Altair Mistral Quick Start Guide

Version 2025.2.0

Contents

1 Introduction and product overview

This guide will show you how to get started with Altair Mistral, I/O monitoring tool. For more information about Mistral please refer to the demo videos and user manuals. Mistral was originally developed by Ellexus, now an Altair company.

For detail about the Mistral dashboards please see `mistral_dashboards.pdf`.

1.1 Mistral: per-job I/O monitoring across HPC and distributed systems

Mistral is the leading application monitoring tool for HPC and scientific applications. Mistral is lightweight enough to run in production, but flexible enough to ensure that you get the most from on-prem HPC and that you have the information you need to manage hybrid cloud. Mistral is a storage agnostic tool that monitors I/O and CPU, memory and GPU usage, quickly locating rogue jobs, storage bottlenecks and keeping track of what is running on the clusters day-to-day.

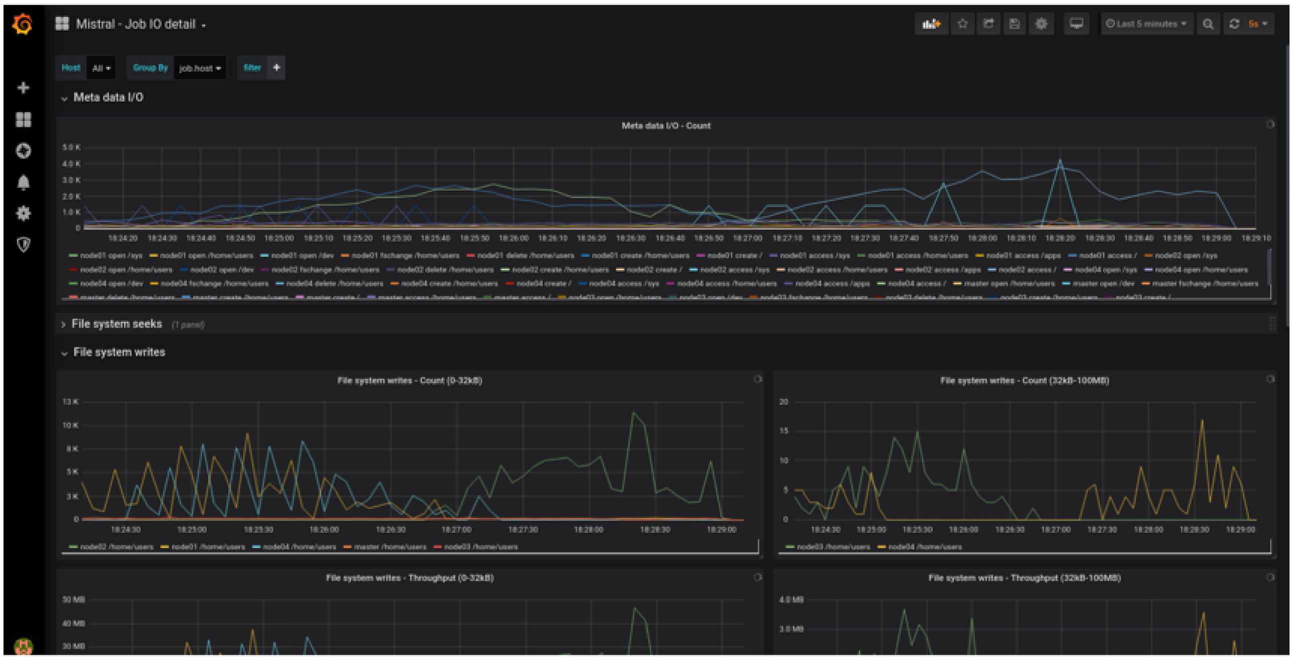

Mistral - Job IO detail
Host
All
Group By
job.host
filter
Meta data I/O
Meta data I/O - Count
File system seeks
File system writes
File system writes - Count (0-32kB)
File system writes - Count (32kB-100MB)
File system writes - Throughput (0-32kB)
File system writes - Throughput (32kB-100MB)
Last 5 minutes
5s

Figure 1: Mistral data in Grafana

1.2 Understanding how applications are using shared storage

As distributed systems and compute clusters become more complex, the need for monitoring becomes more important. To be able to run the compute efficiently and to forecast and design for the future it is important to have visibility on what is being run today.

1.3 Live analytics and historical trends

Mistral logs data in a scalable database such as Elastic Search or InfluxDB so a graphing dashboard such as Grafana or Splunk can be used to view the information. These give you live updates so you can find rogue jobs quickly or look back over historical data.

1.4 Protect shared storage

Protect shared storage from rogue jobs and noisy neighbour applications with per-job I/O monitoring. It is easy for a single user to overload shared storage with too many jobs or with applications that have bad I/O patterns. Mistral shows you which user, host or job is hitting the file system with too many requests.

1.5 Workflow qualification

Catch problems before they happen with workflow qualification at scale. New workloads can be quickly tested with Mistral to find bad I/O patterns before they become a problem.

1.6 Mistral generates a live feed of:

- Data recorded per job or application
- Data recorded for each mountpoint or filesystem
- Reads, writes and meta data
- I/O performance

It's easy to add custom metrics to measure machine health for a full system overview.

2 Installation and getting started

This guide assumes you have access to the user manuals and documentation.

2.1 Download

Go to the Altair One page and register for an account. You can then log in and download Mistral.

We have 3 formats of package, a `.tar.gz`, `.deb` and `.rpm`. Select whichever is easiest for you to work with. Mistral needs to be installed in the same location across all the machines that you wish monitor. This is most easily achieved by installing it to a shared file system, if this is selected then there is no need to install Mistral on each individual host.

2.1.1 Tar file

To install, just untar the product on any filesystem which is accessible from all the hosts that will run workloads.

2.1.2 Debian package

Install using your package manager. This will install Mistral to `/opt/altair/mistral_2025.2.0_x86_64` by default.

e.g.

```
$ sudo dpkg -i mistral_2025.2.0-0_amd64.deb
```

To install to a different location use the `--instdir=dir` argument to `dpkg`.

2.1.3 RPM package

Install using your package manager. This will install Mistral to `/opt/altair/mistral_2025.2.0_x86_64` by default.

e.g.

```
$ sudo rpm --install mistral-2025.2.0-0.x86_64.rpm
```

To install in a different location use the `--prefix=<path>` argument to `rpm`.

2.2 Database and Dashboard

Mistral can be run on a single application, but to visualize data for multiple applications you will need to set up a database, to collect the data from each application, and a dashboard, to present visualisations of the data in the database. Mistral can push data to various databases such as Elasticsearch and Splunk. For customers who do not already have a log-aggregation infrastructure in place we recommend starting with Elasticsearch, and the Grafana dashboard.

The supported Mistral plug-ins that can forward data to databases are in the `plugins` directory alongside the documentation for them. Example start-up scripts can be found in the `docs/samples` directory.

2.3 Test run Mistral

We recommend that you initially run Mistral against a small program, logging the data to a local file on a disk. It is a good idea to put the required settings in a launch script, based on the provided `docs/samples/mistral_start_reference` Here we assume you have done that, and your launch script is called `mistral_start.sh`.

Now run `ls` under Mistral to test the settings:

```
$ bash
$ source mistral_start.sh
$ /bin/ls
$ # Echo out the Mistral log file full path
$ mistral_log_with_hostname $MISTRAL_LOG
$ exit
$ cat <log file>
```

Check that output log file has data and that there are no Mistral errors in `syslog`. This will ensure that you have the files all in the correct location, the license is working, and there are sufficient permissions.

Mistral can produce a very high volume of log output. This is useful for testing, but in production use across a cluster you may prefer to change the Mistral configuration file to reduce log volume, for instance by increasing the “timeframe”. See the Mistral Manual for more detail on configuration options.

2.4 Run Mistral Healthcheck (Optional)

In order to quickly see the data that Mistral can collect you could now run Mistral with a more representative application, to create a Mistral log file. Run your application and generate a Mistral Healthcheck report using the following commands:

```
$ bash
$ source mistral_start.sh
$ <myapp arg0 arg1 ...>
$ # Echo out the Mistral log file full path
$ mistral_log_with_hostname $MISTRAL_LOG
$ exit
$ <mistral path>/bin/mistral_report.sh -o <path to output dir> <log file>
```

This will create a stand-alone HTML report of the I/O from your application.

2.5 Database support

Mistral has several output plug-ins for sending data to different databases. We recommend using Elasticsearch, but we also support Splunk, InfluxDB and FluentBit (a log forwarder).

2.6 Install Elasticsearch and Grafana

Install Elasticsearch and Grafana and set-up the Mistral Elasticsearch plug-in to push data to your Elasticsearch database. For this version of Mistral, we recommend at least Elasticsearch 8.10.3 and Grafana 10.0.4, as these are the respective versions that the given samples were written and tested against.

(For installation tips, see the separate section below).

Add a `plugin:` section to your `mistral_config.yaml` file, either in the Mistral `etc` directory or elsewhere on your system (for example, in `/etc` with other configuration files). We recommend you base this section on the `docs/samples/elasticsearch/plugin.conf` file provided, with the correct plugin path, hostname/IP address, database name, username (if needed - remove these lines if authentication is not used), error log path, executable path and password.

Once this is done you can now run the same test as above, but with an additional variable defined:

```
#!/bin/sh

## Mistral launch script

export ALTAIR_LICENSE_PATH=<path to license file>
export MISTRAL_LOG=<path to output log file>
export MISTRAL_CONFIG=<path>/mistral_config.yaml
```

Now run `ls` under Mistral again to test the settings

```
$ bash
$ source ./mistral_start.sh
$ /bin/ls
$ # Echo out the Mistral log file full path
$ mistral_log_with_hostname $MISTRAL_LOG
$ exit
$ cat <log file>
```

There should be no Mistral errors reported in `syslog` and this time there should be no new lines in the Mistral log because the data should now be going to the database. Mistral will only fall back to writing to the log file if there is a problem passing data to the database.

You should now check if data has been added to the Elasticsearch database. This can be done with either a quick `curl` command or Kibana (if you have it installed). The web interface for Kibana can be accessed by pointing your browser at `<server>:5601`. Once you have connected to Kibana you can search for Mistral data under the discover option and searching for `Mistral*` within the correct time frame. Each log entry should be visible in the dashboard.

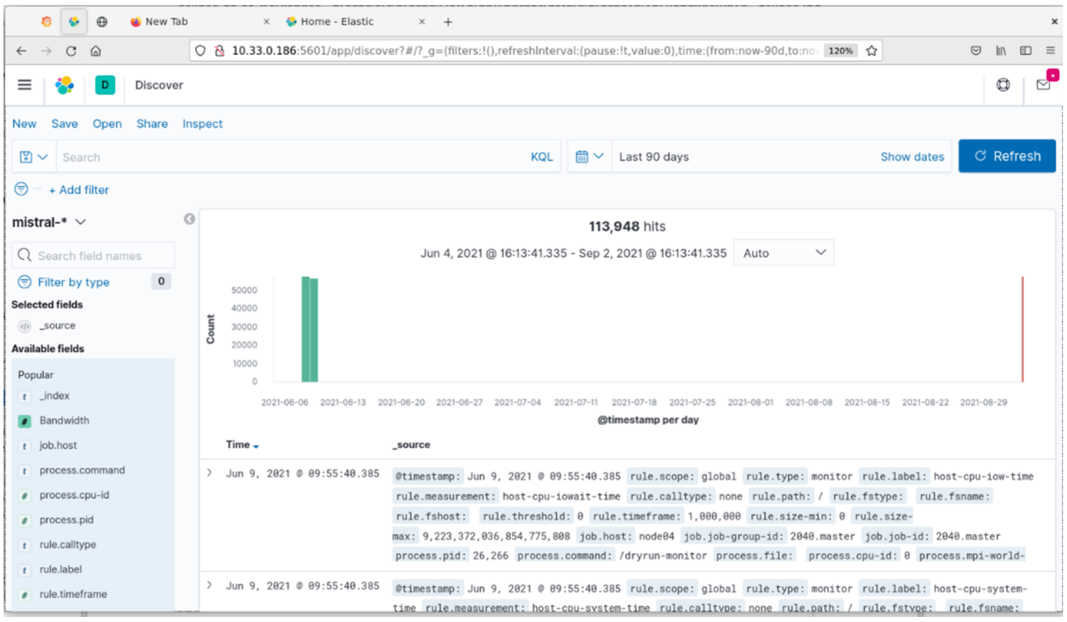
2.7 Set-up the Grafana dashboards

Set-up the Grafana dashboards and check that you can see the data added in the previous step.

First set-up the Mistral Elasticsearch database as a data source in Grafana. If possible, name this data source `Mistral` to match the supplied dashboard. See Grafana data source for further details.

Import the following 5 files as new dashboards in Grafana:

- `docs/samples/elasticsearch/grafana/dashboards/grafana_overview.json`
- `docs/samples/elasticsearch/grafana/dashboards/grafana_jobdetail.json`
- `docs/samples/elasticsearch/grafana/dashboards/grafana_performance.json`
- `docs/samples/elasticsearch/grafana/dashboards/grafana_job_summary.json`



113,948 hits

Jun 4, 2021 @ 16:13:41.335 - Sep 2, 2021 @ 16:13:41.335 Auto

```
Time                      _source
Jun 9, 2021 @ 09:55:40.385  @timestamp: Jun 9, 2021 @ 09:55:40.385 rule.scope: global rule.type: monitor rule.label: host-cpu-iow-time
                            rule.measurement: host-cpu-iowait-time rule.calltype: none rule.path: / rule.fstype:  rule.fsname:
                            rule.fshost:  rule.threshold: 0 rule.timeframe: 1,000,000 rule.size-min: 0 rule.size-
                            max: 9,223,372,036,854,775,808 job.host: node04 job.job-group-id: 2040.master job.job-id: 2040.master
                            process.pid: 26,266 process.command: /dryrun-monitor process.file:  process.cpu-id: 0 process.mpi-world-
Jun 9, 2021 @ 09:55:40.385  @timestamp: Jun 9, 2021 @ 09:55:40.385 rule.scope: global rule.type: monitor rule.label: host-cpu-system-
                            time rule.measurement: host-cpu-system-time rule.calltype: none rule.path: / rule.fstype:  rule.fsname:
```

Figure 2: Mistral data in Kibana

- `docs/samples/elasticsearch/grafana/dashboards/grafana_network.json`

This can be done by going to http://grafana url/dashboard/import. See Importing sample grafana dashboards for further details.

Check that the time range at the top right of the dashboards covers when the previous tests were run, and check to see that there is some data visible. Once this is done, you should now be able to run more representative jobs and fine-tune the dashboards to make sure that you only have the information that you are interested in.

The third of these dashboards (`grafana_performance.json`) will not be populated with data by default. If you are interested in collecting this information then you will need to enable duration measurements. This can be achieved by modifying your Mistral configuration file, setting these duration settings:

```
# Mistral configuration items to enable duration metering, so that
# there is data for the performance dashboard.
read:
    duration: yes
write:
    duration: yes
metadata:
    duration: yes
```

2.8 Job scheduler integration

Integrate Mistral with your job scheduler to automatically insert Mistral on specific queues.

See the Mistral User Manual for scheduler-specific integration instructions, and the `docs/samples` directory for integration scripts.

3 Installing Elasticsearch and Grafana

For an evaluation Elasticsearch and Grafana can easily be installed on a single VM, but when planning to go to production you should take care to estimate the likely volume of Mistral data, and provision and configure both Elasticsearch and Grafana appropriately. The open source community versions of these tools are absolutely fine for the evaluation: we don't use any enterprise-specific functionality.

3.1 Installing with Docker Compose

The easiest way to install Elasticsearch with Kibana and Grafana is through the provided `mistral-elasticsearch-container-setup.sh` script in `sbin`. This script is a wrapper for `docker compose` so you must have this installed to setup in this way. It has a single required argument specifying where the Mistral license file is:

```
./mistral-elasticsearch-container-setup.sh [-d] [-p]
           [-o /path/to/destination/for/elasticsearch_certifcate]
           [-e /path/to/mistral_elasticsearch_directory]
           [-h [<hostname>]]
            -l /path/to/mistral_license
```

Please check the usage message for all documented options and arguments. Before running this script, the `etc/elasticsearch/.env` file provided should be filled in with appropriate default values. Each option is documented as a comment above it in the file. When each option has been set, simply run the shell script. The script must be run in the directory it is provided in, and all provided files should also be present as-is.

Regarding system resources for the containers, memory usage of Elasticsearch can be high, so we highly recommend ensuring that the `etc/elasticsearch/.env` is modified according to your system needs. We have provided a default of 4GB to the Elasticsearch container (and 1GB for Kibana, which is less intensive) as a starting point for evaluation. Elasticsearch itself will automatically assign to it's JVM what it thinks is enough memory. If you wish to fine-tune this further, please consult the official documentation for JVM heap size. Please consider that this is a single node installation and may not be appropriate for large production-sized clusters.

Note that you may need to change the `vm.max_map_count` kernel setting to run the elasticsearch docker image. See the offical documentation. for more details.

```
sysctl -w vm.max_map_count=262144
```

To permanently change the value for the `vm.max_map_count` setting, update the value in `/etc/sysctl.conf`.

Should any of the container setup phases fail then you can see the output by switching to the `etc/elasticsearch` directory and using `docker compose logs` or `docker compose up`.

This install process will attempt to download and install Third Party content. Altair is not responsible for content developed and hosted by Third Parties. ALTAIR DISCLAIM ANY AND ALL LIABILITY, INCLUDING ANY EXPRESS OR IMPLIED WARRANTIES, WHETHER ORAL OR WRITTEN, FOR SUCH THIRD-PARTY CONTENT. IN USING THIS SERVICE TO DOWNLOAD AND INSTALL THIS THIRD-PARTY CONTENT YOU ACKNOWLEDGE THAT NO REPRESENTATION HAS BEEN MADE BY US AS TO THE FITNESS OF THE THIRD-PARTY SERVICES FOR YOUR INTENDED PURPOSE.

3.2 Installing with Podman Compose

The `mistral-elasticsearch-container-setup.sh` script includes a `-c` option to utilize `podman` and `podman-compose` for installing Elasticsearch, Kibana and Grafana. This script has been tested on `Ubuntu 20.04.5` and `Rocky Linux 8.9` with `-p` option as shown in the command below.

```
./mistral-elasticsearch-container-setup.sh -c podman -l <ALTAIR_LICENSE_PATH> \
  -e <MISTRAL_ELASTICSEARCH_PLUGIN_PATH>
```

There is a possibility of error based on the system configuration with `podman` and `podman-compose`, which can be resolved with the following recommendations.

3.2.1 Possible error with subnet allocation

If there is an error with subnet, then remove `mistral_elastic` network and create manually with available private subnet as follow and run the `mistral-elasticsearch-container-setup.sh` script once again.

```
ERRO[0000] "plugin type="bridge" failed (add): cni plugin bridge failed:
range set 0 overlaps with 1"
```

```
sudo podman network rm mistral_elastic
sudo podman network create --subnet X.X.X.X/16 mistral_elastic
```

3.2.2 Possible error with resource limits

If there is an error with resource limits, then update `/etc/security/limits.conf` and restart the system. The error happened in `Ubuntu 20.04.5` by default.

```
ERROR: Cannot start service elasticsearch: OCI runtime create failed:
setting rlimits for ready process caused: error setting rlimit type 8:
operation not permitted: unknown
```

```
$ cat <<EOF | sudo tee -a /etc/security/limits.conf
*                 soft    nofile          1024000
*                 hard    nofile          1024000
*                 soft    memlock         unlimited
*                 hard    memlock         unlimited
elastic           soft    nofile          1024000
elastic           hard    nofile          1024000
elastic           soft    memlock         unlimited
elastic           hard    memlock         unlimited
root              soft    nofile          1024000
root              hard    nofile          1024000
root              soft    memlock         unlimited
EOF

$ sudo reboot
```

3.2.3 Possible error with memory map area

If there is an error with memory map area, then update `/etc/sysctl.conf` and apply the configuration in system. The parameters `net.ipv4.ip_forward`, `net.ipv4.tcp_retries2` and `vm.swappiness` are not directly related to `vm.max_map_count`, they are often configured together to optimize system performance. The error happened in `Rocky Linux 8.9` by default.

```
ERROR: bootstrap check failure [1] of [1]: max virtual memory areas
vm.max_map_count [65530] is too low, increase to at least [262144]
```

```
$ cat <<EOF | sudo tee -a /etc/sysctl.conf
vm.max_map_count=262144
net.ipv4.ip_forward=1
net.ipv4.tcp_retries2=5
vm.swappiness=1
EOF

$ sudo sysctl -p
$ sudo systemctl restart network
```

3.2.4 Possible error with Healthcheck

If there is an error with healthcheck as shown below, then remove healthcheck sections from the `docker-compose.yml` file for all the services. After making these changes, run the `mistral-elasticsearch-container-setup.sh` script again. The `mistral-elasticsearch-container-setup.sh` script manages the healthcheck and container synchronization to ensure proper operation. The error happened in `Rocky Linux 8.9` by default.

```
Traceback (most recent call last):
  File "/bin/podman-compose", line 8, in <module>
    sys.exit(main())
  File "/usr/local/lib/python3.8/site-packages/podman_compose.py", line 3504, in
   ↪  main
    asyncio.run(async_main())
  File "/usr/lib64/python3.8/asyncio/runners.py", line 44, in run
    return loop.run_until_complete(main)
  File "/usr/lib64/python3.8/asyncio/base_events.py", line 616, in
   ↪  run_until_complete
    return future.result()
  File "/usr/local/lib/python3.8/site-packages/podman_compose.py", line 3500, in
   ↪  async_main
    await podman_compose.run()
  File "/usr/local/lib/python3.8/site-packages/podman_compose.py", line 1743, in
   ↪  run
    retcode = await cmd(self, args)
  File "/usr/local/lib/python3.8/site-packages/podman_compose.py", line 2500, in
   ↪  compose_up
    podman_args = await container_to_args(compose, cnt, detached=args.detach)
  File "/usr/local/lib/python3.8/site-packages/podman_compose.py", line 1205, in
   ↪  container_to_args
```

```
    raise ValueError("'CMD_SHELL' takes a single string after it")
ValueError: 'CMD_SHELL' takes a single string after it
Failed to start docker containers with error code 1.
```

3.2.5 Possible error with UIDs or GIDs

If there is an issue with UIDs or GIDs in `Rocky Linux 8.9`, then run the `mistral-elasticsearch-container-setup.sh` script with `sudo` permission or migrate podman storage to ensure it executes correctly.

```
Error: copying system image from manifest list: writing blob: adding layer ...
processing tar file(potentially insufficient UIDs or GIDs available in user
namespace (requested 0:42 for /etc/shadow): Check /etc/subuid and /etc/subgid
if configured locally and run "podman system migrate":
lchown /etc/shadow: invalid argument): exit status 1
```

```
sudo ./mistral-elasticsearch-container-setup.sh -c podman -l <ALTAIR_LICENSE_PATH>
 ↪  \
      -e <MISTRAL_PLUGIN_PATH>
```

```
podman system migrate
./mistral-elasticsearch-container-setup.sh -c podman -l <ALTAIR_LICENSE_PATH> \
  -e <MISTRAL_PLUGIN_PATH>
```

3.3 Manual Installation

We recommend installing both Elasticsearch (database) and Kibana (database web front-end). You can do this from the repositories:

https://www.elastic.co/guide/en/elasticsearch/reference/current/rpm.html#rpm-repo

https://www.elastic.co/guide/en/kibana/current/rpm.html#rpm-repo

3.3.1 Elasticsearch configuration

During an evaluation the following configuration changes allow a single node installation to be accessed without authentication from other machines on the network:

`/etc/elasticsearch/elasticsearch.yml`

```
# Identify this host
node.name: <hostname>
# Store log files in folder in /var/log
path.logs: /var/log/elasticsearch
# Listen on localhost and other network addresses
network.host: ["localhost", "127.0.0.1", "<IP address>", "<hostname>"]
# Mark this host as a master, no discovery required - works for single nodes
cluster.initial_master_nodes: ["<hostname>"]
# Allow larger queries from Grafana
search.max_buckets: 200000
```

```
# Allow more data per node
cluster.max_shards_per_node: 10000
# Mark this host as the only one to query for discovery
discovery.seed_hosts: ["<hostname>"]
```

Should you wish to install user security, you can do this using the xpack.security options. The Mistral output plug-in supports this.

You may need to allow ElasticSearch through the firewall. On RHEL/CentOS 7 the following commands should do this:

```
$ firewall-cmd --zone=public --permanent --add-port=9200/tcp
$ firewall-cmd --reload
```

3.3.2 Elasticsearch index template

Once Elasticsearch is installed you will need to create an index template. Our plug-in creates a new Elasticsearch instance each day, of the form

`<index name (normally mistral)>-YYYY-MM-DD`

To do this, the plug-in will automatically create the index templates when it starts up. However, if you wish to create a different kind of index, we provide a script called `schema/mistral_create_elastic_template.sh` - please run this script before trying to send data from Mistral if you do not wish to have date-based indices. The `-?` argument provides details of how to run the script.

3.3.3 Kibana configuration

The Kibana installation may require the following changes to be externally accessible:

`/etc/kibana/kibana.yml`

```
# Listen on all network connections
server.host: "0.0.0.0"
# Server name - display purposes only
server.name: "<hostname>"
```

You may need to allow Kibana through the firewall. On RHEL/CentOS 7 the following commands should do this:

```
$ firewall-cmd --zone=public --permanent --add-port=5601/tcp
$ firewall-cmd --reload
```

3.3.4 Grafana Install

Grafana (data visualization) can also be installed from the repositories:

https://grafana.com/docs/grafana/latest/installation/rpm/

You shouldn't need to make any configuration changes to the defaults provided.

You should be able to connect to Grafana web interface on port 3000 e.g. http://hostname:3000

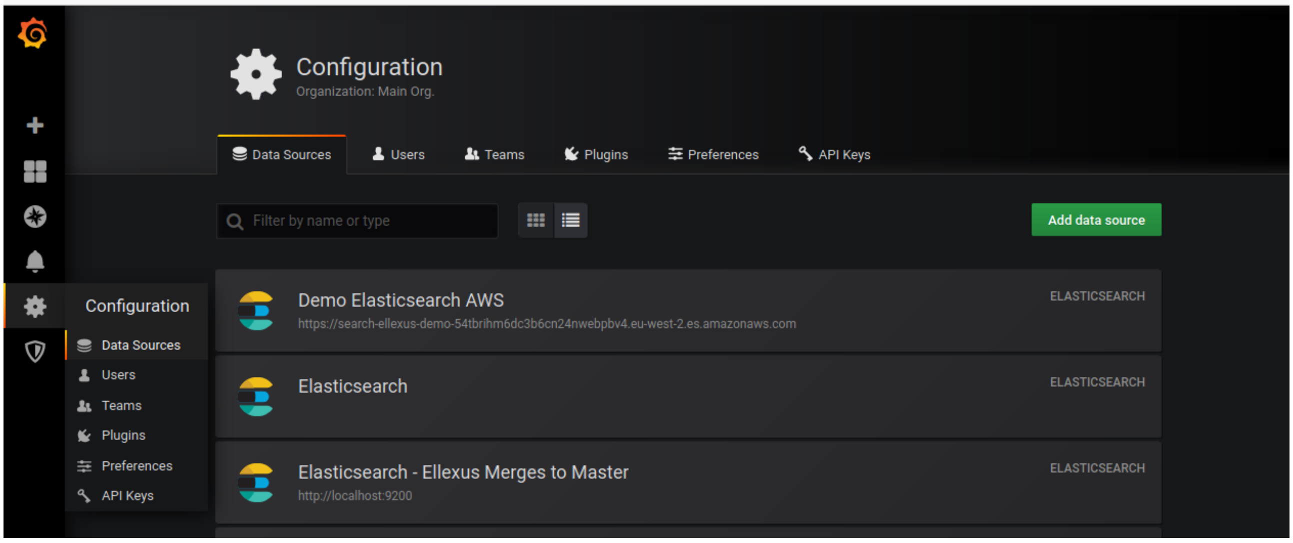
The default username is admin and the password is also admin. When first run the web U/I will ask you to change this.

You may need to allow Grafana through the firewall. On RHEL/CentOS 7 the following commands should do this:

```
$ firewall-cmd --zone=public --permanent --add-port=3000/tcp
$ firewall-cmd --reload
```

3.3.4.1 Grafana data source

Add a data source to Grafana by hovering over the settings cog on the left, followed by clicking on `Data Sources`. On the subsequent page you can then click `Add data source`.



Demo Elasticsearch AWS — ELASTICSEARCH
https://search-ellexus-demo-54tbrihm6dc3b6cn24nwebpbv4.eu-west-2.es.amazonaws.com

Elasticsearch — ELASTICSEARCH

Elasticsearch - Ellexus Merges to Master — ELASTICSEARCH
http://localhost:9200

Figure 3: Grafana data sources menu

On the next page select Elasticsearch as the data source type.

Enter the following into the new data source:

```
Name: Mistral
HTTP:
URL: http://<hostname>:9200
Access: Server (Default)
Auth:
# Leave the default settings, unless you have installed Elasticsearch
# authentication
Elasticsearch details:
Index name: [mistral-]YYYY-MM-DD
Pattern: Daily
Time field name: timestamp
Version: 8.0+
Max concurrent Shard Requests: 256
Min Time Interval: 20s
```

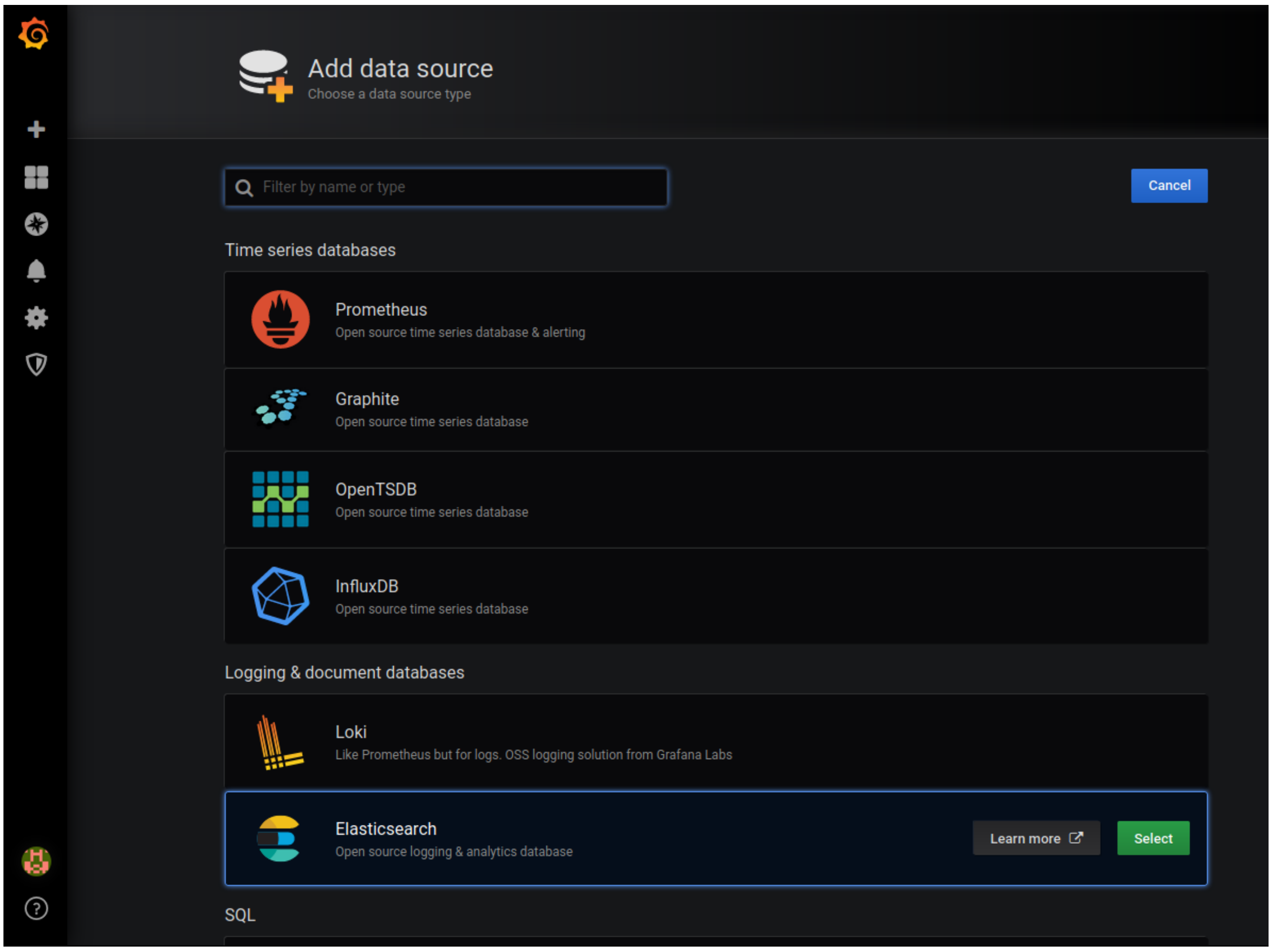
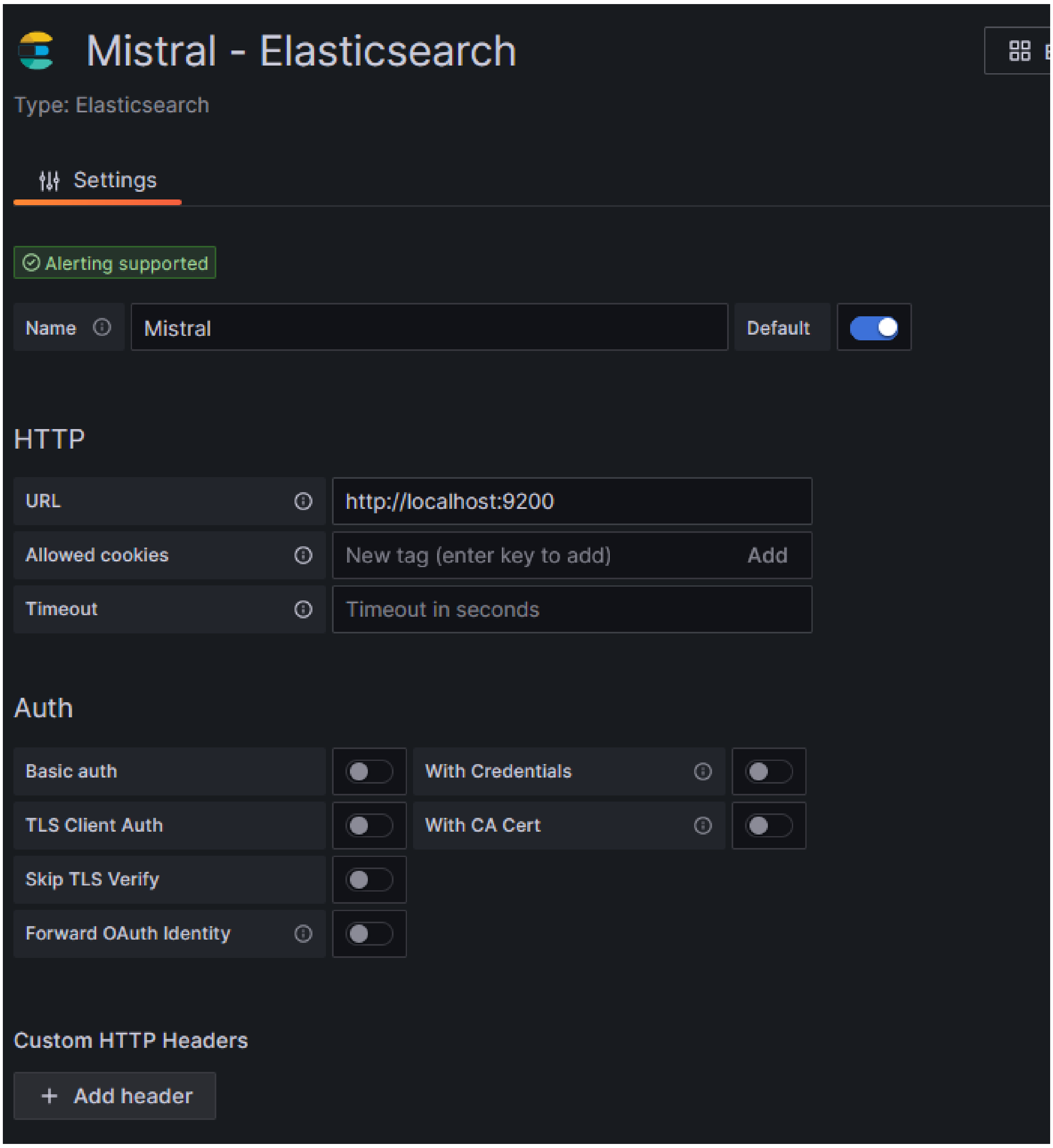

Add data source
Choose a data source type
Filter by name or type
Cancel
Time series databases
Prometheus
Open source time series database & alerting
Graphite
Open source time series database
OpenTSDB
Open source time series database
InfluxDB
Open source time series database
Logging & document databases
Loki
Like Prometheus but for logs. OSS logging solution from Grafana Labs
Elasticsearch
Open source logging & analytics database
Learn more
Select
SQL

Figure 4: Selecting Elasticsearch as a Grafana Data Source



Mistral - Elasticsearch

Type: Elasticsearch

Settings

Alerting supported

Name	Mistral	Default	on

HTTP

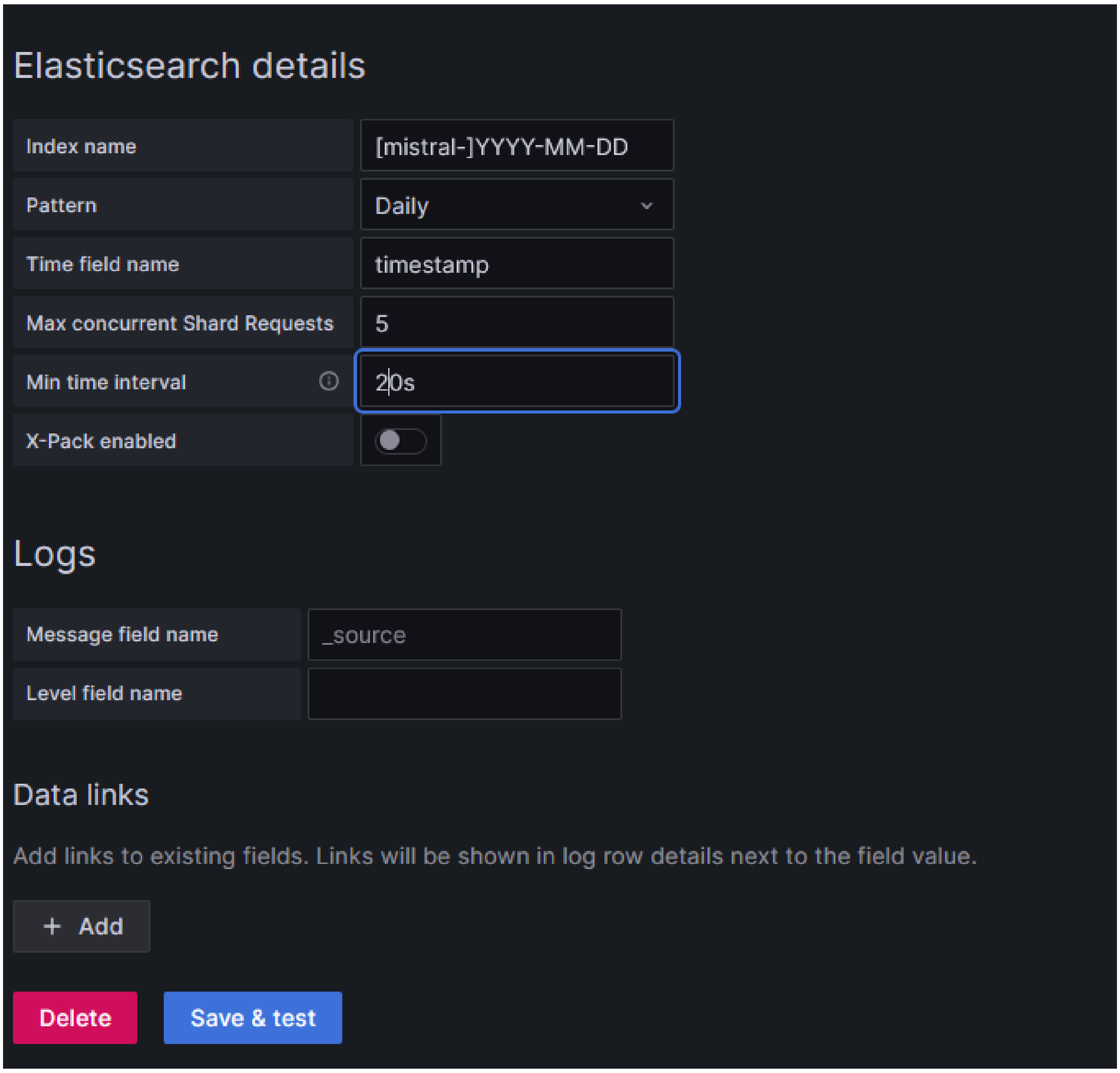
URL	http://localhost:9200
Allowed cookies	New tag (enter key to add) Add
Timeout	Timeout in seconds

Auth

Basic auth	off	With Credentials	off
TLS Client Auth	off	With CA Cert	off
Skip TLS Verify	off		
Forward OAuth Identity	off		

Custom HTTP Headers

Figure 5: Grafana Elasticsearch data source 1



Elasticsearch details

Index name	[mistral-]YYYY-MM-DD
Pattern	Daily
Time field name	timestamp
Max concurrent Shard Requests	5
Min time interval	20s
X-Pack enabled	

Logs

Message field name	_source
Level field name	

Data links

Add links to existing fields. Links will be shown in log row details next to the field value.

Figure 6: Grafana Elasticsearch data source 2

It is important that the `Min Time Interval` setting matches the timeframe in the `etc/mistral_config.yaml` file. If it is of a lower value graphs may look choppy.

All other settings can be left as their defaults, click `Save & Test`. If there are no errors, then this data source is setup.

You will now need to repeat the same steps to add 4 more data sources, with the following names and index patterns:

Data Source Name	Index name
Network - Mistral	[network-mistral-]YYYY-MM-DD
Summary - Mistral	[summary-mistral-]YYYY-MM-DD
Total - Mistral	[total-mistral-]YYYY-MM-DD
Total - Network	[total-network-mistral-]YYYY-MM-DD

Once all of these are created and you see no errors, you should be ready to import the sample dashboards.

3.3.5 Importing sample Grafana dashboards

Import the dashboards by hovering over the dashboards menu item (4 squares) and clicking `Manage`. On the subsequent page click `Import`.

You should then click `Upload .json file`, which will open a file browser. From there you can import the `elasticsearch/grafana/dashboards/grafana_overview.json` from the Mistral `docs/samples` directory.

Repeat for the `elasticsearch/grafana/dashboards/grafana_jobdetail.json`, `elastic-search/grafana/dashboards/grafana_performance.json`, `elasticsearch/grafana/dashboards/grafana_` and `elasticsearch/grafana/dashboards/grafana_network.json` dashboards.

4 Troubleshooting: There is no data in my Grafana dashboard

Here we are going to trouble shoot the problem from the top down, checking the Mistral configuration and finishing with the dashboard. If you think the dashboard is likely to be a problem then you may want to complete these steps in reverse order or skip to the middle.

4.1 Ensuring everything matches

Mistral comes with sample dashboards and launch scripts. These have been written to work together. If you are upgrading to a new version of Mistral please ensure that you have set up Mistral to work with the supplied plugins, dashboards and sample configuration and launch scripts. Once you have done that and Mistral is working you can then apply any changes you would like to make to the configuration to suit your environments.

4.2 Running Mistral on one job and logging to disk

If you are upgrading you may have skipped the initial steps at the start of this guide to run a simple job and to log the data to disk. Please return to the beginning of the guide and ensure that you can run a simple job without a data pipeline.

4.3 Checking the scheduler integration

Once you have verified that Mistral can run correctly and is logging to disk, you can test it by running it using the scheduler integration if you have that set up. Mistral can be configured to run automatically on specific jobs or queues by installing the templates in the samples dir as job hooks or starter methods.

- Make sure your starter method or job hook is enabled.
- Make sure that the settings in the starter method or launch script match the launch script that you just tested.
- Keep the plugin config setting disabled for now
- Submit a job to the relevant queue; something simple like `ls` will do. `sleep 0` will not, as it doesn't do I/O.
- Make sure that the program you are using for testing can run and does some I/O
- Once the job has completed you should check the output log and error log again. If you have data in the output log then Mistral is working correctly with your scheduler and it is now time to start pushing data to your data pipeline.

4.4 Checking the database and database plugin

Now you have verified that Mistral is working correctly and logging data to disk, re-enable the plugin configuration by un-commenting that section in your launch configuration, double-checking that the settings are correct in the plugin configuration file. Mistral will now use the plugin to push data to the database.

You should also check that you can ping the database server from your job machines to make sure that there are not going to be firewall or DNS issues. If all this is OK then run another short job.

```
$ qsub -q mistral ls
$ cat <mistral log>
$ cat <mistral error log>
$ cat <plugin error log>
```

The plugin error log might be the same as the mistral error log or it might be a separate file defined in the Mistral config file (under `plugin…options…error`). If Mistral is pushing data to the database the mistral log file should be missing and the error log files should be empty. If so, you should follow the steps above to check the database.

4.5 Checking the dashboard

Finally, we have verified that Mistral is working correctly and pushing data to the database. If you are still not seeing data in the dashboard then check the following things:

- Are you using the right dashboard? The dashboard needs to match the Mistral version. If you have changed versions, that can prevent data from being picked up.
- Have you upgraded from a previous version of Mistral and copied settings from old configuration files? You should check that your config files match the templates in the `docs/samples` directory.
- If you are using one of the template dashboards, try reimporting the dashboard.
- Once you have imported the dashboard check that you are looking at the right time window for your data and that you have it set to refresh automatically.
- Check that you have the right data source in the dashboard by looking at one of the panels/graphs to see which data source it uses
- Look at that data source to see what index pattern it's searching for and check that it matches the index pattern created by the plugin you are using.

If you have completed all the steps above and still can't see any data, then please contact support on breeze_mistral_support@altair.com