- change_db for the USE statement.
- commit for the COMMIT statement.
- flush for the FLUSH statement.
- insert for the INSERT statement.
- rollback for the ROLLBACK statement.
- select for the SELECT statement.
- set_option for the SET statement.

- `show_collations` for the SHOW COLLATION statement.
- `show_databases` for the SHOW DATABASES statement.
- `show_fields` for the SHOW FIELDS statement.
- `show_master_status` for the SHOW MASTER STATUS statement.
- `show_status` for the SHOW STATUS statement.
- `show_tables` for the SHOW TABLES statement.
- `show_variables` for the SHOW VARIABLES statement.
- `show_warnings` for the SHOW WARNINGS statement.
- `update` for the UPDATE statement.

Handlers

`mysql_handler`, the number of times per second a given handler has been executed. The metric has a `handler` field that contains the handler it applies to. The values can be as follows:

- `commit` for the internal COMMIT statements.
- `delete` for the internal DELETE statements.
- `external_lock` for the external locks.
- `read_first` for the requests that read the first entry in an index.
- `read_key` for the requests that read a row based on a key.
- `read_next` for the requests that read the next row in key order.
- `read_prev` for the requests that read the previous row in key order.
- `read_rnd` for the requests that read a row based on a fixed position.
- `read_rnd_next` for the requests that read the next row in the data file.
- `rollback` the requests that perform the rollback operation.
- `update` the requests that update a row in a table.
- `write` the requests that insert a row in a table.

Locks

- `mysql_locks_immediate`, the number of times per second the requests for table locks could be granted immediately.
- `mysql_locks_waited`, the number of times per second the requests for table locks had to wait.

Network

- `mysql_octets_rx`, the number of bytes per second received by the server.
- `mysql_octets_tx`, the number of bytes per second sent by the server.

Threads

- `mysql_threads_cached`, the number of threads in the thread cache.
- `mysql_threads_connected`, the number of currently open connections.
- `mysql_threads_created`, the number of threads created per second to handle connections.
- `mysql_threads_running`, the number of threads that are not sleeping.

Cluster

The following metrics are collected with statement 'SHOW STATUS'. For details, see Percona documentation.

- `mysql_cluster_connected`, `1` when the node is connected to the cluster, if not, then `0`.
- `mysql_cluster_local_cert_failures`, the number of write sets that failed the certification test.
- `mysql_cluster_local_commits`, the number of write sets committed on the node.
- `mysql_cluster_local_recv_queue`, the number of write sets waiting to be applied.
- `mysql_cluster_local_send_queue`, the number of write sets waiting to be sent.
- `mysql_cluster_ready`, `1` when the node is ready to accept queries, if not, then `0`.
- `mysql_cluster_received`, the total number of write sets received from other nodes.
- `mysql_cluster_received_bytes`, the total size in bytes of write sets received from other nodes.
- `mysql_cluster_replicated`, the total number of write sets sent to other nodes.
- `mysql_cluster_replicated_bytes` the total size in bytes of write sets sent to other nodes.
- `mysql_cluster_size`, the current number of nodes in the cluster.
- `mysql_cluster_status`, `1` when the node is 'Primary', `2` if 'Non-Primary', and `3` if 'Disconnected'.

Slow queries

The following metric is collected with statement 'SHOW STATUS where Variable_name = 'Slow_queries'.

- `mysql_slow_queries`, the number of queries that have taken more than X seconds, depending on the MySQL configuration parameter 'long_query_time' (10s per default).

RabbitMQ

Cluster

- `rabbitmq_connections`, the total number of connections.
- `rabbitmq_consumers`, the total number of consumers.
- `rabbitmq_channels`, the total number of channels.
- `rabbitmq_exchanges`, the total number of exchanges.
- `rabbitmq_messages`, the total number of messages which are ready to be consumed or not yet acknowledged.
- `rabbitmq_queues`, the total number of queues.

- `rabbitmq_running_nodes`, the total number of running nodes in the cluster.
- `rabbitmq_disk_free`, the free disk space.
- `rabbitmq_disk_free_limit`, the minimum amount of free disk space for RabbitMQ. When `rabbitmq_disk_free` drops below this value, all producers are blocked.
- `rabbitmq_remaining_disk`, the difference between `rabbitmq_disk_free` and `rabbitmq_disk_free_limit`.
- `rabbitmq_used_memory`, bytes of memory used by the whole RabbitMQ process.
- `rabbitmq_vm_memory_limit`, the maximum amount of memory allocated for RabbitMQ. When `rabbitmq_used_memory` uses more than this value, all producers are blocked.
- `rabbitmq_remaining_memory`, the difference between `rabbitmq_vm_memory_limit` and `rabbitmq_used_memory`.

HAProxy

The `frontend` and `backend` field values can be as follows:

- cinder-api
- glance-api
- glance-registry-api
- heat-api
- heat-cfn-api
- heat-cloudwatch-api
- horizon-web (when Horizon is deployed without TLS)
- horizon-https (when Horizon is deployed with TLS)
- keystone-public-api
- keystone-admin-api
- mysqld-tcp
- murano-api
- neutron-api
- nova-api
- nova-metadata-api
- nova-novncproxy-websocket
- sahara-api
- swift-api

Server

- `haproxy_connections`, the number of current connections.
- `haproxy_pipes_free`, the number of free pipes.
- `haproxy_pipes_used`, the number of used pipes.

- haproxy_run_queue, the number of connections waiting in the queue.
- haproxy_ssl_connections, the number of current SSL connections.
- haproxy_tasks, the number of tasks.
- haproxy_uptime, the HAProxy server uptime in seconds.

Frontends

The following metrics have a frontend field that contains the name of the front-end server:

- haproxy_frontend_bytes_in, the number of bytes received by the frontend.
- haproxy_frontend_bytes_out, the number of bytes transmitted by the frontend.
- haproxy_frontend_denied_requests, the number of denied requests.
- haproxy_frontend_denied_responses, the number of denied responses.
- haproxy_frontend_error_requests, the number of error requests.
- haproxy_frontend_response_1xx, the number of HTTP responses with 1xx code.
- haproxy_frontend_response_2xx, the number of HTTP responses with 2xx code.
- haproxy_frontend_response_3xx, the number of HTTP responses with 3xx code.
- haproxy_frontend_response_4xx, the number of HTTP responses with 4xx code.
- haproxy_frontend_response_5xx, the number of HTTP responses with 5xx code.
- haproxy_frontend_response_other, the number of HTTP responses with other code.
- haproxy_frontend_session_current, the number of current sessions.
- haproxy_frontend_session_total, the cumulative number of sessions.

Backends

The following metrics have a backend field that contains the name of the back-end server:

- haproxy_backend_bytes_in, the number of bytes received by the back end.
- haproxy_backend_bytes_out, the number of bytes transmitted by the back end.
- haproxy_backend_denied_requests, the number of denied requests.
- haproxy_backend_denied_responses, the number of denied responses.
- haproxy_backend_downtime, the total downtime in seconds.
- haproxy_backend_error_connection, the number of error connections.
- haproxy_backend_error_responses, the number of error responses.
- haproxy_backend_queue_current, the number of requests in queue.
- haproxy_backend_redistributed, the number of times a request was redispatched to another server.
- haproxy_backend_response_1xx, the number of HTTP responses with 1xx code.
- haproxy_backend_response_2xx, the number of HTTP responses with 2xx code.
- haproxy_backend_response_3xx, the number of HTTP responses with 3xx code.
- haproxy_backend_response_4xx, the number of HTTP responses with 4xx code.

- haproxy_backend_response_5xx, the number of HTTP responses with 5xx code.
- haproxy_backend_response_other, the number of HTTP responses with other code.
- haproxy_backend_retries, the number of times a connection to a server was retried.
- haproxy_backend_server, the status of the backend server where values 0 and 1 represent, respectively, DOWN and UP. This metric has two additional fields: a state field that contains the state of the backend (either 'down' or 'up') and a server field that contains the hostname of the backend server.
- haproxy_backend_servers, the count of servers grouped by state. This metric has an additional state field that contains the state of the back ends (either 'down' or 'up').
- haproxy_backend_servers_percent, the percentage of servers by state. This metric has an additional state field that contains the state of the back ends (either 'down' or 'up').
- haproxy_backend_session_current, the number of current sessions.
- haproxy_backend_session_total, the cumulative number of sessions.
- haproxy_backend_status, the global back-end status where values 0 and 1 represent, respectively, DOWN (all back ends are down) and UP (at least one back end is up).

Memcached

- memcached_command_flush, the cumulative number of flush reqs.
- memcached_command_get, the cumulative number of retrieval reqs.
- memcached_command_set, the cumulative number of storage reqs.
- memcached_command_touch, the cumulative number of touch reqs.
- memcached_connections_current, the number of open connections.
- memcached_df_cache_free, the current number of free bytes to store items.
- memcached_df_cache_used, the current number of bytes used to store items.
- memcached_items_current, the current number of items stored.
- memcached_octets_rx, the total number of bytes read by this server from the network.
- memcached_octets_tx, the total number of bytes sent by this server to the network.
- memcached_ops_decr_hits, the number of successful decr reqs.
- memcached_ops_decr_misses, the number of decr reqs against missing keys.
- memcached_ops_evictions, the number of valid items removed from cache to free memory for new items.
- memcached_ops_hits, the number of keys that have been requested.
- memcached_ops_incr_hits, the number of successful incr reqs.
- memcached_ops_incr_misses, the number of successful incr reqs.
- memcached_ops_misses, the number of items that have been requested and not found.
- memcached_percent_hitratio, the percentage of get command hits (in cache).
- memcached_ps_cputime_syst, the percentage of CPU time spent in system mode by memcached. It can be greater than 100% when the node has more than one CPU.
- memcached_ps_cputime_user, the percentage of CPU time spent in user mode by memcached. It can be greater than 100% when the node has more than one CPU.

For details, see the Memcached documentation.

Libvirt

Every metric contains an `instance_id` field, which is the UUID of the instance for the Nova service.

CPU

- `virt_cpu_time`, the average amount of CPU time (in nanoseconds) allocated to the virtual instance in a second.
- `virt_vcpu_time`, the average amount of CPU time (in nanoseconds) allocated to the virtual CPU in a second. The metric contains a `vcpu_number` field which is the virtual CPU number.

Disk

Metrics have a `device` field that contains the virtual disk device to which the metric applies. For example, 'vda', 'vdb', and others.

- `virt_disk_octets_read`, the number of octets (bytes) read per second.
- `virt_disk_octets_write`, the number of octets (bytes) written per second.
- `virt_disk_ops_read`, the number of read operations per second.
- `virt_disk_ops_write`, the number of write operations per second.

Memory

- `virt_memory_total`, the total amount of memory (in bytes) allocated to the virtual instance.

Network

Metrics have an `interface` field that contains the interface name to which the metric applies. For example, 'tap0dc043a6-dd', 'tap769b123a-2e', and others.

- `virt_if_dropped_rx`, the number of dropped packets per second when receiving from the interface.
- `virt_if_dropped_tx`, the number of dropped packets per second when transmitting from the interface.
- `virt_if_errors_rx`, the number of errors per second detected when receiving from the interface.
- `virt_if_errors_tx`, the number of errors per second detected when transmitting from the interface.
- `virt_if_octets_rx`, the number of octets (bytes) received per second by the interface.
- `virt_if_octets_tx`, the number of octets (bytes) transmitted per second by the interface.
- `virt_if_packets_rx`, the number of packets received per second by the interface.
- `virt_if_packets_tx`, the number of packets transmitted per second by the interface.

OpenStack

Service API checks

- **`openstack_check_api`, the service's API status through the load balancer** VIP, `1` if it is responsive, if not, then `0`. The metric contains a `service` field that identifies the OpenStack service being checked.
- `openstack_check_local_api`, the service's API status checked locally. `1` if it is responsive, if not, then `0`. The metric contains a `service` field that identifies `<service>` identifies the OpenStack service being checked.

`<service>` is one of the following values with their respective resource checks:

- 'ceilometer-api': '/v2/capabilities'
- 'cinder-api': '/'
- 'cinder-v2-api': '/'
- 'glance-api': '/'
- 'heat-api': '/'
- 'heat-cfn-api': '/'
- 'keystone-public-api': '/'
- 'neutron-api': '/'
- 'nova-api': '/'
- 'swift-api': '/healthcheck'
- 'swift-s3-api': '/healthcheck'

Note: All checks except for Ceilometer are performed without authentication.

Compute

The following metrics are emitted per compute node:

- `openstack_nova_free_disk`, the disk space in GB available for new instances.
- `openstack_nova_free_ram`, the memory in MB available for new instances.
- `openstack_nova_free_vcpus`, the number of virtual CPU available for new instances.
- `openstack_nova_instance_creation_time`, the time in seconds it took to launch a new instance.
- `openstack_nova_running_instances`, the number of running instances.
- `openstack_nova_running_tasks`, the number of tasks currently executed.
- `openstack_nova_used_disk`, the disk space in GB used by the instances.
- `openstack_nova_used_ram`, the memory in MB used by the instances.
- `openstack_nova_used_vcpus`, the number of virtual CPU used by the instances.

If Nova aggregates are defined then the following metrics are emitted per aggregate. These metrics contain a `aggregate` field containing the aggregate name and a `aggregate_id` field containing the ID (integer) of the aggregate.

- `openstack_nova_aggregate_free_disk`, the total amount of disk space in GB available in given aggregate for new instances.
- `openstack_nova_aggregate_free_ram`, the total amount of memory in MB available in given aggregate for new instances.
- `openstack_nova_aggregate_free_vcpus`, the total number of virtual CPU available in given aggregate for new instances.
- `openstack_nova_aggregate_running_instances`, the total number of running instances in given aggregate.
- `openstack_nova_aggregate_running_tasks`, the total number of tasks currently executed in given aggregate.
- `openstack_nova_aggregate_used_disk`, the total amount of disk space in GB used by the instances in given aggregate.
- `openstack_nova_aggregate_used_ram`, the total amount of memory in MB used by the instances in given aggregate.
- `openstack_nova_aggregate_used_vcpus`, the total number of virtual CPU used by the instances in given aggregate.

The following metrics are retrieved from the Nova API and represent the aggregated values across all compute nodes.

- `openstack_nova_total_free_disk`, the total amount of disk space in GB available for new instances.
- `openstack_nova_total_free_ram`, the total amount of memory in MB available for new instances.
- `openstack_nova_total_free_vcpus`, the total number of virtual CPU available for new instances.
- `openstack_nova_total_running_instances`, the total number of running instances.
- `openstack_nova_total_running_tasks`, the total number of tasks currently executed.
- `openstack_nova_total_used_disk`, the total amount of disk space in GB used by the instances.
- `openstack_nova_total_used_ram`, the total amount of memory in MB used by the instances.
- `openstack_nova_total_used_vcpus`, the total number of virtual CPU used by the instances.

The following metrics are retrieved from the Nova API:

- `openstack_nova_instances`, the total count of instances in a given state. The metric contains a `state` field which is one of 'active', 'deleted', 'error', 'paused', 'resumed', 'rescued', 'resized', 'shelved_offloaded' or 'suspended'.
- `openstack_nova_service`, the Nova service state (either `0` for 'up', `1` for 'down' or `2` for 'disabled'). The metric contains a `service` field (one of 'compute', 'conductor', 'scheduler', 'cert' or 'consoleauth') and a `state` field (one of 'up', 'down' or 'disabled').
- `openstack_nova_services`, the total count of Nova services by state. The metric contains a `service` field (one of 'compute', 'conductor', 'scheduler', 'cert' or 'consoleauth') and a `state` field (one of 'up', 'down', or 'disabled').
- `openstack_nova_services_percent`, the percentage of Nova services by state. The metric contains a `service` field (one of 'compute', 'conductor', 'scheduler', 'cert' or 'consoleauth') and a `state` field (one of 'up', 'down', or 'disabled').

Identity

The following metrics are retrieved from the Keystone API:

- `openstack_keystone_roles`, the total number of roles.
- `openstack_keystone_tenants`, the number of tenants by state. The metric contains a `state` field (either 'enabled' or 'disabled').
- `openstack_keystone_users`, the number of users by state. The metric contains a `state` field (either 'enabled' or 'disabled').

Volume

The following metrics are emitted per volume node:

- `openstack_cinder_volume_attachement_time`, the time in seconds it took to attach a volume to an instance.
- `openstack_cinder_volume_creation_time`, the time in seconds it took to create a new volume.

Note: When using Ceph as the back end storage for volumes, the `hostname` value is always set to `rbd`.

The following metrics are retrieved from the Cinder API:

- `openstack_cinder_snapshots`, the number of snapshots by state. The metric contains a `state` field.
- `openstack_cinder_snapshots_size`, the total size (in bytes) of snapshots by state. The metric contains a `state` field.
- `openstack_cinder_volumes`, the number of volumes by state. The metric contains a `state` field.
- `openstack_cinder_volumes_size`, the total size (in bytes) of volumes by state. The metric contains a `state` field.

`state` is one of 'available', 'creating', 'attaching', 'in-use', 'deleting', 'backing-up', 'restoring-backup', 'error', 'error_deleting', 'error_restoring', 'error_extending'.

- `openstack_cinder_service`, the Cinder service state (either `0` for 'up', `1` for 'down', or `2` for 'disabled'). The metric contains a `service` field (one of 'volume', 'backup', 'scheduler') and a `state` field (one of 'up', 'down' or 'disabled').
- `openstack_cinder_services`, the total count of Cinder services by state. The metric contains a `service` field (one of 'volume', 'backup', 'scheduler') and a `state` field (one of 'up', 'down' or 'disabled').
- `openstack_cinder_services_percent`, the percentage of Cinder services by state. The metric contains a `service` field (one of 'volume', 'backup', 'scheduler') and a `state` field (one of 'up', 'down', or 'disabled').

Image

The following metrics are retrieved from the Glance API:

- `openstack_glance_images`, the number of images by state and visibility. The metric contains `state` and `visibility` fields.
- `openstack_glance_images_size`, the total size (in bytes) of images by state and visibility. The metric contains `state` and `visibility` fields.
- `openstack_glance_snapshots`, the number of snapshot images by state and visibility. The metric contains `state` and `visibility` fields.
- `openstack_glance_snapshots_size`, the total size (in bytes) of snapshots by state and visibility. The metric contains `state` and `visibility` fields.

`state` is one of 'queued', 'saving', 'active', 'killed', 'deleted', 'pending_delete'. `visibility` is either 'public' or 'private'.

Network

The following metrics are retrieved from the Neutron API:

- `openstack_neutron_floatingips`, the total number of floating IP addresses.
- `openstack_neutron_networks`, the number of virtual networks by state. The metric contains a `state` field.
- `openstack_neutron_ports`, the number of virtual ports by owner and state. The metric contains `owner` and `state` fields.
- `openstack_neutron_routers`, the number of virtual routers by state. The metric contains a `state` field.
- `openstack_neutron_subnets`, the number of virtual subnets.

`<state>` is one of 'active', 'build', 'down' or 'error'.

`<owner>` is one of 'compute', 'dhcp', 'floatingip', 'floatingip_agent_gateway', 'router_interface', 'router_gateway', 'router_ha_interface', 'router_interface_distributed', or 'router_centralized_snat'.

Note: These metrics are not collected when the Contrail plugin is deployed.

- `openstack_neutron_agent`, the Neutron agent state (either `0` for 'up', `1` for 'down', or `2` for 'disabled'). The metric contains a `service` field (one of 'dhcp', 'l3', 'metadata', or 'openvswitch'), and a `state` field (one of 'up', 'down' or 'disabled').
- `openstack_neutron_agents`, the total number of Neutron agents by service and state. The metric contains `service` (one of 'dhcp', 'l3', 'metadata' or 'openvswitch') and `state` (one of 'up', 'down' or 'disabled') fields.
- `openstack_neutron_agents_percent`, the percentage of Neutron agents by state. The metric contains a `service` field (one of 'dhcp', 'l3', 'metadata' or 'openvswitch') and a `state` field (one of 'up', 'down', or 'disabled').

API response times

- `openstack_<service>_http_response_times`, HTTP response time statistics. The statistics are `min`, `max`, `sum`, `count`, `upper_90` (90 percentile) over 10 seconds. The metric contains an `http_method` field, for example, 'GET', 'POST', and others, and an `http_status` field, for example, '2xx', '4xx', and others.

`<service>` is one of 'cinder', 'glance', 'heat' 'keystone', 'neutron' or 'nova'.

Logs

- `log_messages`, the number of log messages per second for the given service and severity level. The metric contains `service` and `level` (one of 'debug', 'info', and others) fields.

Ceph

All Ceph metrics have a `cluster` field containing the name of the Ceph cluster (*ceph* by default).

For details, see Cluster monitoring and RADOS monitoring.

Cluster

- `ceph_health`, the health status of the entire cluster where values `1`, `2`, `3` represent `OK`, `WARNING` and `ERROR`, respectively.
- `ceph_monitor_count`, the number of ceph-mon processes.
- `ceph_quorum_count`, the number of ceph-mon processes participating in the quorum.

Pools

- `ceph_pool_total_avail_bytes`, the total available size in bytes for all pools.
- `ceph_pool_total_bytes`, the total number of bytes for all pools.
- `ceph_pool_total_number`, the total number of pools.
- `ceph_pool_total_used_bytes`, the total used size in bytes by all pools.

The following metrics have a `pool` field that contains the name of the Ceph pool.

- `ceph_pool_bytes_used`, the amount of data in bytes used by the pool.
- `ceph_pool_max_avail`, the available size in bytes for the pool.
- `ceph_pool_objects`, the number of objects in the pool.
- `ceph_pool_op_per_sec`, the number of operations per second for the pool.
- `ceph_pool_pg_num`, the number of placement groups for the pool.
- `ceph_pool_read_bytes_sec`, the number of bytes read by second for the pool.
- `ceph_pool_size`, the number of data replications for the pool.
- `ceph_pool_write_bytes_sec`, the number of bytes written by second for the pool.

Placement Groups

- `ceph_pg_bytes_avail`, the available size in bytes.
- `ceph_pg_bytes_total`, the cluster total size in bytes.
- `ceph_pg_bytes_used`, the data stored size in bytes.
- `ceph_pg_data_bytes`, the stored data size in bytes before it is replicated, cloned or snapshotted.
- `ceph_pg_state`, the number of placement groups in a given state. The metric contains a `state` field whose `<state>` value is a combination separated by `+` of 2 or more states of this list: `creating`, `active`, `clean`, `down`, `replay`, `splitting`, `scrubbing`, `degraded`, `inconsistent`, `peering`, `repair`, `recovering`, `recovery_wait`, `backfill`, `backfill-wait`, `backfill_toofull`, `incomplete`, `stale`, `remapped`.
- `ceph_pg_total`, the total number of placement groups.

OSD Daemons

- ceph_osd_down, the number of OSD daemons DOWN.
- ceph_osd_in, the number of OSD daemons IN.
- ceph_osd_out, the number of OSD daemons OUT.
- ceph_osd_up, the number of OSD daemons UP.

The following metrics have an osd field that contains the OSD identifier:

- ceph_osd_apply_latency, apply latency in ms for the given OSD.
- ceph_osd_commit_latency, commit latency in ms for the given OSD.
- ceph_osd_total, the total size in bytes for the given OSD.
- ceph_osd_used, the data stored size in bytes for the given OSD.

OSD Performance

All the following metrics are retrieved per OSD daemon from the corresponding /var/run/ceph/ceph-osd.<ID>.asok socket by issuing the **perf dump** command.

All metrics have an osd field that contains the OSD identifier.

Note: These metrics are not collected when a node has both the ceph-osd and controller roles.

For details, see OSD performance counters.

- ceph_perf_osd_op, the number of client operations.
- ceph_perf_osd_op_in_bytes, the number of bytes received from clients for write operations.
- ceph_perf_osd_op_latency, the average latency in ms for client operations (including queue time).
- ceph_perf_osd_op_out_bytes, the number of bytes sent to clients for read operations.
- ceph_perf_osd_op_process_latency, the average latency in ms for client operations (excluding queue time).
- ceph_perf_osd_op_r, the number of client read operations.
- ceph_perf_osd_op_r_latency, the average latency in ms for read operation (including queue time).
- ceph_perf_osd_op_r_out_bytes, the number of bytes sent to clients for read operations.
- ceph_perf_osd_op_r_process_latency, the average latency in ms for read operation (excluding queue time).
- ceph_perf_osd_op_rw, the number of client read-modify-write operations.
- ceph_perf_osd_op_rw_in_bytes, the number of bytes per second received from clients for read-modify-write operations.
- ceph_perf_osd_op_rw_latency, the average latency in ms for read-modify-write operations (including queue time).
- ceph_perf_osd_op_rw_out_bytes, the number of bytes per second sent to clients for read-modify-write operations.
- ceph_perf_osd_op_rw_process_latency, the average latency in ms for read-modify-write operations (excluding queue time).

- ceph_perf_osd_op_rw_rlat, the average latency in ms for read-modify-write operations with readable/applied.
- ceph_perf_osd_op_w, the number of client write operations.
- ceph_perf_osd_op_wip, the number of replication operations currently being processed (primary).
- ceph_perf_osd_op_w_in_bytes, the number of bytes received from clients for write operations.
- ceph_perf_osd_op_w_latency, the average latency in ms for write operations (including queue time).
- ceph_perf_osd_op_w_process_latency, the average latency in ms for write operation (excluding queue time).
- ceph_perf_osd_op_w_rlat, the average latency in ms for write operations with readable/applied.
- ceph_perf_osd_recovery_ops, the number of recovery operations in progress.

Pacemaker

Cluster

- pacemaker_local_dc_active, 1 when the Designated Controller (DC) is the local host, if not, then 0.
- pacemaker_dc[1], 1 when the Designated Controller (DC) is present, if not, then 0.
- pacemaker_quorum_status[1], 1 when the cluster's quorum is reached, if not, then 0.
- pacemaker_configured_nodes[1], the number of configured nodes in the cluster.
- pacemaker_configured_resources[1], the number of configured nodes in the cluster.

Node

The following metrics are only emitted from the node that is the Designated Controller (DC) of the Pacemaker cluster. They have a status field which is one of 'offline', 'maintenance', or 'online':

- pacemaker_node_status, the status of the node, 0 when offline, 1 when in maintenance or 2 when online.
- pacemaker_node_count, the total number of nodes with the given status.
- pacemaker_node_percent, the percentage of nodes with the given status.

Resource

- pacemaker_local_resource_active, 1 when the resource is located on the host reporting the metric, if not, then 0. The metric contains a resource field which is one of 'vip__public', 'vip__management', 'vip__vrouter_pub', or 'vip__vrouter'.
- pacemaker_resource_failures[2], the total number of failures that Pacemaker detected for the resource. The counter is reset every time the collector restarts. The metric contains a resource field which one of 'vip__management', 'vip__public', 'vip__vrouter_pub', 'vip__vrouter', 'rabbitmq', 'mysqld' or 'haproxy'.

[1] this metric is only emitted from the node that is the Designated Controller (DC) of the Pacemaker cluster.
[2] this metric is only emitted from the node that is the Designated Controller (DC) of the Pacemaker cluster.

- `pacemaker_resource_operations`[2], the total number of operations that Pacemaker applied to the `resource`. The counter is reset every time the collector restarts. The metric contains a `resource` field which one of 'vip__management', 'vip__public', 'vip__vrouter_pub', 'vip__vrouter', 'rabbitmq', 'mysqld' or 'haproxy'.

The following metrics have `resource` and `status` fields.

`status` is one of 'offline', 'maintenance', or 'online'.

`resource` is one of 'vip__management', 'vip__public', 'vip__vrouter_pub', 'vip__vrouter', 'rabbitmq', 'mysqld' or 'haproxy'.

- `pacemaker_resource_count`[2], the total number of instances for the given `status` and `resource`.
- `pacemaker_resource_percent`[2], the percentage of instances for the given `status` and `resource`.

Clusters

The cluster metrics are emitted by the GSE plugins. For details, see *Configuring alarms*.

- `cluster_node_status`, the status of the node cluster. The metric contains a `cluster_name` field that identifies the node cluster.
- `cluster_service_status`, the status of the service cluster. The metric contains a `cluster_name` field that identifies the service cluster.
- `cluster_status`, the status of the global cluster. The metric contains a `cluster_name` field that identifies the global cluster.

The supported values for these metrics are:

- `0` for the *Okay* status.
- `1` for the *Warning* status.
- `2` for the *Unknown* status.
- `3` for the *Critical* status.
- `4` for the *Down* status.

Self-monitoring

System

The metrics have a `service` field with the name of the service it applies to. The values can be: `hekad`, `collectd`, `influxd`, `grafana-server` or `elasticsearch`.

- `lma_components_count_processes`, the number of processes currently running.
- `lma_components_count_threads`, the number of threads currently running.
- `lma_components_cputime_syst`, the percentage of CPU time spent in system mode by the service. It can be greater than 100% when the node has more than one CPU.
- `lma_components_cputime_user`, the percentage of CPU time spent in user mode by the service. It can be greater than 100% when the node has more than one CPU.
- `lma_components_disk_bytes_read`, the number of bytes read from disk(s) per second.
- `lma_components_disk_bytes_write`, the number of bytes written to disk(s) per second.

- `lma_components_disk_ops_read`, the number of read operations from disk(s) per second.
- `lma_components_disk_ops_write`, the number of write operations to disk(s) per second.
- `lma_components_memory_code`, the physical memory devoted to executable code in bytes.
- `lma_components_memory_data`, the physical memory devoted to other than executable code in bytes.
- `lma_components_memory_rss`, the non-swapped physical memory used in bytes.
- `lma_components_memory_vm`, the virtual memory size in bytes.
- `lma_components_pagefaults_majflt`, major page faults per second.
- `lma_components_pagefaults_minflt`, minor page faults per second.
- `lma_components_stacksize`, the absolute value of the start address (the bottom) of the stack minus the address of the current stack pointer.

Heka pipeline

The metrics have two fields: `name` that contains the name of the decoder or filter as defined by *Heka* and `type` that is either *decoder* or *filter*.

The metrics for both types are as follows:

- `hekad_memory`, the total memory in bytes used by the Sandbox.
- `hekad_msg_avg_duration`, the average time in nanoseconds for processing the message.
- `hekad_msg_count`, the total number of messages processed by the decoder. This resets to `0` when the process is restarted.

Additional metrics for *filter* type:

- `heakd_timer_event_avg_duration`, the average time in nanoseconds for executing the *timer_event* function.
- `hekad_timer_event_count`, the total number of executions of the *timer_event* function. This resets to `0` when the process is restarted.

Back-end checks

- `http_check`, the API status of the back end, `1` if it is responsive, if not, then `0`. The metric contains a `service` field that identifies the LMA back-end service being checked.

`<service>` is one of the following values, depending on which Fuel plugins are deployed in the environment:

- 'influxdb'

Elasticsearch

The following metrics represent the simple status on the health of the cluster. For details, see Cluster health.

- `elasticsearch_cluster_active_primary_shards`, the number of active primary shards.
- `elasticsearch_cluster_active_shards`, the number of active shards.
- `elasticsearch_cluster_health`, the health status of the entire cluster where values `1`, `2` , `3` represent `green`, `yellow` and `red`, respectively. The `red` status may also be reported when the Elasticsearch API returns an unexpected result, for example, a network failure.

- elasticsearch_cluster_initializing_shards, the number of initializing shards.
- elasticsearch_cluster_number_of_nodes, the number of nodes in the cluster.
- elasticsearch_cluster_number_of_pending_tasks, the number of pending tasks.
- elasticsearch_cluster_relocating_shards, the number of relocating shards.
- elasticsearch_cluster_unassigned_shards, the number of unassigned shards.

InfluxDB

The following metrics are extracted from the output of the **show stats** command. The values are reset to zero when InfluxDB is restarted.

cluster

The following metrics are only available if there is more than one node in the cluster:

- influxdb_cluster_write_shard_points_requests, the number of requests for writing a time series points to a shard.
- influxdb_cluster_write_shard_requests, the number of requests for writing to a shard.

httpd

- influxdb_httpd_failed_auths, the number of failed authentications.
- influxdb_httpd_ping_requests, the number of ping requests.
- influxdb_httpd_query_requests, the number of query requests received.
- influxdb_httpd_query_response_bytes, the number of bytes returned to the client.
- influxdb_httpd_requests, the number of requests received.
- influxdb_httpd_write_points_ok, the number of points successfully written.
- influxdb_httpd_write_request_bytes, the number of bytes received for write requests.
- influxdb_httpd_write_requests, the number of write requests received.

write

- influxdb_write_local_point_requests, the number of write points requests from the local data node.
- influxdb_write_ok, the number of successful writes of consistency level.
- influxdb_write_point_requests, the number of write points requests across all data nodes.
- influxdb_write_remote_point_requests, the number of write points requests to remote data nodes.
- influxdb_write_requests, the number of write requests across all data nodes.
- influxdb_write_sub_ok, the number of successful points sent to subscriptions.

runtime

- `influxdb_garbage_collections`, the number of garbage collections.
- `influxdb_go_routines`, the number of Golang routines.
- `influxdb_heap_idle`, the number of bytes in idle spans.
- `influxdb_heap_in_use`, the number of bytes in non-idle spans.
- `influxdb_heap_objects`, the total number of allocated objects.
- `influxdb_heap_released`, the number of bytes released to the operating system.
- `influxdb_heap_system`, the number of bytes obtained from the system.
- `influxdb_memory_alloc`, the number of bytes allocated and not yet freed.
- `influxdb_memory_frees`, the number of free operations.
- `influxdb_memory_lookups`, the number of pointer lookups.
- `influxdb_memory_mallocs`, the number of malloc operations.
- `influxdb_memory_system`, the number of bytes obtained from the system.
- `influxdb_memory_total_alloc`, the number of bytes allocated (even if freed).

Checks

The check metrics are emitted to express the success or the failure of the metric collections for the local services. The value is `1` when successful and `0` if it fails.

- `apache_check`, for Apache.
- `ceph_mon_check`, for Ceph monitor.
- `ceph_osd_check`, for Ceph OSD.
- `elasticsearch_check`, for Elasticsearch.
- `haproxy_check`, for HAProxy.
- `influxdb_check`, for InfluxDB.
- `libvirt_check`, for Libvirt.
- `mysql_check`, for MySQL.
- `pacemaker_check`, for Pacemaker.
- `rabbitmq_check`, for RabbitMQ.

List of built-in alarms

The following is a list of StackLight built-in alarms:

```
alarms:
  - name: 'cpu-critical-controller'
    description: 'The CPU usage is too high (controller node)'
    severity: 'critical'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_idle
          relational_operator: '<='
          threshold: 5
          window: 120
          periods: 0
          function: avg
        - metric: cpu_wait
          relational_operator: '>='
          threshold: 35
          window: 120
          periods: 0
          function: avg
  - name: 'cpu-warning-controller'
    description: 'The CPU usage is high (controller node)'
    severity: 'warning'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_idle
          relational_operator: '<='
          threshold: 15
          window: 120
          periods: 0
          function: avg
        - metric: cpu_wait
          relational_operator: '>='
          threshold: 25
          window: 120
          periods: 0
          function: avg
  - name: 'swap-usage-critical'
    description: 'There is no more swap free space'
    severity: 'critical'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: swap_free
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: max
  - name: 'swap-activity-warning'
    description: 'The swap activity is high'
```

```
    severity: 'warning'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: swap_io_in
          relational_operator: '>='
          threshold: 1048576 # 1 Mb/s
          window: 120
          periods: 0
          function: avg
        - metric: swap_io_out
          relational_operator: '>='
          threshold: 1048576 # 1 Mb/s
          window: 120
          periods: 0
          function: avg
  - name: 'swap-usage-warning'
    description: 'The swap free space is low'
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: swap_percent_used
          relational_operator: '>='
          threshold: 0.8
          window: 60
          periods: 0
          function: avg
  - name: 'cpu-critical-compute'
    description: 'The CPU usage is too high (compute node)'
    severity: 'critical'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_wait
          relational_operator: '>='
          threshold: 30
          window: 120
          periods: 0
          function: avg
  - name: 'cpu-warning-compute'
    description: 'The CPU usage is high (compute node)'
    severity: 'warning'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_wait
          relational_operator: '>='
          threshold: 20
          window: 120
          periods: 0
          function: avg
  - name: 'cpu-critical-rabbitmq'
    description: 'The CPU usage is too high (RabbitMQ node)'
    severity: 'critical'
```

```
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_idle
          relational_operator: '<='
          threshold: 5
          window: 120
          periods: 0
          function: avg
  - name: 'cpu-warning-rabbitmq'
    description: 'The CPU usage is high (RabbitMQ node)'
    severity: 'warning'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_idle
          relational_operator: '<='
          threshold: 15
          window: 120
          periods: 0
          function: avg
  - name: 'cpu-critical-mysql'
    description: 'The CPU usage is too high (MySQL node)'
    severity: 'critical'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_idle
          relational_operator: '<='
          threshold: 5
          window: 120
          periods: 0
          function: avg
  - name: 'cpu-warning-mysql'
    description: 'The CPU usage is high (MySQL node)'
    severity: 'warning'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_idle
          relational_operator: '<='
          threshold: 15
          window: 120
          periods: 0
          function: avg
  - name: 'cpu-critical-storage'
    description: 'The CPU usage is too high (storage node)'
    severity: 'critical'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_wait
          relational_operator: '>='
```

```
          threshold: 40
          window: 120
          periods: 0
          function: avg
        - metric: cpu_idle
          relational_operator: '<='
          threshold: 5
          window: 120
          periods: 0
          function: avg
  - name: 'cpu-warning-storage'
    description: 'The CPU usage is high (storage node)'
    severity: 'warning'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_wait
          relational_operator: '>='
          threshold: 30
          window: 120
          periods: 0
          function: avg
        - metric: cpu_idle
          relational_operator: '<='
          threshold: 15
          window: 120
          periods: 0
          function: avg
  - name: 'cpu-critical-default'
    description: 'The CPU usage is too high'
    severity: 'critical'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: cpu_wait
          relational_operator: '>='
          threshold: 35
          window: 120
          periods: 0
          function: avg
        - metric: cpu_idle
          relational_operator: '<='
          threshold: 5
          window: 120
          periods: 0
          function: avg
  - name: 'rabbitmq-disk-limit-critical'
    description: 'RabbitMQ has reached the free disk threshold. All producers are␣
↪blocked'
    severity: 'critical'
    # If the local RabbitMQ instance is down, it will be caught by the
    # rabbitmq-check alarm
    no_data_policy: 'okay'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
```

```
      rules:
        - metric: rabbitmq_remaining_disk
          relational_operator: '<='
          threshold: 0
          window: 20
          periods: 0
          function: min
  - name: 'rabbitmq-disk-limit-warning'
    description: 'RabbitMQ is getting close to the free disk threshold'
    severity: 'warning'
    # If the local RabbitMQ instance is down, it will be caught by the
    # rabbitmq-check alarm
    no_data_policy: 'okay'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: rabbitmq_remaining_disk
          relational_operator: '<='
          threshold: 104857600 # 100MB
          window: 20
          periods: 0
          function: min
  - name: 'rabbitmq-memory-limit-critical'
    description: 'RabbitMQ has reached the memory threshold. All producers are blocked
↪'
    severity: 'critical'
    # If the local RabbitMQ instance is down, it will be caught by the
    # rabbitmq-check alarm
    no_data_policy: 'okay'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: rabbitmq_remaining_memory
          relational_operator: '<='
          threshold: 0
          window: 20
          periods: 0
          function: min
  - name: 'rabbitmq-memory-limit-warning'
    description: 'RabbitMQ is getting close to the memory threshold'
    severity: 'warning'
    # If the local RabbitMQ instance is down, it will be caught by the
    # rabbitmq-check alarm
    no_data_policy: 'okay'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: rabbitmq_remaining_memory
          relational_operator: '<='
          threshold: 104857600 # 100MB
          window: 20
          periods: 0
          function: min
  - name: 'rabbitmq-queue-warning'
    description: 'The number of outstanding messages is too high'
```

```
    severity: 'warning'
    # If the local RabbitMQ instance is down, it will be caught by the
    # rabbitmq-check alarm
    no_data_policy: 'okay'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: rabbitmq_messages
          relational_operator: '>='
          threshold: 200
          window: 120
          periods: 0
          function: avg
  - name: 'rabbitmq-pacemaker-down'
    description: 'The RabbitMQ cluster is down'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      logical_operator: 'and'
      rules:
        - metric: pacemaker_resource_percent
          fields:
            resource: rabbitmq
            status: up
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'rabbitmq-pacemaker-critical'
    description: 'The RabbitMQ cluster is critical because less than half of the␣
↪nodes are up'
    severity: 'critical'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      logical_operator: 'and'
      rules:
        - metric: pacemaker_resource_percent
          fields:
            resource: rabbitmq
            status: up
          relational_operator: '<'
          threshold: 50
          window: 60
          periods: 0
          function: last
  - name: 'rabbitmq-pacemaker-warning'
    description: 'The RabbitMQ cluster is degraded because some RabbitMQ nodes are␣
↪missing'
    severity: 'warning'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      logical_operator: 'and'
      rules:
```

```
      - metric: pacemaker_resource_percent
        fields:
          resource: rabbitmq
          status: up
        relational_operator: '<'
        threshold: 100
        window: 60
        periods: 0
        function: last
- name: 'apache-warning'
  description: 'There is no Apache idle workers available'
  severity: 'warning'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: apache_idle_workers
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: min
- name: 'apache-check'
  description: 'Apache cannot be checked'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: apache_check
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'log-fs-warning'
  description: "The log filesystem's free space is low"
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: fs_space_percent_free
        fields:
          fs: '/var/log'
        relational_operator: '<'
        threshold: 10
        window: 60
        periods: 0
        function: min
- name: 'log-fs-critical'
  description: "The log filesystem's free space is too low"
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: fs_space_percent_free
        fields:
          fs: '/var/log'
        relational_operator: '<'
```

```
        threshold: 5
        window: 60
        periods: 0
        function: min
- name: 'root-fs-warning'
  description: "The root filesystem's free space is low"
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: fs_space_percent_free
        fields:
          fs: '/'
        relational_operator: '<'
        threshold: 10
        window: 60
        periods: 0
        function: min
- name: 'root-fs-critical'
  description: "The root filesystem's free space is too low"
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: fs_space_percent_free
        fields:
          fs: '/'
        relational_operator: '<'
        threshold: 5
        window: 60
        periods: 0
        function: min
- name: 'mysql-fs-warning'
  description: "The MySQL filesystem's free space is low"
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: fs_space_percent_free
        fields:
          fs: '/var/lib/mysql'
        relational_operator: '<'
        threshold: 10
        window: 60
        periods: 0
        function: min
- name: 'mysql-fs-critical'
  description: "The MySQL filesystem's free space is too low"
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: fs_space_percent_free
        fields:
          fs: '/var/lib/mysql'
        relational_operator: '<'
        threshold: 5
        window: 60
```

```
          periods: 0
          function: min
  - name: 'nova-fs-warning'
    description: "The filesystem's free space is low (compute node)"
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: fs_space_percent_free
          fields:
            fs: '/var/lib/nova'
          relational_operator: '<'
          threshold: 10
          window: 60
          periods: 0
          function: min
  - name: 'nova-fs-critical'
    description: "The filesystem's free space is too low (compute node)"
    severity: 'critical'
    enabled: 'true'
    trigger:
      rules:
        - metric: fs_space_percent_free
          fields:
            fs: '/var/lib/nova'
          relational_operator: '<'
          threshold: 5
          window: 60
          periods: 0
          function: min
  - name: 'other-fs-warning'
    description: "The filesystem's free space is low"
    severity: 'warning'
    enabled: 'true'
    no_data_policy: 'okay'
    trigger:
      rules:
        - metric: fs_space_percent_free
          fields:
            fs: '!= /var/lib/nova && != /var/log && != /var/lib/mysql && != / && !~␣
↪ceph%-%d+$'
          group_by: [fs]
          relational_operator: '<'
          threshold: 10
          window: 60
          periods: 0
          function: min
  - name: 'other-fs-critical'
    description: "The filesystem's free space is too low"
    severity: 'critical'
    enabled: 'true'
    no_data_policy: 'okay'
    trigger:
      rules:
        - metric: fs_space_percent_free
          fields:
            fs: '!= /var/lib/nova && != /var/log && != /var/lib/mysql && != / && !~␣
↪ceph%-%d+$'
```

```
        group_by: [fs]
        relational_operator: '<'
        threshold: 5
        window: 60
        periods: 0
        function: min
- name: 'osd-disk-critical'
  description: "The filesystem's free space is too low (OSD disk)"
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: fs_space_percent_free
        fields:
          # Real FS is /var/lib/ceph/osd/ceph-0 but Collectd substituted '/' by '-'
          fs: '=~ ceph/%d+$'
        group_by: [fs]
        relational_operator: '<'
        threshold: 5
        window: 60
        periods: 0
        function: min
- name: 'nova-api-http-errors'
  description: 'Too many 5xx HTTP errors have been detected on nova-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: haproxy_backend_response_5xx
        fields:
          backend: 'nova-api'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 1
        function: diff
- name: 'nova-logs-error'
  description: 'Too many errors have been detected in Nova logs'
  severity: 'warning'
  no_data_policy: 'okay'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: log_messages
        fields:
          service: 'nova'
          level: 'error'
        relational_operator: '>'
        threshold: 0.1
        window: 70
        periods: 0
        function: max
- name: 'heat-api-http-errors'
  description: 'Too many 5xx HTTP errors have been detected on heat-api'
  severity: 'warning'
  enabled: 'true'
```

```
    trigger:
      logical_operator: 'or'
      rules:
        - metric: haproxy_backend_response_5xx
          fields:
            backend: 'heat-api'
          relational_operator: '>'
          threshold: 0
          window: 60
          periods: 1
          function: diff
  - name: 'heat-logs-error'
    description: 'Too many errors have been detected in Heat logs'
    severity: 'warning'
    no_data_policy: 'okay'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: log_messages
          fields:
            service: 'heat'
            level: 'error'
          relational_operator: '>'
          threshold: 0.1
          window: 70
          periods: 0
          function: max
  - name: 'swift-api-http-errors'
    description: 'Too many 5xx HTTP errors have been detected on swift-api'
    severity: 'warning'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: haproxy_backend_response_5xx
          fields:
            backend: 'swift-api || object-storage'
          relational_operator: '>'
          threshold: 0
          window: 60
          periods: 1
          function: diff
  - name: 'swift-logs-error'
    description: 'Too many errors have been detected in Swift logs'
    severity: 'warning'
    no_data_policy: 'okay'
    enabled: 'true'
    trigger:
      logical_operator: 'or'
      rules:
        - metric: log_messages
          fields:
            service: 'swift'
            level: 'error'
          relational_operator: '>'
          threshold: 0.1
          window: 70
```

```
        periods: 0
        function: max
- name: 'cinder-api-http-errors'
  description: 'Too many 5xx HTTP errors have been detected on cinder-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: haproxy_backend_response_5xx
        fields:
          backend: 'cinder-api'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 1
        function: diff
- name: 'cinder-logs-error'
  description: 'Too many errors have been detected in Cinder logs'
  severity: 'warning'
  no_data_policy: 'okay'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: log_messages
        fields:
          service: 'cinder'
          level: 'error'
        relational_operator: '>'
        threshold: 0.1
        window: 70
        periods: 0
        function: max
- name: 'glance-api-http-errors'
  description: 'Too many 5xx HTTP errors have been detected on glance-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: haproxy_backend_response_5xx
        fields:
          backend: 'glance-api'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 1
        function: diff
- name: 'glance-logs-error'
  description: 'Too many errors have been detected in Glance logs'
  severity: 'warning'
  no_data_policy: 'okay'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: log_messages
```

```
        fields:
          service: 'glance'
          level: 'error'
        relational_operator: '>'
        threshold: 0.1
        window: 70
        periods: 0
        function: max
- name: 'neutron-api-http-errors'
  description: 'Too many 5xx HTTP errors have been detected on neutron-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: haproxy_backend_response_5xx
        fields:
          backend: 'neutron-api'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 1
        function: diff
- name: 'neutron-logs-error'
  description: 'Too many errors have been detected in Neutron logs'
  severity: 'warning'
  no_data_policy: 'okay'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: log_messages
        fields:
          service: 'neutron'
          level: 'error'
        relational_operator: '>'
        threshold: 0.1
        window: 70
        periods: 0
        function: max
- name: 'keystone-response-time-duration'
  description: 'Keystone API is too slow'
  severity: 'warning'
  no_data_policy: 'okay'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: openstack_keystone_http_response_times
        fields:
          http_method: '== GET || == POST'
          http_status: '!= 5xx'
        relational_operator: '>'
        threshold: 0.3
        window: 60
        periods: 0
        value: upper_90
        function: max
```

```
- name: 'keystone-public-api-http-errors'
  description: 'Too many 5xx HTTP errors have been detected on keystone-public-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: haproxy_backend_response_5xx
        fields:
          backend: 'keystone-public-api'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 1
        function: diff
- name: 'keystone-admin-api-http-errors'
  description: 'Too many 5xx HTTP errors have been detected on keystone-admin-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: haproxy_backend_response_5xx
        fields:
          backend: 'keystone-admin-api'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 1
        function: diff
- name: 'horizon-web-http-errors'
  description: 'Too many 5xx HTTP errors have been detected on horizon'
  severity: 'warning'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: haproxy_backend_response_5xx
        fields:
          backend: 'horizon-web || horizon-https'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 1
        function: diff
- name: 'keystone-logs-error'
  description: 'Too many errors have been detected in Keystone logs'
  severity: 'warning'
  no_data_policy: 'okay'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: log_messages
        fields:
          service: 'keystone'
          level: 'error'
        relational_operator: '>'
```

```
        threshold: 0.1
        window: 70
        periods: 0
        function: max
- name: 'mysql-node-connected'
  description: 'The MySQL service has lost connectivity with the other nodes'
  severity: 'critical'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: mysql_cluster_connected
        relational_operator: '=='
        threshold: 0
        window: 30
        periods: 1
        function: min
- name: 'mysql-node-ready'
  description: "The MySQL service isn't ready to serve queries"
  severity: 'critical'
  enabled: 'true'
  trigger:
    logical_operator: 'or'
    rules:
      - metric: mysql_cluster_ready
        relational_operator: '=='
        threshold: 0
        window: 30
        periods: 1
        function: min
- name: 'ceph-health-critical'
  description: 'Ceph health is critical'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: ceph_health
        relational_operator: '=='
        threshold: 3 # HEALTH_ERR
        window: 60
        function: max
- name: 'ceph-health-warning'
  description: 'Ceph health is warning'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: ceph_health
        relational_operator: '=='
        threshold: 2 # HEALTH_WARN
        window: 60
        function: max
- name: 'ceph-capacity-critical'
  description: 'Ceph free capacity is too low'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
```

```
        - metric: ceph_pool_total_percent_free
          relational_operator: '<'
          threshold: 2
          window: 60
          function: max
  - name: 'ceph-capacity-warning'
    description: 'Ceph free capacity is low'
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: ceph_pool_total_percent_free
          relational_operator: '<'
          threshold: 5
          window: 60
          function: max
  - name: 'elasticsearch-health-critical'
    description: 'Elasticsearch cluster health is critical'
    severity: 'critical'
    enabled: 'true'
    trigger:
      rules:
        - metric: elasticsearch_cluster_health
          relational_operator: '=='
          threshold: 3 # red
          window: 60
          function: min
  - name: 'elasticsearch-health-warning'
    description: 'Elasticsearch health is warning'
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: elasticsearch_cluster_health
          relational_operator: '=='
          threshold: 2 # yellow
          window: 60
          function: min
  - name: 'elasticsearch-fs-warning'
    description: "The filesystem's free space is low (Elasticsearch node)"
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: fs_space_percent_free
          fields:
            fs: '/opt/es/data' # Real FS is /opt/es-data but Collectd substituted '/'␣
↪by '-'
          relational_operator: '<'
          threshold: 20 # The low watermark for disk usage is 85% by default
          window: 60
          periods: 0
          function: min
  - name: 'elasticsearch-fs-critical'
    description: "The filesystem's free space is too low (Elasticsearch node)"
    severity: 'critical'
    enabled: 'true'
    trigger:
```

```
      rules:
        - metric: fs_space_percent_free
          fields:
            fs: '/opt/es/data' # Real FS is /opt/es-data but Collectd substituted '/'␣
↪by '-'
          relational_operator: '<'
          threshold: 15 # The high watermark for disk usage is 90% by default
          window: 60
          periods: 0
          function: min
  - name: 'influxdb-fs-warning'
    description: "The filesystem's free space is low (InfluxDB node)"
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: fs_space_percent_free
          fields:
            fs: '/var/lib/influxdb'
          relational_operator: '<'
          threshold: 10
          window: 60
          periods: 0
          function: min
  - name: 'influxdb-fs-critical'
    description: "The filesystem's free space is too low (InfluxDB node)"
    severity: 'critical'
    enabled: 'true'
    trigger:
      rules:
        - metric: fs_space_percent_free
          fields:
            fs: '/var/lib/influxdb'
          relational_operator: '<'
          threshold: 5
          window: 60
          periods: 0
          function: min
  - name: 'haproxy-check'
    description: "HAProxy cannot be checked"
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_check
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'rabbitmq-check'
    description: "RabbitMQ cannot be checked"
    # This alarm's severity is warning because the effective status of the
    # RabbitMQ cluster is computed by rabbitmq-pacemaker-* alarms.
    # This alarm is still useful because it will report the node(s) on which
    # RabbitMQ isn't running.
    severity: 'warning'
    enabled: 'true'
```

```
    trigger:
      rules:
        - metric: rabbitmq_check
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'ceph-mon-check'
    description: "Ceph monitor cannot be checked"
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: ceph_mon_check
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'ceph-osd-check'
    description: "Ceph OSD cannot be checked"
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: ceph_osd_check
          relational_operator: '=='
          threshold: 0
          window: 80  # The metric interval collection is 60s
          periods: 0
          function: last
  - name: 'pacemaker-check'
    description: "Pacemaker cannot be checked"
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: pacemaker_check
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'elasticsearch-check'
    description: "Elasticsearch cannot be checked"
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: elasticsearch_check
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'influxdb-check'
    description: "InfluxDB cannot be checked"
```

```
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: influxdb_check
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'libvirt-check'
    description: "Libvirt cannot be checked"
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: libvirt_check
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'memcached-check'
    description: "memcached cannot be checked"
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: memcached_check
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'mysql-check'
    description: "MySQL cannot be checked"
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: mysql_check
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'network-warning-dropped-rx'
    description: "Some received packets have been dropped"
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: if_dropped_rx
          relational_operator: '>'
          threshold: 100
          window: 60
          periods: 0
          function: avg
```

```
- name: 'network-critical-dropped-rx'
  description: "Too many received packets have been dropped"
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: if_dropped_rx
        relational_operator: '>'
        threshold: 1000
        window: 60
        periods: 0
        function: avg
- name: 'network-warning-dropped-tx'
  description: "Some transmitted packets have been dropped"
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: if_dropped_tx
        relational_operator: '>'
        threshold: 100
        window: 60
        periods: 0
        function: avg
- name: 'network-critical-dropped-tx'
  description: "Too many transmitted packets have been dropped"
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: if_dropped_tx
        relational_operator: '>'
        threshold: 1000
        function: avg
        window: 60
- name: 'instance-creation-time-warning'
  description: "Instance creation takes too much time"
  severity: 'warning'
  no_data_policy: 'okay' # This is a sporadic metric
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_instance_creation_time
        relational_operator: '>'
        threshold: 20
        window: 600
        periods: 0
        function: avg
- name: 'hdd-errors-critical'
  description: 'Errors on hard drive(s) have been detected'
  severity: 'critical'
  enabled: 'true'
  no_data_policy: okay
  trigger:
    rules:
      - metric: hdd_errors_rate
        group_by: ['device']
        relational_operator: '>'
```

```
        threshold: 0
        window: 60
        periods: 0
        function: max
- name: 'total-nova-free-vcpu-warning'
  description: 'There is none VCPU available for new instances'
  severity: 'warning'
  enabled: 'true'
  no_data_policy: skip # the metric is only collected from the aggregator node
  trigger:
    rules:
      - metric: openstack_nova_total_free_vcpus
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: max
- name: 'total-nova-free-memory-warning'
  description: 'There is none memory available for new instances'
  severity: 'warning'
  enabled: 'true'
  no_data_policy: skip  # the metric is only collected from the aggregator node
  trigger:
    rules:
      - metric: openstack_nova_total_free_ram
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: max
- name: 'nova-aggregates-free-memory-warning'
  description: "The nova aggregates free memory percent is low"
  severity: 'warning'
  enabled: 'true'
  no_data_policy: skip # the metric is only collected from the aggregator node
  trigger:
    rules:
      - metric: openstack_nova_aggregate_free_ram_percent
        group_by: [aggregate]
        relational_operator: '<'
        threshold: 10.0
        window: 60
        periods: 0
        function: min
- name: 'nova-aggregates-free-memory-critical'
  description: "The nova aggregates free memory percent is too low"
  severity: 'critical'
  enabled: 'true'
  no_data_policy: skip # the metric is only collected from the aggregator node
  trigger:
    rules:
      - metric: openstack_nova_aggregate_free_ram_percent
        group_by: [aggregate]
        relational_operator: '<'
        threshold: 1.0
        window: 60
        periods: 0
        function: min
```

```
# Adds alarm on local check for OpenStack services endpoint
- name: 'cinder-api-local-endpoint'
  description: 'Cinder API is locally down'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_check_local_api
        fields:
          service: 'cinder-api'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'glance-api-local-endpoint'
  description: 'Glance API is locally down'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_check_local_api
        fields:
          service: 'glance-api'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'heat-api-local-endpoint'
  description: 'Heat API is locally down'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_check_local_api
        fields:
          service: 'heat-api'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'heat-cfn-api-local-endpoint'
  description: 'Heat CFN API is locally down'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_check_local_api
        fields:
          service: 'heat-cfn-api'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
```

```
  - name: 'keystone-public-api-local-endpoint'
    description: 'Keystone public API is locally down'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_local_api
          fields:
            service: 'keystone-public-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'neutron-api-local-endpoint'
    description: 'Neutron API is locally down'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_local_api
          fields:
            service: 'neutron-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'nova-api-local-endpoint'
    description: 'Nova API is locally down'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_local_api
          fields:
            service: 'nova-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'swift-api-local-endpoint'
    description: 'Swift API is locally down'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_local_api
          fields:
            service: 'swift-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last

  # Following are the OpenStack service check API definitions and
```

```
  # also InfluxDB API
  - name: 'influxdb-api-check-failed'
    description: 'Endpoint check for InfluxDB is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: http_check
          fields:
            service: 'influxdb-cluster'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'nova-api-check-failed'
    description: 'Endpoint check for nova-api is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
          fields:
            service: 'nova-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'neutron-api-check-failed'
    description: 'Endpoint check for neutron-api is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
          fields:
            service: 'neutron-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'cinder-api-check-failed'
    description: 'Endpoint check for cinder-api is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
```

```
          fields:
            service: 'cinder-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'cinder-v2-api-check-failed'
    description: 'Endpoint check for cinder-v2-api is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
          fields:
            service: 'cinder-v2-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'glance-api-check-failed'
    description: 'Endpoint check for glance-api is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
          fields:
            service: 'glance-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'heat-api-check-failed'
    description: 'Endpoint check for heat-api is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
          fields:
            service: 'heat-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'heat-cfn-api-check-failed'
    description: 'Endpoint check for heat-cfn-api is failed'
    severity: 'down'
```

```
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
          fields:
            service: 'heat-cfn-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'swift-api-check-failed'
    description: 'Endpoint check for swift-api is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
          fields:
            service: 'swift-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'swift-s3-api-check-failed'
    description: 'Endpoint check for swift-s3-api is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
          fields:
            service: 'swift-s3-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'keystone-public-api-check-failed'
    description: 'Endpoint check for keystone-public-api is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
          fields:
            service: 'keystone-public-api'
          relational_operator: '=='
          threshold: 0
```

```
          window: 60
          periods: 0
          function: last
  - name: 'ceilometer-api-check-failed'
    description: 'Endpoint check for ceilometer-api is failed'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the controller running␣
↪the management VIP
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_check_api
          fields:
            service: 'ceilometer-api'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last

  # Following are the AFD generated to check API backends
  # All backends are down
  - name: 'elasticsearch-api-backends-all-down'
    description: 'All Elasticsearch backends are down'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'elasticsearch-rest'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'kibana-api-backends-all-down'
    description: 'All API backends are down for Kibana'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'kibana'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'influxdb-api-backends-all-down'
    description: 'All API backends are down for InfluxDB'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
```

```
      - metric: haproxy_backend_servers
        fields:
          backend: 'influxdb'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'grafana-api-backends-all-down'
  description: 'All API backends are down for Grafana'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'grafana'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'glance-registry-api-backends-all-down'
  description: 'All API backends are down for glance-registry-api'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'glance-registry-api'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-api-backends-all-down'
  description: 'All API backends are down for nova-api'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'nova-api'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'cinder-api-backends-all-down'
  description: 'All API backends are down for cinder-api'
  severity: 'down'
  enabled: 'true'
```

```
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'cinder-api'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'object-storage-api-backends-all-down'
    description: 'All API backends are down for object-storage'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'object-storage'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'heat-cfn-api-backends-all-down'
    description: 'All API backends are down for heat-cfn-api'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'heat-cfn-api'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'horizon-web-api-backends-all-down'
    description: 'All API backends are down for horizon-web'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'horizon-web || horizon-https'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'nova-novncproxy-websocket-api-backends-all-down'
    description: 'All API backends are down for nova-novncproxy-websocket'
```

```
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'nova-novncproxy-websocket'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'heat-api-backends-all-down'
    description: 'All API backends are down for heat-api'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'heat-api'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'keystone-public-api-backends-all-down'
    description: 'All API backends are down for keystone-public-api'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'keystone-public-api'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'heat-cloudwatch-api-backends-all-down'
    description: 'All API backends are down for heat-cloudwatch-api'
    severity: 'down'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'heat-cloudwatch-api'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
```

```
- name: 'nova-metadata-api-backends-all-down'
  description: 'All API backends are down for nova-metadata-api'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'nova-metadata-api'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'mysqld-tcp-api-backends-all-down'
  description: 'All API backends are down for mysqld-tcp'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'mysqld-tcp'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'keystone-admin-api-backends-all-down'
  description: 'All API backends are down for keystone-admin-api'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'keystone-admin-api'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'glance-api-backends-all-down'
  description: 'All API backends are down for glance-api'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'glance-api'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
```

```
        periods: 0
        function: last
- name: 'neutron-api-backends-all-down'
  description: 'All API backends are down for neutron-api'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'neutron-api'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'swift-api-backends-all-down'
  description: 'All API backends are down for swift-api'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'swift-api || object-storage'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'ceilometer-api-backends-all-down'
  description: 'All API backends are down for ceilometer-api'
  severity: 'down'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'ceilometer-api'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
# At least one backend is down
- name: 'elasticsearch-api-backends-one-down'
  description: 'At least one API backend is down for elasticsearch'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'elasticsearch-rest'
          state: 'down'
```

```
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'kibana-api-backends-one-down'
  description: 'At least one API backend is down for kibana'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'kibana'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'influxdb-api-backends-one-down'
  description: 'At least one API backend is down for influxdb'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'influxdb'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'grafana-api-backends-one-down'
  description: 'At least one API backend is down for grafana'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'grafana'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'glance-registry-api-backends-one-down'
  description: 'At least one API backend is down for glance-registry-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
```

```
          backend: 'glance-registry-api'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-api-backends-one-down'
  description: 'At least one API backend is down for nova-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'nova-api'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'cinder-api-backends-one-down'
  description: 'At least one API backend is down for cinder-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'cinder-api'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'object-storage-api-backends-one-down'
  description: 'At least one API backend is down for object-storage'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'object-storage'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'heat-cfn-api-backends-one-down'
  description: 'At least one API backend is down for heat-cfn-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
```

```
      - metric: haproxy_backend_servers
        fields:
          backend: 'heat-cfn-api'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'horizon-web-api-backends-one-down'
  description: 'At least one API backend is down for horizon-web'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'horizon-web || horizon-https'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-novncproxy-websocket-api-backends-one-down'
  description: 'At least one API backend is down for nova-novncproxy-websocket'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'nova-novncproxy-websocket'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'heat-api-backends-one-down'
  description: 'At least one API backend is down for heat-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'heat-api'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'keystone-public-api-backends-one-down'
  description: 'At least one API backend is down for keystone-public-api'
  severity: 'warning'
  enabled: 'true'
```

```
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'keystone-public-api'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'heat-cloudwatch-api-backends-one-down'
  description: 'At least one API backend is down for heat-cloudwatch-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'heat-cloudwatch-api'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-metadata-api-backends-one-down'
  description: 'At least one API backend is down for nova-metadata-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'nova-metadata-api'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'mysqld-tcp-api-backends-one-down'
  description: 'At least one API backend is down for mysqld-tcp'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'mysqld-tcp'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'keystone-admin-api-backends-one-down'
  description: 'At least one API backend is down for keystone-admin-api'
```

```
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'keystone-admin-api'
            state: 'down'
          relational_operator: '>'
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'glance-api-backends-one-down'
    description: 'At least one API backend is down for glance-api'
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'glance-api'
            state: 'down'
          relational_operator: '>'
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'neutron-api-backends-one-down'
    description: 'At least one API backend is down for neutron-api'
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'neutron-api'
            state: 'down'
          relational_operator: '>'
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'swift-api-backends-one-down'
    description: 'At least one API backend is down for swift-api'
    severity: 'warning'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers
          fields:
            backend: 'swift-api || object-storage'
            state: 'down'
          relational_operator: '>'
          threshold: 0
          window: 60
          periods: 0
          function: last
```

```
- name: 'ceilometer-api-backends-one-down'
  description: 'At least one API backend is down for ceilometer-api'
  severity: 'warning'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers
        fields:
          backend: 'ceilometer-api'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
# Less than 50% of backends are up
- name: 'elasticsearch-api-backends-majority-down'
  description: 'Less than 50% of backends are up for elasticsearch'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'elasticsearch-rest'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'kibana-api-backends-majority-down'
  description: 'Less than 50% of backends are up for kibana'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'kibana'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'influxdb-api-backends-majority-down'
  description: 'Less than 50% of backends are up for influxdb'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'influxdb'
          state: 'up'
        relational_operator: '<='
        threshold: 50
```

```
        window: 60
        periods: 0
        function: last
- name: 'grafana-api-backends-majority-down'
  description: 'Less than 50% of backends are up for grafana'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'grafana'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'glance-registry-api-backends-majority-down'
  description: 'Less than 50% of backends are up for glance-registry-api'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'glance-registry-api'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'nova-api-backends-majority-down'
  description: 'Less than 50% of backends are up for nova-api'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'nova-api'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'cinder-api-backends-majority-down'
  description: 'Less than 50% of backends are up for cinder-api'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'cinder-api'
          state: 'up'
```

```
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'object-storage-api-backends-majority-down'
  description: 'Less than 50% of backends are up for object-storage'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'object-storage'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'heat-cfn-api-backends-majority-down'
  description: 'Less than 50% of backends are up for heat-cfn-api'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'heat-cfn-api'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'horizon-web-api-backends-majority-down'
  description: 'Less than 50% of backends are up for horizon-web'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'horizon-web || horizon-https'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'nova-novncproxy-websocket-api-backends-majority-down'
  description: 'Less than 50% of backends are up for nova-novncproxy-websocket'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
```

```
        fields:
          backend: 'nova-novncproxy-websocket'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'heat-api-backends-majority-down'
  description: 'Less than 50% of backends are up for heat-api'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'heat-api'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'keystone-public-api-backends-majority-down'
  description: 'Less than 50% of backends are up for keystone-public-api'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'keystone-public-api'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'heat-cloudwatch-api-backends-majority-down'
  description: 'Less than 50% of backends are up for heat-cloudwatch-api'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'heat-cloudwatch-api'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'nova-metadata-api-backends-majority-down'
  description: 'Less than 50% of backends are up for nova-metadata-api'
  severity: 'critical'
  enabled: 'true'
  trigger:
```

```
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'nova-metadata-api'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'mysqld-tcp-api-backends-majority-down'
  description: 'Less than 50% of backends are up for mysqld-tcp'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'mysqld-tcp'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'keystone-admin-api-backends-majority-down'
  description: 'Less than 50% of backends are up for keystone-admin-api'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'keystone-admin-api'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'glance-api-backends-majority-down'
  description: 'Less than 50% of backends are up for glance-api'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: haproxy_backend_servers_percent
        fields:
          backend: 'glance-api'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'neutron-api-backends-majority-down'
  description: 'Less than 50% of backends are up for neutron-api'
  severity: 'critical'
```

```
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers_percent
          fields:
            backend: 'neutron-api'
            state: 'up'
          relational_operator: '<='
          threshold: 50
          window: 60
          periods: 0
          function: last
  - name: 'swift-api-backends-majority-down'
    description: 'Less than 50% of backends are up for swift-api'
    severity: 'critical'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers_percent
          fields:
            backend: 'swift-api || object-storage'
            state: 'up'
          relational_operator: '<='
          threshold: 50
          window: 60
          periods: 0
          function: last
  - name: 'ceilometer-api-backends-majority-down'
    description: 'Less than 50% of backends are up for ceilometer-api'
    severity: 'critical'
    enabled: 'true'
    trigger:
      rules:
        - metric: haproxy_backend_servers_percent
          fields:
            backend: 'ceilometer-api'
            state: 'up'
          relational_operator: '<='
          threshold: 50
          window: 60
          periods: 0
          function: last

  # Following are the AFD generated to check workers
  # All workers are down
  - name: 'nova-scheduler-all-down'
    description: 'All Nova schedulers are down'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_nova_services
          fields:
            service: 'scheduler'
            state: 'up'
          relational_operator: '=='
          threshold: 0
```

```
        window: 60
        periods: 0
        function: last
- name: 'nova-cert-all-down'
  description: 'All Nova certs are down'
  severity: 'down'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_services
        fields:
          service: 'cert'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-consoleauth-all-down'
  description: 'All Nova consoleauths are down'
  severity: 'down'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_services
        fields:
          service: 'consoleauth'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-compute-all-down'
  description: 'All Nova computes are down'
  severity: 'down'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_services
        fields:
          service: 'compute'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-conductor-all-down'
  description: 'All Nova conductors are down'
  severity: 'down'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
```

```
      - metric: openstack_nova_services
        fields:
          service: 'conductor'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'cinder-scheduler-all-down'
  description: 'All Cinder schedulers are down'
  severity: 'down'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_cinder_services
        fields:
          service: 'scheduler'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'cinder-volume-all-down'
  description: 'All Cinder volumes are down'
  severity: 'down'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_cinder_services
        fields:
          service: 'volume'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'neutron-l3-all-down'
  description: 'All Neutron L3 agents are down'
  severity: 'down'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_neutron_agents
        fields:
          service: 'l3'
          state: 'up'
        relational_operator: '=='
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'neutron-dhcp-all-down'
```

```
    description: 'All Neutron DHCP agents are down'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_neutron_agents
          fields:
            service: 'dhcp'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'neutron-metadata-all-down'
    description: 'All Neutron metadata agents are down'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_neutron_agents
          fields:
            service: 'metadata'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'neutron-openvswitch-all-down'
    description: 'All Neutron openvswitch agents are down'
    severity: 'down'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_neutron_agents
          fields:
            service: 'openvswitch'
            state: 'up'
          relational_operator: '=='
          threshold: 0
          window: 60
          periods: 0
          function: last
  # At least one backend is down
  - name: 'nova-scheduler-one-down'
    description: 'At least one Nova scheduler is down'
    severity: 'warning'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_nova_services
          fields:
            service: 'scheduler'
```

```
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-cert-one-down'
  description: 'At least one Nova cert is down'
  severity: 'warning'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_services
        fields:
          service: 'cert'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-consoleauth-one-down'
  description: 'At least one Nova consoleauth is down'
  severity: 'warning'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_services
        fields:
          service: 'consoleauth'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-compute-one-down'
  description: 'At least one Nova compute is down'
  severity: 'warning'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_services
        fields:
          service: 'compute'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'nova-conductor-one-down'
  description: 'At least one Nova conductor is down'
  severity: 'warning'
  no_data_policy: 'skip' # the metric is only collected from the DC node
```

```
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_nova_services
          fields:
            service: 'conductor'
            state: 'down'
          relational_operator: '>'
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'cinder-scheduler-one-down'
    description: 'At least one Cinder scheduler is down'
    severity: 'warning'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_cinder_services
          fields:
            service: 'scheduler'
            state: 'down'
          relational_operator: '>'
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'cinder-volume-one-down'
    description: 'At least one Cinder volume is down'
    severity: 'warning'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_cinder_services
          fields:
            service: 'volume'
            state: 'down'
          relational_operator: '>'
          threshold: 0
          window: 60
          periods: 0
          function: last
  - name: 'neutron-l3-one-down'
    description: 'At least one L3 agent is down'
    severity: 'warning'
    no_data_policy: 'skip' # the metric is only collected from the DC node
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_neutron_agents
          fields:
            service: 'l3'
            state: 'down'
          relational_operator: '>'
          threshold: 0
          window: 60
```

```
        periods: 0
        function: last
- name: 'neutron-dhcp-one-down'
  description: 'At least one DHCP agent is down'
  severity: 'warning'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_neutron_agents
        fields:
          service: 'dhcp'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'neutron-metadata-one-down'
  description: 'At least one metadata agents is down'
  severity: 'warning'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_neutron_agents
        fields:
          service: 'metadata'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
- name: 'neutron-openvswitch-one-down'
  description: 'At least one openvswitch agents is down'
  severity: 'warning'
  no_data_policy: 'skip' # the metric is only collected from the DC node
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_neutron_agents
        fields:
          service: 'openvswitch'
          state: 'down'
        relational_operator: '>'
        threshold: 0
        window: 60
        periods: 0
        function: last
# Less than 50% of service are up (compared to up and down).
- name: 'nova-scheduler-majority-down'
  description: 'Less than 50% of Nova schedulers are up'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_services_percent
```

```
        fields:
          service: 'scheduler'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'nova-cert-majority-down'
  description: 'Less than 50% of Nova certs are up'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_services_percent
        fields:
          service: 'cert'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'nova-consoleauth-majority-down'
  description: 'Less than 50% of Nova consoleauths are up'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_services_percent
        fields:
          service: 'consoleauth'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'nova-compute-majority-down'
  description: 'Less than 50% of Nova computes are up'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_nova_services_percent
        fields:
          service: 'compute'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'nova-conductor-majority-down'
  description: 'Less than 50% of Nova conductors are up'
  severity: 'critical'
  enabled: 'true'
  trigger:
```

```
    rules:
      - metric: openstack_nova_services_percent
        fields:
          service: 'conductor'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'cinder-scheduler-majority-down'
  description: 'Less than 50% of Cinder schedulers are up'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_cinder_services_percent
        fields:
          service: 'scheduler'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'cinder-volume-majority-down'
  description: 'Less than 50% of Cinder volumes are up'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_cinder_services_percent
        fields:
          service: 'volume'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'neutron-l3-majority-down'
  description: 'Less than 50% of Neutron L3 agents are up'
  severity: 'critical'
  enabled: 'true'
  trigger:
    rules:
      - metric: openstack_neutron_agents_percent
        fields:
          service: 'l3'
          state: 'up'
        relational_operator: '<='
        threshold: 50
        window: 60
        periods: 0
        function: last
- name: 'neutron-dhcp-majority-down'
  description: 'Less than 50% of Neutron DHCP agents are up'
  severity: 'critical'
```

```
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_neutron_agents_percent
          fields:
            service: 'dhcp'
            state: 'up'
          relational_operator: '<='
          threshold: 50
          window: 60
          periods: 0
          function: last
  - name: 'neutron-metadata-majority-down'
    description: 'Less than 50% of Neutron metadata agents are up'
    severity: 'critical'
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_neutron_agents_percent
          fields:
            service: 'metadata'
            state: 'up'
          relational_operator: '<='
          threshold: 50
          window: 60
          periods: 0
          function: last
  - name: 'neutron-openvswitch-majority-down'
    description: 'Less than 50% of Neutron openvswitch agents are up'
    severity: 'critical'
    enabled: 'true'
    trigger:
      rules:
        - metric: openstack_neutron_agents_percent
          fields:
            service: 'openvswitch'
            state: 'up'
          relational_operator: '<='
          threshold: 50
          window: 60
          periods: 0
          function: last
```