The Elasticsearch-Kibana plugin for Fuel Documentation

Release 0.9-0.9.0-1

Mirantis Inc.

April 26, 2016

CONTENTS

CHAPTER

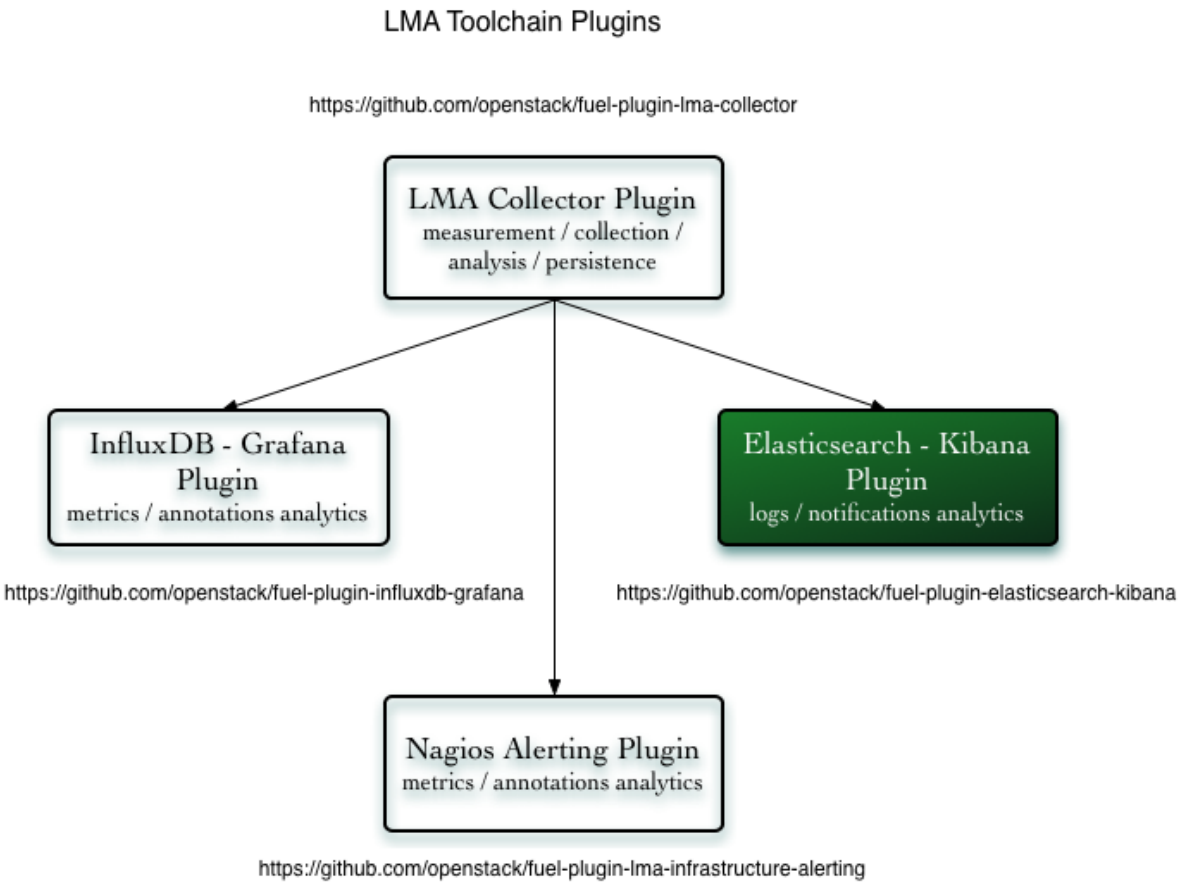
ONE

USER DOCUMENTATION

1.1 Overview

The **Elasticsearch-Kibana Fuel Plugin** is used to install and configure Elasticsearch and Kibana which collectively provide access to the OpenStack logs and notifications analytics. Those analytics can be used to search and correlate service-affecting events which occurred in your OpenStack environment. It is an indispensable tool to troubleshooting problems.

Elasticsearch and Kibana are key components of the LMA Toolchain project as shown in the figure below.



1.1.1 Requirements

Requirement	Version/Comment
Disk space	The plugin's specification requires to provision at least 15GB of disk space for the system, 10GB for the logs and 30GB for the database. As a result, the installation of the plugin will fail if there is less than 55GB of disk space available on the node.
Mirantis OpenStack	8.0
Hardware configuration	The hardware configuration (RAM, CPU, disk) required by this plugin depends on the size of your cloud environment and other parameters like the retention period and log level. A typical setup would at least require a quad-core server with 8GB of RAM and fast disks (ideally, SSDs). The actual disk space you need to run the plugin depends on several factors including the size of your OpenStack environment, the retention period, the logging level and workload. The more of the above, the more disk space you will need to run the Elaticsearch-Kibana Plugin. It is also highly recommended to use dedicated disk(s) for your data storage.

1.1.2 Limitations

Currently, the maximum size of an Elasticsearch cluster that can be installed by Fuel is limited to five nodes. Each node of an Elasticsearch cluster is configured as *master candidate* and a *storage node*. This means, that each node of the Elasticsearch cluster can be elected as a master and all nodes will store data.

The *cluster operations* can require some manual operations some times.

1.1.3 Key terms, acronyms and abbreviations

Terms & acronyms	Definition
LMA Collector	Logging, Monitoring and Alerting (LMA) Collector. A service running on each node which collects all the logs and the OpenStak notifications.
Elasticsearch	An open source (Apache Licensed) application based on the Lucene™ search engine that makes data like log messages easy to explore and correlate. Elasticsearch is written in Java and uses Lucene internally for all of its indexing and searching, but it aims to make full-text search easy by hiding the complexities of Lucene behind a simple, coherent, RESTful API.
Kibana	An open source (Apache Licensed), browser based analytics and search dashboard for Elasticsearch. Kibana is easy to setup and start using.

1.2 Release Notes

1.2.1 Version 0.9.0

- Support Elasticsearch and Kibana clustering for scale-out and high availability of those services.
- Upgrade to Elasticsearch 1.7.4.
- Upgrade to Kibana 3.1.3.

1.2.2 Version 0.8.0

- Add support for the "elasticsearch_kibana" Fuel Plugin role instead of the "base-os" role which had several limitations.
- Add support for retention policy configuration with Elastic Curator.
- Upgrade to Elasticsearch 1.4.5.

1.2.3 Version 0.7.0

- Initial release of the plugin. This is a beta version.

1.3 Installation Guide

1.3.1 Elasticsearch-Kibana Fuel Plugin Installation using the RPM file of the Fuel Plugins Catalog

To install the Elasticsearch-Kibana Fuel Plugin using the RPM file of the Fuel Plugins Catalog, you need to follow these steps:

1. Download the RPM file from the Fuel Plugins Catalog.
2. Copy the RPM file to the Fuel Master node:

```
[root@home ~]# scp elasticsearch_kibana-0.9-0.9.0-0.noarch.rpm \
root@<Fuel Master node IP address>:
```

3. Install the plugin using the Fuel CLI:

```
[root@fuel ~]# fuel plugins --install elasticsearch_kibana-0.9-0.9.0-0.noarch.rpm
```

4. Verify that the plugin is installed correctly:

```
[root@fuel ~]# fuel plugins --list
id | name                 | version | package_version
---|----------------------|---------|----------------
1  | elasticsearch_kibana | 0.9.0   | 4.0.0
```

1.3.2 Elasticsearch-Kibana Fuel Plugin installation from source

Alternatively, you may want to build the RPM file of the plugin from source if, for example, you want to test the latest features, modify some built-in configuration or implement your own customization. But note that running a Fuel plugin that you have built yourself is at your own risk.

To install Elasticsearch-Kibana Plugin from source, you first need to 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 won't have to copy that file later on.

Prepare an environment for building the plugin on the Fuel Master Node

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*. In this case, please refer to the section "Preparing an environment for plugin development" of the Fuel Plugins wiki, if you need further instructions about how to build the Fuel Plugin Builder.

4. Clone the plugin git repository:

```
[root@home ~] git clone \
  https://github.com/openstack/fuel-plugin-elasticsearch-kibana.git
```

5. Check that the plugin is valid:

```
[root@home ~] fpb --check ./fuel-plugin-elasticsearch-kibana
```

6. And finally, build the plugin:

```
[root@home ~] fpb --build ./fuel-plugin-elasticsearch-kibana
```

7. Now that you have created the RPM file, you can install the plugin using the *fuel plugins –install* command:

```
[root@fuel ~] fuel plugins --install \
  ./fuel-plugin-elasticsearch-kibana/*.noarch.rpm
```

1.3.3 Elasticsearch-Kibana Fuel Plugin software components

List of software components installed by the plugin

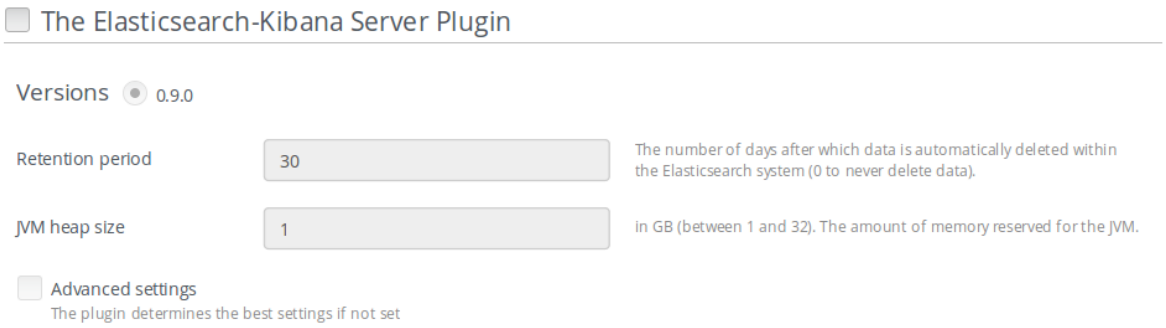
Components	Version
Elasticsearch	v1.7.4 for Ubuntu (64-bit)
Kibana	v3.1.3
Nginx	Version coming by default with the Ubuntu distribution

1.4 User Guide

1.4.1 Plugin configuration

To configure your plugin, you need to follow these steps:

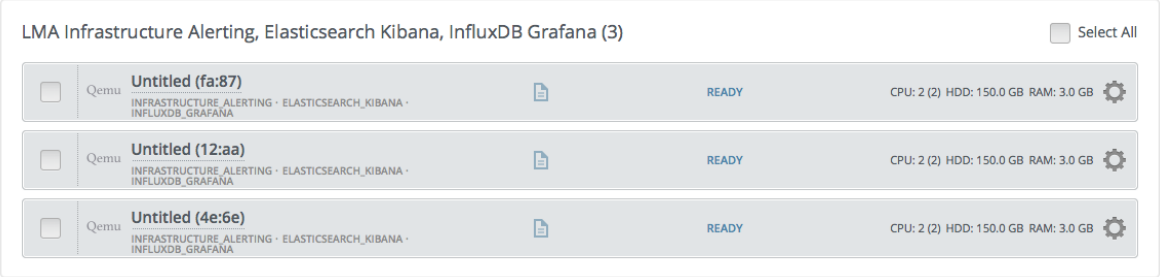
1. Create a new environment from the Fuel web user interface.
2. Click the **Settings** tab and select the **Other** category.
3. Scroll down through the settings until you find the **Elasticsearch-Kibana Server Plugin** section. You should see a page like this.



4. Check the *Elasticsearch-Kibana Server Plugin* box and fill-in the required fields as indicated below.
   (a) Specify the number of days of retention for your data.
   (b) Specify the JVM heap size for Elastisearch. See configuration recommendations below.

Note: By default, 1GB of heap memory is allocated to the Elasticsearch process. This value is too small to run Elasticsearch for anything else than local testing. To run Elasticsearch in production you need to allocate at least 4 GB of memory but it is recommended to allocate 50% of the available memory up to 32 GB maximum. If you set a value that is greater than the memory size, Elasticsearch won't start. Keep in mind also to reserve enough memory for the operating system and the other services.

   At this point, you can choose to edit advanced settings or let the plugin apply sane defaults for you. The advanced settings are used to specify the clustering parameters when the *Elasticsearch-Kibana Server Plugin* is installed on more than one node. To manually configure those advanced settings, check the *Advanced settings* box and fill-in the required parameters.
5. When you are done with the settings, scroll down to the bottom of the page and click the **Save Settings** button.
6. Click the *Nodes* tab and assign the *Elasticsearch_Kibana* role to nodes as shown in the figure below. You can see in this example that the *Elasticsearch_Kibana* role is assigned to three different nodes along with the *Infrastructure_Alerting* role and the *InfluxDB_Grafana* role. This means that the three plugins of the LMA toolchain can be installed on the same nodes.

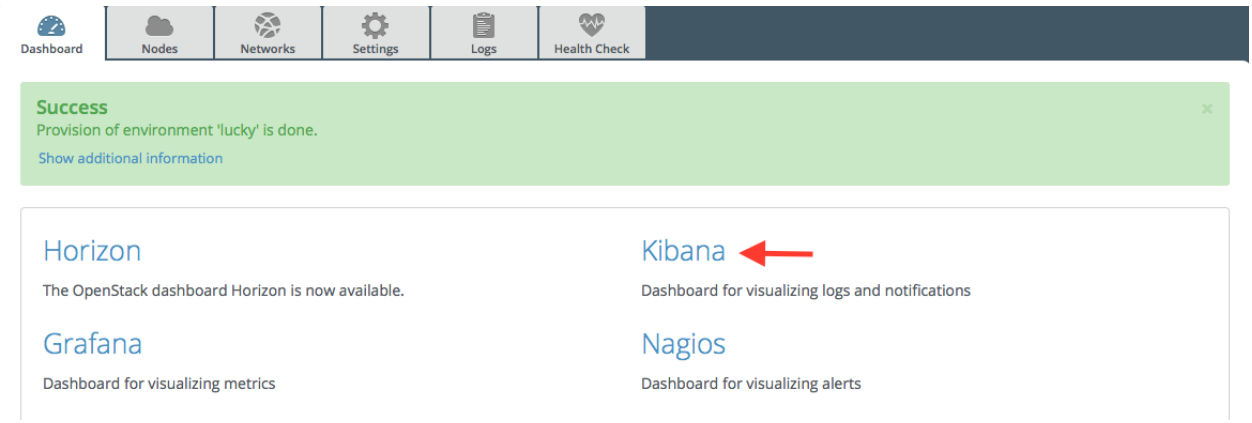


Note: You can assign the *Elasticsearch_Kibana* role up to five nodes. The Elasticsearch clustering for high availability requires that you assign the *Elasticsearch_Kibana* role to at least three nodes. Note also that is possible to add or remove a node with the *Elasticsearch_Kibana* role after deployment.

7. Click on **Apply Changes**
8. Adjust the disk configuration if necessary (see the Fuel User Guide for details). By default, the Elasticsearch-Kibana Plugin allocates:
   - 20% of the first available disk for the operating system by honoring a range of 15GB minimum and 50GB maximum.
   - 10GB for */var/log*.
   - At least 30 GB for the Elasticsearch database in */opt/es-data*.
9. Configure your environment as needed.
10. Verify the networks on the Networks tab of the Fuel web UI.
11. And finally, Deploy your changes.

1.4.2 Plugin verification

Be aware, that depending on the number of nodes and deployment setup, deploying a Mirantis OpenStack environment can typically take anything from 30 minutes to several hours. But once your deployment is complete, you should see a deployment success notification message with a link to the Kibana dashboard as shown in the figure below:



Note: Be aware that Kibana is attached to the *management network*. Your desktop machine must have access to the OpenStack environment's *management network* you just created, to get access to the Kibana dashboard

Verifying Elasticsearch

You should verify that the Elasticsearch cluster is running properly. To do that, you need first to retrieve the Elasticsearch cluster VIP address. Here is how to proceed.

1. On the Fuel Master node, find the IP address of a node where the Elasticsearch server is installed using the following command:

```
[root@fuel ~]# fuel nodes
id | status   | name              | cluster | ip         | mac               | .....
---|----------|-------------------|---------|------------|-------------------| .....
1  | ready    | Untitled (fa:87) | 1       | 10.109.0.8 | 64:18:ef:86:fa:87 | .....
2  | ready    | Untitled (12:aa) | 1       | 10.109.0.3 | 64:5f:c6:88:12:aa | .....
3  | ready    | Untitled (4e:6e) | 1       | 10.109.0.7 | 64:ca:bf:a4:4e:6e | .....

.... roles                      |
.... -------------------------|
.... elasticsearch_kibana, ... |
.... elasticsearch_kibana, ... |
.... elasticsearch_kibana, ... |
```

2. Then *ssh* to anyone of these nodes (ex. *node-1*) and type the command:

```
root@node-1:~# hiera lma::elasticsearch::vip
10.109.1.5
```

   This tells you that the VIP address of your Elasticsearch cluster is *10.109.1.5*.

3. With that VIP address type the command:

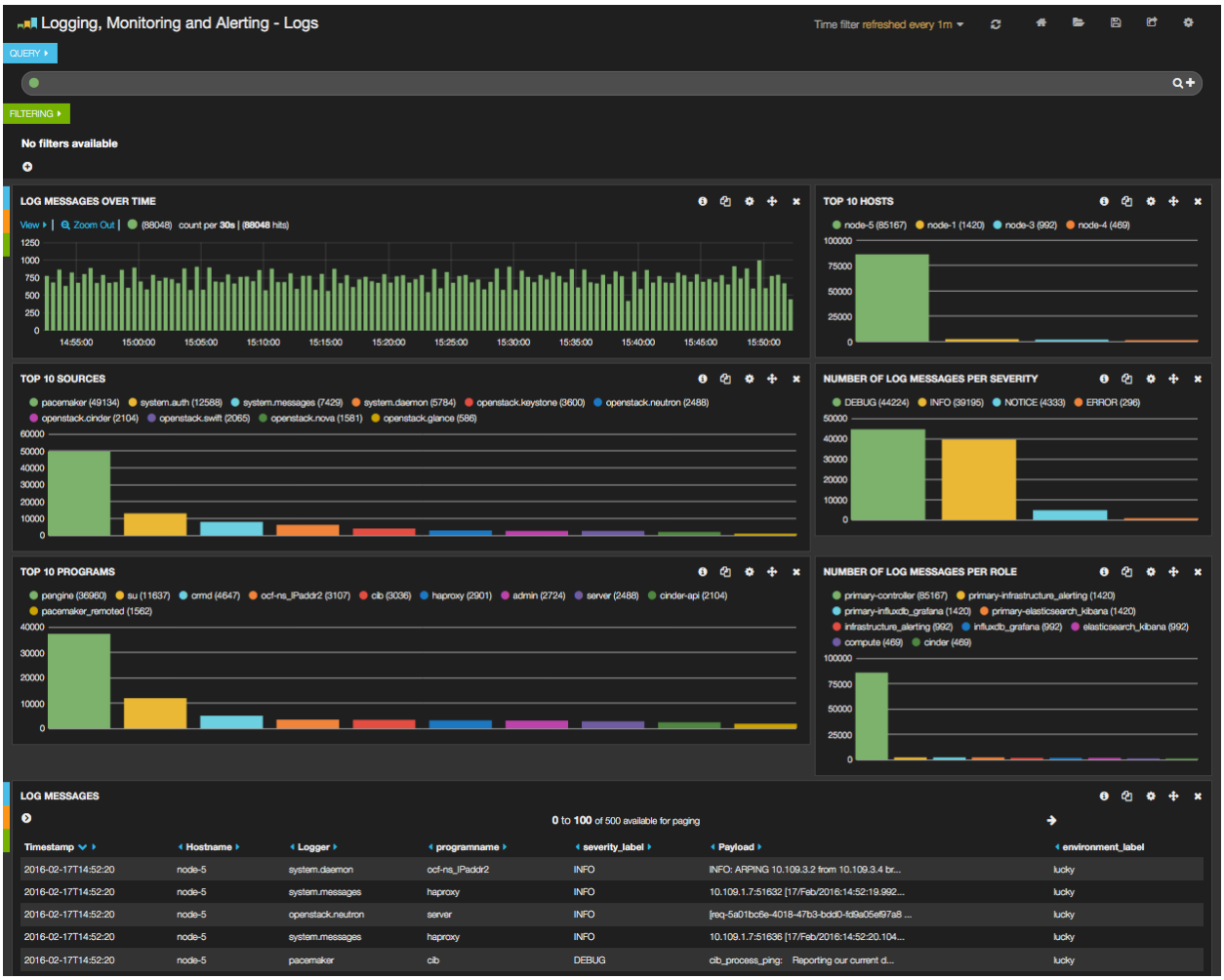
```
curl http://10.109.1.5:9200/
```

   The output should look like this:

```
{
  "status" : 200,
  "name" : "node-3.test.domain.local_es-01",
  "cluster_name" : "lma",
  "version" : {
    "number" : "1.7.4",
    "build_hash" : "0d3159b9fc8bc8e367c5c40c09c2a57c0032b32e",
    "build_timestamp" : "2015-12-15T11:25:18Z",
    "build_snapshot" : false,
    "lucene_version" : "4.10.4"
  },
  "tagline" : "You Know, for Search"
}
```

Verifying Kibana

From the Fuel web UI **Dashboard** view, click on the **Kibana** link, you should be directed to the *Logs Dashboard* as shown in the figure below.

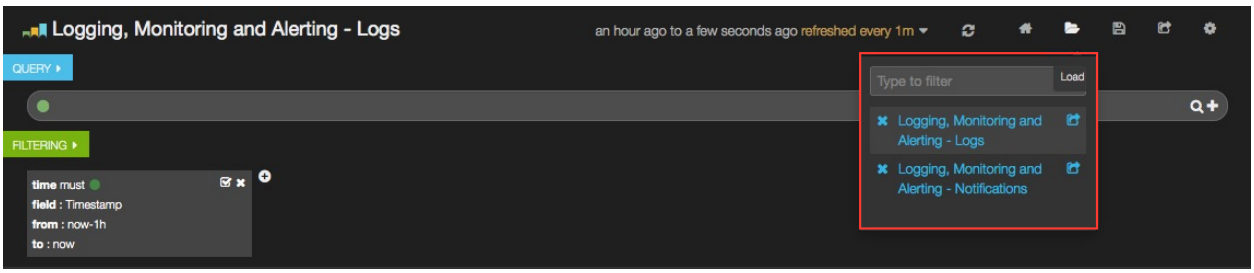


1.4.3 Dashboards management

The *Elasticsearch-Kibana Server Plugin* comes with two predefined dashboards:

- The *Logs Dashboard* which is the Kibana Home Dashboard for viewing the log messages.
- The *Notifications Dashboard* for viewing the OpenStack notifications if you enabled this option in the LMA Collector settings.

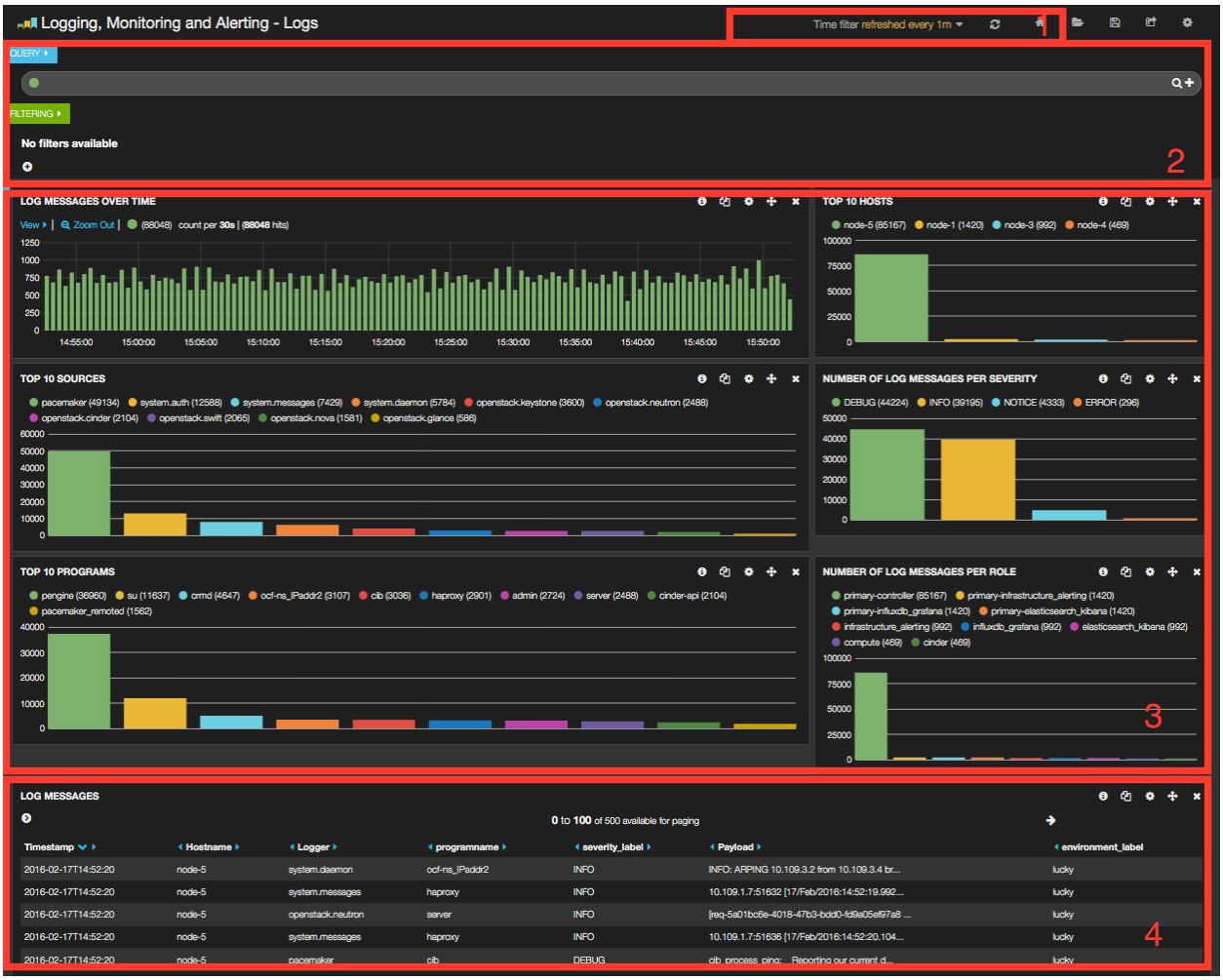
You can switch from one dashboard to another by clicking on the top-right *Load* icon in the toolbar to select the requested dashboard from the list, as shown below.



Each dashboard provides a single pane of glass for visualizing and searching all the logs and notifications of your OpenStack environment. Note that in the LMA Collector settings, it is possible to tag the logs by environment name

so that you can distinguish which logs (and notifications) belong to which environment.

As you can see, the Kibana dashboard for logs is divided into four main sections:



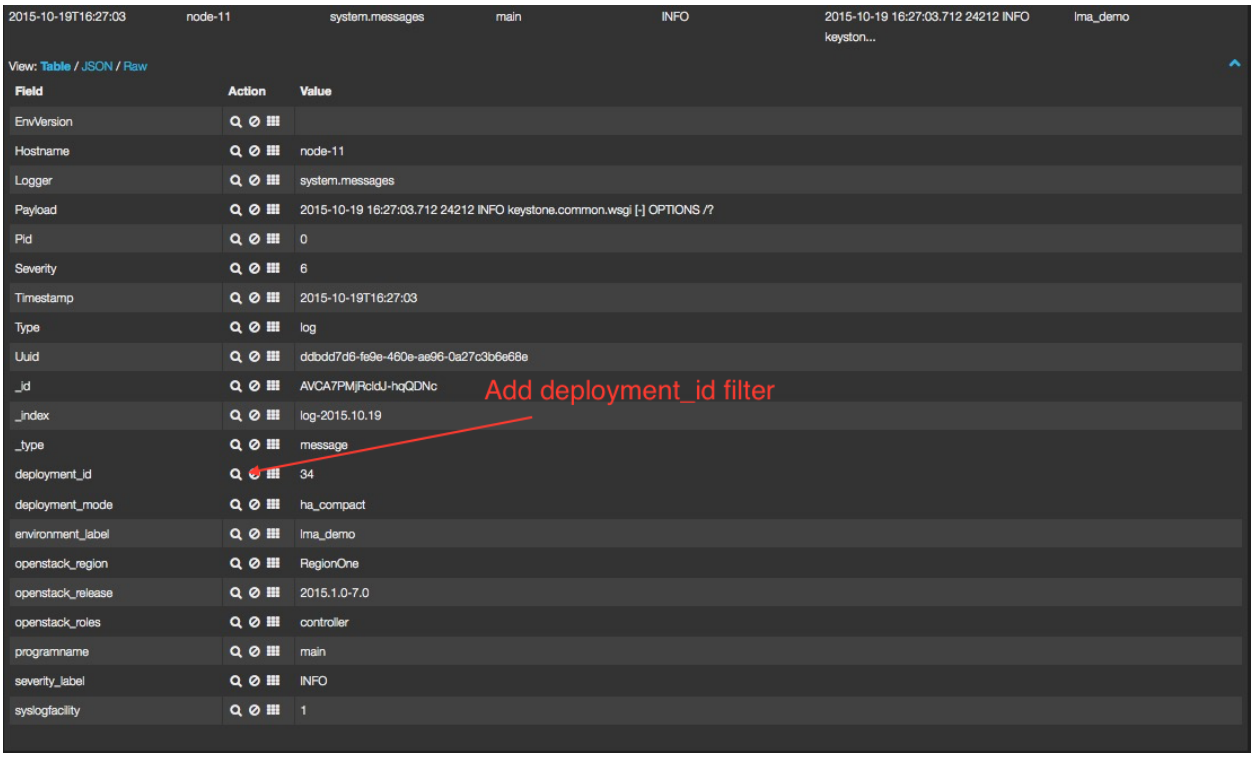
1. A time-picker control that lets you choose the time period you want to select and refresh frequency.
2. A query and filter section where all the filters are displayed.
3. A log analytics row which contains six panels to visualize:
1. The number of log messages for the chosen time period.
2. The top 10 hosts filter.
3. The top 10 log sources.
4. The number of log messages grouped by severity.
5. The top 10 programs.
6. The number of log messages grouped by role.
4. A table of log messages sorted in reverse chronological order.

1.4.4 Filters and queries

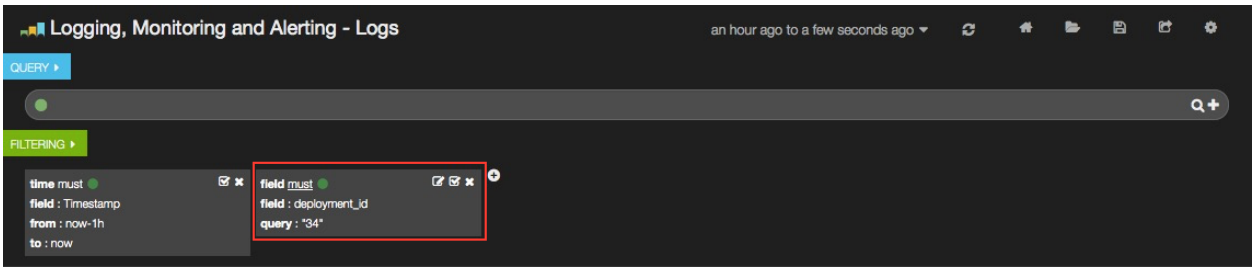
Filters and queries have similar syntax but they are used for different purposes.

- The filters are used to restrict what is displayed in the dashboard.
- The queries are used for free-text search.

You can also combine multiple queries and compare their results. To further filter the log messages to, for example, select the *deployment_id*, you need to expand a log entry and then select the *deployment_id* field by clicking on the magnifying glass icon as shown below.



This will apply a new filter in the dashboard.



Filtering will work for any field that has been indexed for the log entries that are in the dashboard.

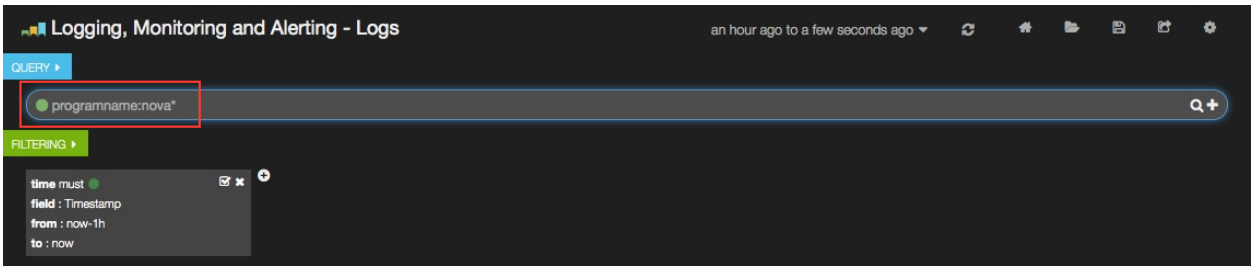
Filters and queries can also use wildcards which can be combined with *field names* like in:

```
programname: <name>*
```

For example, to display only the Nova logs you could enter:

```
programname:nova*
```

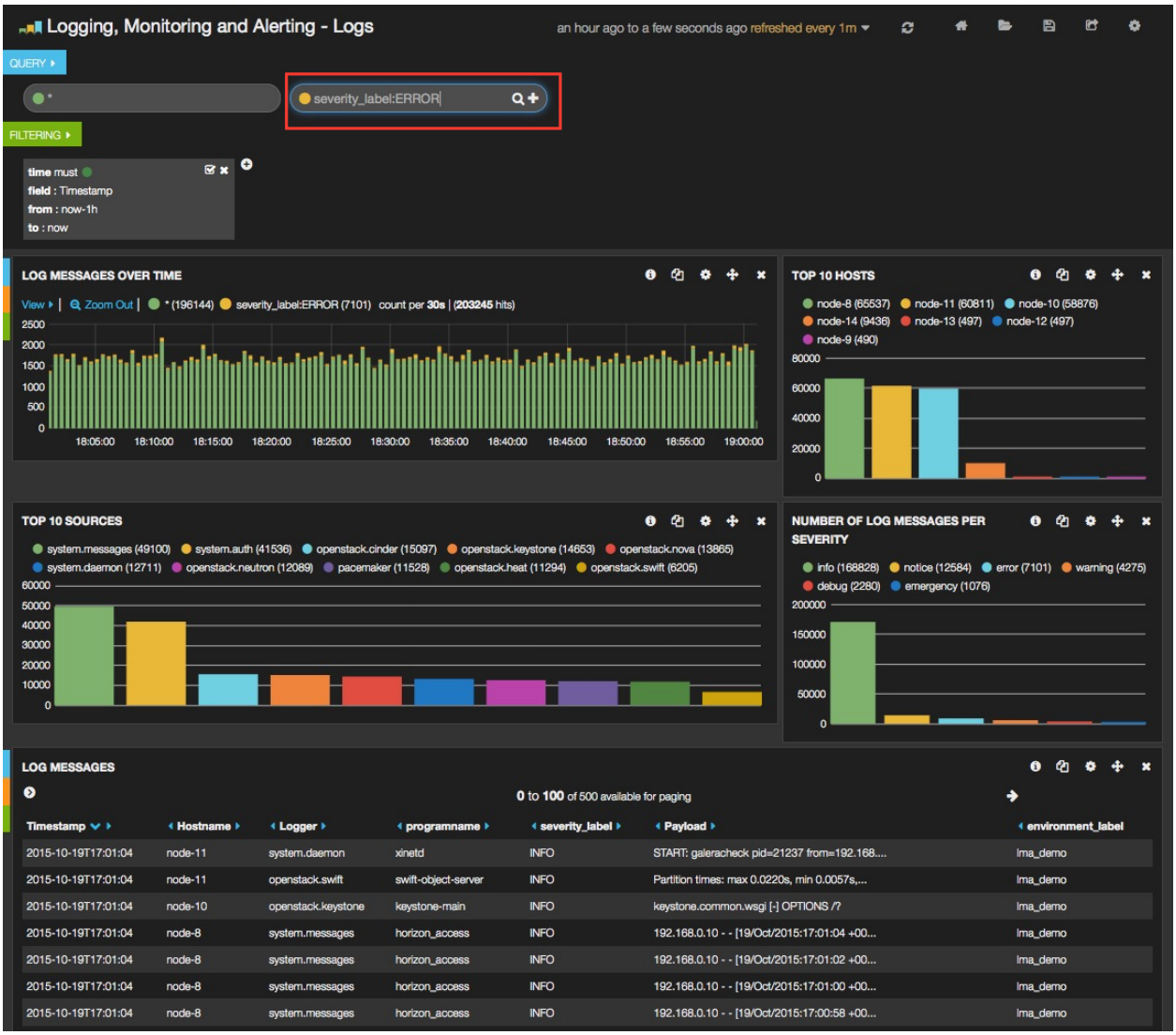
in the query textbox as shown below.



You can also specify multiple queries to compare different data sets.

To add a new query, click on the **+** sign at the right-end of the query textbox and enter a new search query.

The resulting filtering should appear comparing those logs that are in *ERROR* versus those that are not as shown below.



1.4.5 Troubleshooting

If you cannot access the Kibana dashboard or you get no data in the dashboard, follow these troubleshooting tips.

1. First, check that the LMA Collector is running properly by following the LMA Collector troubleshooting instructions in the LMA Collector Fuel Plugin User Guide.
2. Check that the nodes are able to connect to the Elasticsearch cluster via the VIP address on port *9200* as explained in the *Verifying Elasticsearch* section above.
3. On anyone of the *Elasticsearch_Kibana* role nodes, check the status of the VIP address and HAProxy resources in the Pacemaker cluster:

```
root@node-1:~# crm resource status vip__es_vip_mgmt
resource vip__es_vip_mgmt is running on: node-1.test.domain.local

root@node-1:~# crm resource status p_haproxy
resource p_haproxy is running on: node-1.test.domain.local
```

4. If the VIP or HAProxy resources are down, restart them:

```
root@node-1:~# crm resource start vip__es_vip_mgmt
root@node-1:~# crm resource start p_haproxy
```

5. Check that the Elasticsearch server is up and running:

```
# On both CentOS and Ubuntu
[root@node-1 ~]# /etc/init.d/elasticsearch-es-01 status
```

6. If Elasticsearch is down, restart it:

```
# On both CentOS and Ubuntu
[root@node-1 ~]# /etc/init.d/elasticsearch-es-01 start
```

7. Check if nginx is up and running:

```
# On both CentOS and Ubuntu
[root@node-1 ~]# /etc/init.d/nginx status
```

8. If nginx is down, restart it:

```
# On both CentOS and Ubuntu
[root@node-1 ~]# /etc/init.d/nginx start
```

9. Look for errors in the Elasticsearch log files (located at /var/log/elasticsearch/es-01/).
10. Look for errors in the nginx log files (located at /var/log/nginx/).

1.5 Cluster Operations

Because of certain limitations in the current implementation of the Fuel plugin, it is necessary to perform some manual operations after the Elasticsearch cluster is scaled up or scaled down. Those operations are needed to adjust the replication factor of the Elasticsearch indices, based on the new number of nodes in the cluster. There are three types of indices used by the plugin:

- The log indices named *log-%{+YYYY.MM.dd}* which are created on a daily basis.
- The notification indices named *notification-%{+YYYY.MM.dd}* which are also created on a daily basis.
- The Kibana index named *kibana-int* which is created once at installation time to store the templates of the Kibana dashboards.

Adjusting the replication factor for the *kibana-int* index is performed automatically by the plugin and therefore there is no need to do any manual operation for that index when the cluster is scaled up or down.

This is not the case for the replication factor of the other two indices which needs to be updated manualy as described in the official documentation.

The following sections provide more details, describing what do when scaling up/down the Elasticsearch cluster. Scaling up from one node to three nodes, and scaling down from three nodes to one node, are used as examples. Your mileage may vary but the principal of (re)configuring the replication factor of the indices should remain the same.

1.5.1 Scaling Up

The problem the manual operation aims to address is that the replication factor for the old indices is not updated automatically by the plugin when a new node is added in the cluster. If you want the old indices to be replicated on the new node(s), you need to adjust the *number_of_replicas* parameter to the current size of the cluster for those indices as shown below.

The output below shows that the replication factor of the indices created before the scale-up is zero. Here, a scale-up was performed on the 3rd of February, so the indices created after that date (*log-2016.02.04* here) are automatically updated with the correct number of replicas (number of cluster nodes - 1).

> [root@node-1 ~]# curl <VIP>:9200/_cat/indices?v health status index pri rep docs.count docs.deleted store.size pri.store.size green open log-2016.02.03 5 0 270405 0 48.7mb 48.7mb green open log-2016.02.04 5 2 1934581 0 1gb 384.6mb

Then, if you want the *log-2016.02.03* index to be replicated, you need to update the *number_of_replicas* parameter of that index as shown below:

```
 [root@node-1 ~]# curl -XPUT  <VIP>:9200/log-2016.02.03/_settings -d ....
 .....' { "index": { "number_of_replicas": 2 } }'
....... {"acknowledged":true}

 [root@node-1 ~]# curl <VIP>:9200/_cat/indices?v
 health status index                   pri rep docs.count docs.deleted
 green  open   log-2016.02.03            5   2     270405            0
 green  open   log-2016.02.04            5   2    1934581            0

 .... store.size pri.store.size
 ....    146.3mb         48.7mb
 ....        1gb        384.6mb
```

Note that replicating the old indices on the new node(s) will increase the load on the cluster as well as the size required to store the data.

1.5.2 Scaling down

Similarly, after a scale-down the *number_of_replicas* of all indices must be aligned with the new size of the cluster. Not doing so will be reported by LMA as a critical status for the Elasticsearch cluster:

```
[root@node-1 ~]# # the current index health is 'red' after the scale-down
[root@node-1 ~]# curl <VIP>:9200/_cat/indices?v
health  status index                   .....
red     open   log-2016.02.04          .....

..... pri rep docs.count docs.deleted store.size pri.store.size
..... 5   2    1934581            0        1gb        384.6mb
```

```
[root@node-1 ~]# curl -XPUT  <VIP>:9200/log-2016.02.04/_settings -d .....
..... '{ "index": { "number_of_replicas": 0 } }'
{"acknowledged":true}

[root@node-1 ~]# # the cluster health is now 'green'
[root@node-1 ~]# curl <VIP>:9200/_cat/indices?v
health  status index                  pri rep docs.count docs.deleted .....
green     open   log-2016.02.04           5   0    1934581            .....

....  store.size pri.store.size
.... 0    384.6mb        384.6mb
```

1.6 Licenses

1.6.1 Third Party Components

Name	Project Web Site	License
Elasticsearch	https://www.elastic.co/products/elasticsearch	Apache V2
Kibana	https://www.elastic.co/products/kibana	Apache V2

1.6.2 Puppet modules

Name	Project Web Site	License
Elasticsearch	https://forge.puppetlabs.com/elasticsearch/elasticsearch	Apache V2
Concat	https://github.com/puppetlabs/puppetlabs-concat	Apache V2
Stdlib	https://github.com/puppetlabs/puppetlabs-stdlib	Apache V2
Nginx	https://github.com/jfryman/puppet-nginx	MIT license
Firewall	https://github.com/puppetlabs/puppetlabs-firewall	Apache V2
Datacat	https://github.com/richardc/puppet-datacat	Apache V2

1.7 Appendix

- The Elasticsearch-Kibana plugin project at GitHub.
- The official Kibana documentation.
- The official Elasticsearch documentation.

CHAPTER

TWO

INDICES AND TABLES

- search