Lab 10. Monitoring Server Infrastructure

In this lab, we will cover the following topics:

- Metricbeat
- Configuring Metricbeat
- Capturing system metrics
- Deployment architecture

Note:

Run metricbeat commands from following directory:

```
su elasticsearch
cd /elasticstack/metricbeat-7.12.1-linux-x86_64
```

Configuring Metricbeat

The configurations related to Metricbeat are stored in a configuration file named `metricbeat.yml`, which uses YAML syntax.

The `metricbeat.yml` file contains the following:

- Module configuration
- General settings
- Output configuration
- Processor configuration
- Path configuration
- Dashboard configuration
- Logging configuration

Let's explore some of these sections.

Module configuration

Metricbeat comes bundled with various modules to collect metrics from the system and applications, such as Apache, MongoDB, Redis, MySQL, and so on.

Metricbeat provides two ways of enabling modules and metricsets as follows:

- Enabling module configs in the `modules.d` directory
- Enabling module configs in the `metricbeat.yml` file

Enabling module configs in the modules.d directory

The `modules.d` directory contains default configurations for all the modules that are available in Metricbeat. The configuration that's specific to a module is stored in a `.yml` file, with the name of the file being the name of the module. For example, the configuration related to the MySQL module will be stored in the `mysql.yml` file. By default, except for the `system` module, all other modules are disabled. To list the modules that are available in Metricbeat, execute the following command:

```
metricbeat modules list
```

The `modules list` command displays all the available modules and also lists which modules are currently enabled/disabled.

Note

If a module is disabled, then in the `modules.d` directory, the configuration related to the module will be stored with the `.disabled` extension.

Since each module comes with default configurations, make the appropriate changes in the module configuration file.

The basic configuration for the `mongodb` module will look as follows:

```
- module: mongodb
  metricsets: ["dbstats", "status"]
  period: 10s
  hosts: ["localhost:27017"]
  username: user
  password: pass
```

To enable it, execute the `modules enable` command, passing one or more module names. For example:

```
metricbeat modules enable redis mongodb
```

Similar to disabling modules, execute the `modules disable` command, passing one or more module names to it. For example:

```
metricbeat modules disable redis mongodb
```

Note

To enable dynamic config reloading, set `reload.enabled` to true and specify a frequency with which to look for config file changes. Set the `reload.period` parameter under the `metricbeat.config.modules` property.

Enabling module configs in the metricbeat.yml file

If you're used to using earlier versions of Metricbeat, you can enable the appropriate modules and metricsets in the `metricbeat.yml` file directly by adding entries to the `metricbeat.modules` list. Each entry in the list begins with a dash (`-`) and is followed by the settings for that module. For example:

```
metricbeat.modules:
#------------------ Memcached Module -----------------------------
- module: memcached
  metricsets: ["stats"]
  period: 10s
  hosts: ["localhost:11211"]

#------------------- MongoDB Module -----------------------------
- module: mongodb
  metricsets: ["dbstats", "status"]
  period: 5s
```

General settings

This section contains configuration options and some general settings to control the behavior of Metricbeat.

Some of these configuration options/settings are as follows:

- `name` : The name of the shipper that publishes the network data. By default, the hostname is used for this field, as follows:

```
name: "dc1-host1"
```

- `tags` : A list of tags that will be included in the `tags` field of every event Metricbeat ships. Tags make it easy to group servers by different logical properties and are useful when filtering events in Kibana and Logstash, as follows:

```
tags: ["staging", "web-tier","dc1"]
```

- `max_procs` : The maximum number of CPUs that can be executing simultaneously. The default is the number of logical CPUs available in the system:

```
max_procs: 2
```

Output configuration

This section is all about configuring outputs where the events need to be shipped. Events can be sent to single or multiple outputs simultaneously. The allowed outputs are Elasticsearch, Logstash, Kafka, Redis, file, and console. Some outputs that can be configured are as follows:

- `elasticsearch` : This is used to send events directly to Elasticsearch. A sample Elasticsearch output configuration is shown in the following code snippet:

```
output.elasticsearch:
  enabled: true
  hosts: ["localhost:9200"]
```

Using the `enabled` setting, you can enable or disable the output. `hosts` accepts one or more Elasticsearch node/servers. Multiple hosts can be defined for failover purposes. When multiple hosts are configured, the events are distributed to these nodes in a round-robin order. If Elasticsearch is secure, then credentials can be passed using the `username` and `password` settings, as follows:

```
output.elasticsearch:
  enabled: true
  hosts: ["localhost:9200"]
  username => "elasticsearch"
  password => "elasticpassword"
```

To ship events to the Elasticsearch ingest node pipeline so that they can be preprocessed before being stored in Elasticsearch, pipeline information can be provided using the `pipleline` setting, as follows:

```
output.elasticsearch:
  enabled: true
  hosts: ["localhost:9200"]
  pipeline: "ngnix_log_pipeline"
```

The default index the data gets written to is in the `metricbeat-%{[beat.version]}-%{+yyyy.MM.dd}` format. This will create a new index every day. For example, if today is April 02, 2019, then all the events are placed in the `metricbeat-7.0.0-2019-04-02` index. You can override the index name or the pattern using the `index` setting. In the following configuration snippet, a new index is created for every month, as follows:

```
output.elasticsearch:
hosts: ["http://localhost:9200"]
  index: "metricbeat-%{[beat.version]}-%{+yyyy.MM}"
```

Using the `indices` setting, you can conditionally place the events in the appropriate index that matches the specified condition. In the following code snippet, if the message contains the `DEBUG` string, it will be placed in the `debug-%{+yyyy.MM.dd}` index. If the message contains the `ERR` string, it will be placed in the `error-%{+yyyy.MM.dd}` index. If the message contains neither of these strings, then those events will be pushed to the `logs-%{+yyyy.MM.dd}` index, as specified in the `index` parameter, as follows:

```
output.elasticsearch:
hosts: ["http://localhost:9200"]
  index: "logs-%{+yyyy.MM.dd}"indices:
-index: "debug-%{+yyyy.MM.dd}"
      when.contains:
message: "DEBUG"-index: "error-%{+yyyy.MM.dd}"
      when.contains:
message: "ERR"
```

Note

When the `index` parameter is overridden, disable templates and dashboards by adding the following settings:

`setup.dashboards.enabled: false``setup.template.enabled: false`

Alternatively, provide the values for `setup.template.name` and `setup.template.pattern` in the `metricbeat.yml` configuration file; otherwise, Metricbeat will fail to run.

- `logstash` : This is used to send events to Logstash.

Note

To use Logstash as output, Logstash needs to be configured with the Beats input plugin so it can receive incoming Beats events.

A sample Logstash output configuration is as follows:

```
output.logstash:
  enabled: true
  hosts: ["localhost:5044"]
```

Using the `enabled` setting, you can enable or disable the output. `hosts` accepts one or more Logstash servers. Multiple hosts can be defined for failover purposes. If the configured host is unresponsive, then the event will be sent to one of the other configured hosts. When multiple hosts are configured,events are distributed in a random order. To enable load-balancing events across the Logstash hosts, use the `loadbalance` flag, set to `true` , as follows:

```
output.logstash:
hosts: ["localhost:5045", "localhost:5046"]
  loadbalance: true
```

- `console` : This is used to send events to `stdout` . These events are written in JSON format. This is useful during debugging or testing.

A sample console configuration is as follows:

```
output.console:
  enabled: true
pretty: true
```

Logging

This section contains the options for configuring the Metricbeat logging output. The logging system can write logs to syslog or rotate log files. If logging is not explicitly configured, file output is used on Windows systems, and syslog output is used on Linux and OS X.

A sample configuration is as follows:

```
logging.level: debug
logging.to_files: true
logging.files:
 path: /elasticstack/logs/metricbeat
 name: metricbeat.log
 keepfiles: 10
```

Some of the available configuration options are as follows:

- `level`: To specify the logging level.
- `to_files`: To write all logging output to files. The files are subject to file rotation. This is the default value.
- `to_syslog`: To write logging output to syslogs if this setting is set to `true`.
- `files.path`, `files.name`, and `files.keepfiles`: These are used to specify the location of the file, the name of the file, and the number of recently rotated log files to keep on the disk.

Running Metricbeat with the system module

Let's make use of Metricbeat and capture system metrics.

Make sure that Kibana 7.0 and Elasticsearch 7.0 are running:

1. Replace the content of `metricbeat.yml` with the following configuration and save the file:

```
############### Metricbeat Configuration Example #################
#============== Modules configuration ==========================

metricbeat.config.modules:
  # Glob pattern for configuration loading
  path: ${path.config}/modules.d/*.yml

  # Set to true to enable config reloading
  reload.enabled: false

  # Period on which files under path should be checked for changes
  #reload.period: 10s

#========== Elasticsearch template setting =================

setup.template.settings:
  index.number_of_shards: 1
  index.codec: best_compression
  #_source.enabled: false
```

```
#================== General  Settings================================
name: metricbeat_inst1

tags: ["system-metrics", "localhost"]

fields:
  env: test-env

#============================== Dashboards ============================
setup.dashboards.enabled: true

#============================== Kibana Settings ============================
setup.kibana:
  host: "localhost:5601"
  #username: "elastic"
  #password: "changeme"

#------------------------- Elasticsearch output Settings ---------------------
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
  #username: "elastic"
  #password: "changeme"
```

Note

The `setup.dashboards.enabled: true` setting loads sample dashboards to the Kibana index during startup, which are loaded via the Kibana API. If Elasticsearch and Kibana are secured, make sure that you uncomment the `username` and `password` parameters and set the appropriate values.

2. By default, the `system` module is enabled. Make sure that it is enabled by executing the following command:

```
cd /elasticstack/metricbeat-7.12.1-linux-x86_64

./metricbeat modules enable system
```

3. You can verify the metricsets that are enabled for the `system` module by opening the `system.yml` file, which can be found under the `modules.d` directory, as follows:

```
#system.yml
- module: system
  period: 10s
  metricsets:
    - cpu
    #- load
    - memory
    - network
    - process
    - process_summary
    #- socket_summary
```

```
    #- core
    #- diskio
    #- socket
  processes: ['.*']
  process.include_top_n:
    by_cpu: 5 # include top 5 processes by CPU
    by_memory: 5 # include top 5 processes by memory

- module: system
  period: 1m
  metricsets:
    - filesystem
    - fsstat
  processors:
  - drop_event.when.regexp:
      system.filesystem.mount_point: '^/(sys|cgroup|proc|dev|etc|host|lib)($|/)'
```

As seen in the preceding code, the configuration module is defined twice, with different periods to use for a set of metricsets. The `cpu`, `memory`, `network`, `process`, `process_summary`, `filesystem`, and `fsstats` metricsets are enabled.

4. Start Metricbeat by executing the following command:

```
cd /elasticstack/metricbeat-7.12.1-linux-x86_64

./metricbeat -e
```

Note: You will get error message because we have enabled mongodb and redis but they are not installed in the lab environment.

Once Metricbeat is started, it loads sample Kibana dashboards and starts shipping metrics to Elasticsearch. To validate this, execute the following command:

```
curl -X GET 'http://localhost:9200/_cat/indices?v=&format=json'

Sample Response:
[
    {
        "health": "yellow",
        "status": "open",
        "index": "metricbeat-7.0.0-2019.04.02",
        "uuid": "w2WoP2IhQ9eG7vSU_HmgnA",
        "pri": "1",
        "rep": "1",
        "docs.count": "29",
        "docs.deleted": "0",
        "store.size": "45.3kb",
        "pri.store.size": "45.3kb"
    },
    {
        "health": "yellow",
        "status": "open",
        "index": ".kibana",
```

```
        "uuid": "sSzeYu-YTtWR8vr2nzKrbg",
        "pri": "1",
        "rep": "1",
        "docs.count": "108",
        "docs.deleted": "59",
        "store.size": "289.3kb",
        "pri.store.size": "289.3kb"
    }
]

curl -X GET 'http://localhost:9200/_cat/indices?v'

health status index uuid pri rep docs.count docs.deleted store.size pri.store.size
yellow open metricbeat-7.0.0-2019.04.02 w2WoP2IhQ9eG7vSU_HmgnA 1 1 29 0 45.3kb 45.3kb
yellow open .kibana sSzeYu-YTtWR8vr2nzKrbg 1 1 108 59 289.3kb 289.3kb
```

Specifying aliases

Let's say the IT admin creates an alias pointing to all the indexes containing the metrics for a specific month. For example, as shown in the following code snippet, an alias called `april_04_metrics` is created for all the indexes of the `metricbeat-7.0.0-2019.04.*` pattern, that is, those Metricbeats indexes that are created on a daily basis in the month of April 2019:

```
curl -X POST   http://localhost:9200/_aliases  -H 'content-type: application/json' -d
'
{
 "actions":
 [
   {"add":{ "index" : "metricbeat-7.0.0-2019.04.*", "alias": "april_04_metrics"} }
   ]
}'
```

Now, using the `april_04_metrics` alias name, the query can be executed against all the indexes of the `metricbeat-7.0.0-2019.04.*` pattern as follows:

```
curl -X GET http://localhost:9200/april_04_metrics/_search
```

In the following example, the `sales` alias is created against the `it_sales` and `retail_sales` indexes. In the future, if a new sales index gets created, then that index can also point to the `sales` index so that the end user/application can always make use of the `sales` endpoint to query all sales data, as follows:

```
curl -X POST   http://localhost:9200/_aliases -d '{
"actions":[
    {"add":{"index":"it_sales","alias":"sales"}},
    {"add":{"index":"retail_sales","alias":"sales"}}
]}
```

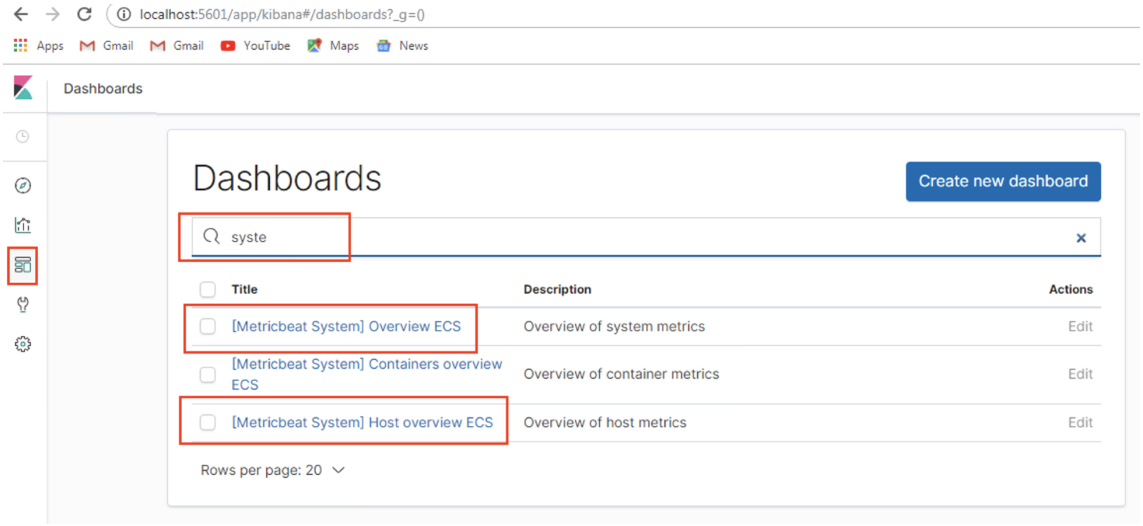
To remove an alias from an index, use the `remove` action of the aliases API, as follows:

```
curl -X POST   http://localhost:9200/_aliases -d '
{"actions":[{"remove":{"index":"retail_sales","alias":"sales"}}]}
```

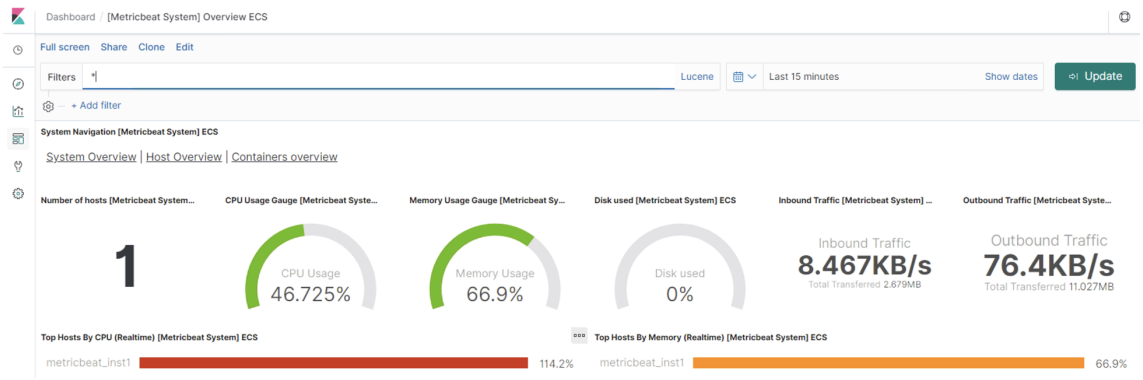
Visualizing system metrics using Kibana

To visualize the system metrics using Kibana, execute the following steps:

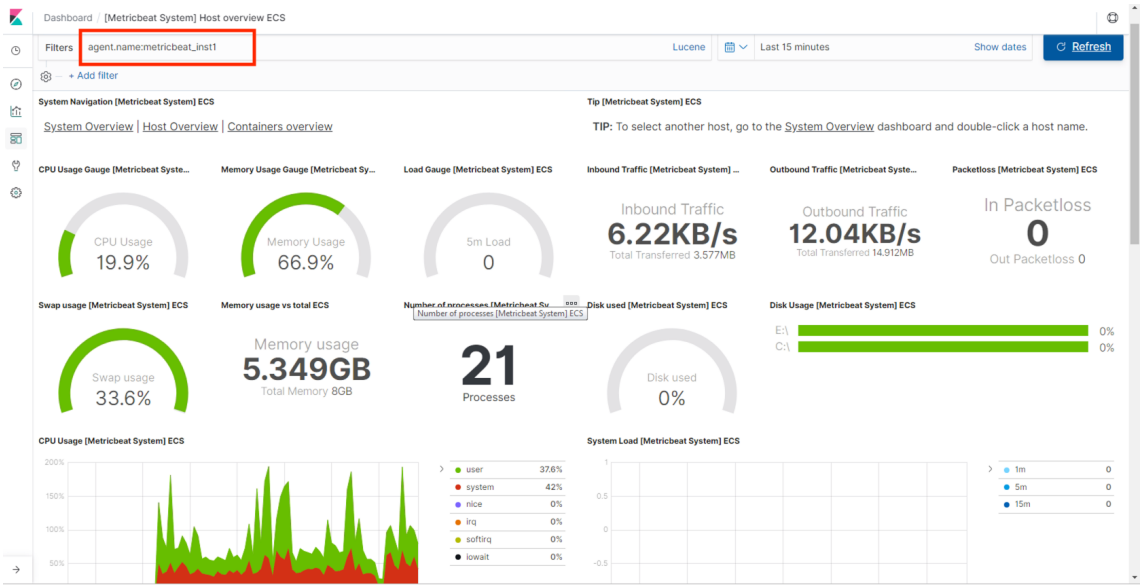
1. Navigate to `http://localhost:5601` and open up Kibana.
2. Click on the `Dashboard` link found in the left navigation menu and select either `[Metricbeat System] Overview ECS` or `[Metricbeat System] Host Overview ECS` from the dashboard, as shown in the following screenshot:



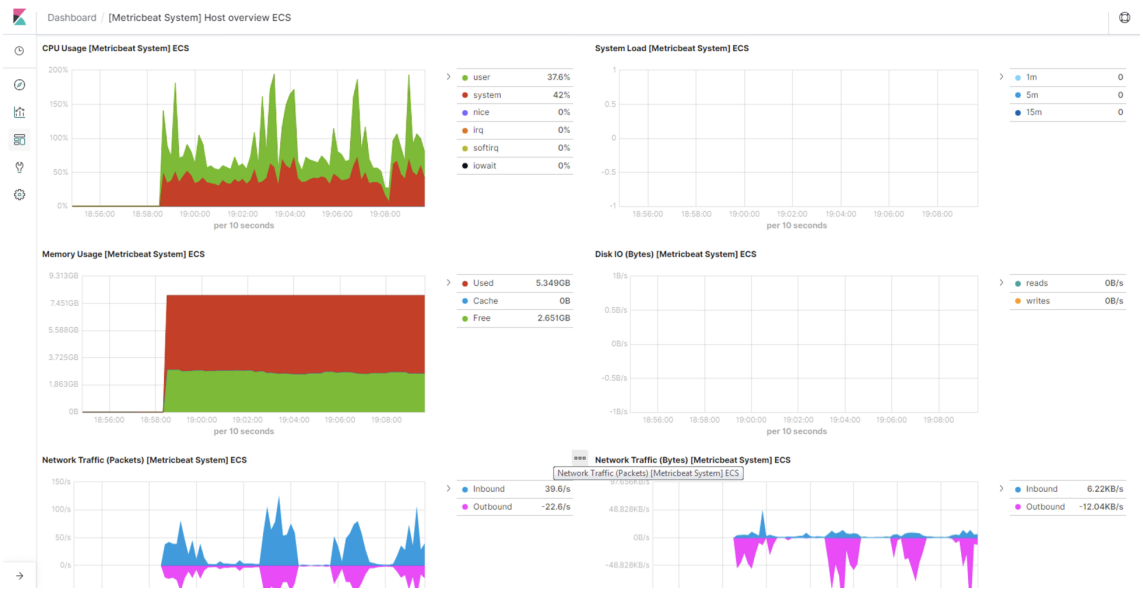
[Metricbeat System] Overview Dashboard ECS: This dashboard provides an overview of all the systems that are being monitored. Since we are monitoring only a single host, we see that the **Number of hosts** is **1**, as shown in the following screenshot:



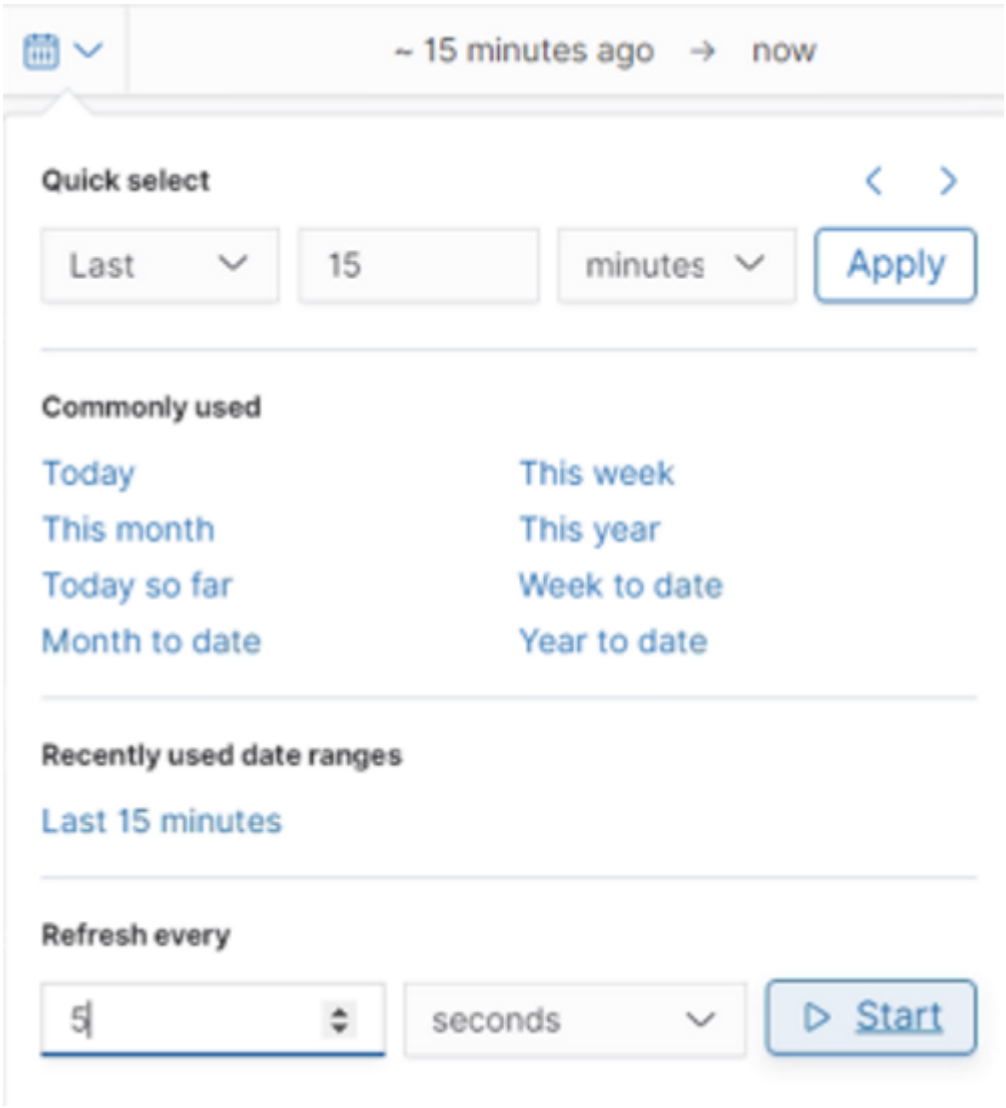
Metricbeat Host Overview Dashboard: This dashboard is useful for finding the detailed metrics of individual systems/hosts. In order to filter metrics based on a particular host, enter the search/filter criterion in the search/query bar. In the following screenshot, the filter criterion is **agent.name:metricbeat_inst1**. Any attribute that uniquely identifies a system/host can be used; for example, you can filter based on **host.hostname**, as follows:



Since the `diskio` and `load` metricsets were disabled in the system module configuration, we will see empty visualizations for the **Disk IO** and **System Load** visualizations, as shown in the following screenshot:



To see the dashboard refresh in real time, in the top right corner select the time and enter the appropriate refresh interval. Then, click the `Start` button as shown in the following screenshot:



Summary

In this lab, we covered another Beat library called Metricbeat in detail. We covered how to install and configure Metricbeat so that it can send operational metrics to Elasticsearch. We also covered the various deployment architectures for building real-time monitoring solutions using Elasticsearch Stack in order to monitor servers and applications. This helps IT administrators and application support personnel gain insights into the behavior of applications and servers, and allows them to respond in a timely manner in the event of an infrastructure outage.