A. Create two linux servers,

server1 => Install and configure kibana and elasticsearch with basic username and password authentication

server2 => install and configure metricbeat.

Kibana and elastic search are installed and configured in my host device which runs Ubuntu 20.04.

I already had some required dependencies like default-jdk, nginx.

And used the following commands to install Kibana and ElasticSearch:

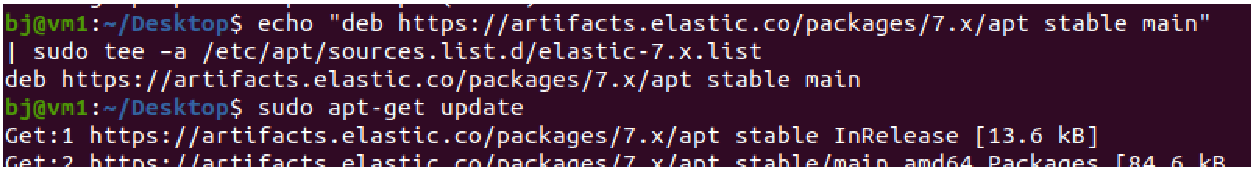
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -

sudo apt-get install apt-transport-https

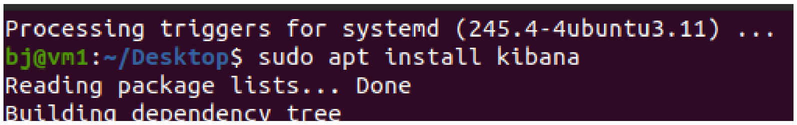
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee –a /etc/apt/sources.list.d/elastic-7.x.list

sudo apt-get update

sudo apt-get install elasticsearch kibana



```
bj@vm1:~/Desktop$ echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main"
| sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
deb https://artifacts.elastic.co/packages/7.x/apt stable main
bj@vm1:~/Desktop$ sudo apt-get update
Get:1 https://artifacts.elastic.co/packages/7.x/apt stable InRelease [13.6 kB]
Get:2 https://artifacts.elastic.co/packages/7.x/apt stable/main amd64 Packages [84.6 kB
```

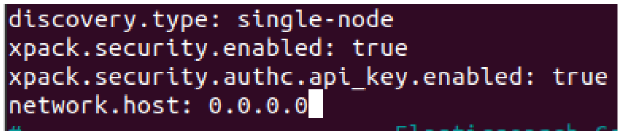


```
Processing triggers for systemd (245.4-4ubuntu3.11) ...
bj@vm1:~/Desktop$ sudo apt install kibana
Reading package lists... Done
Building dependency tree
```

After installation, the configuration files are edited for elasticsearch using command:

Nano /etc/elasticsearch/elasticsearch.yml

Inside the document, I added following configurations for authentication:



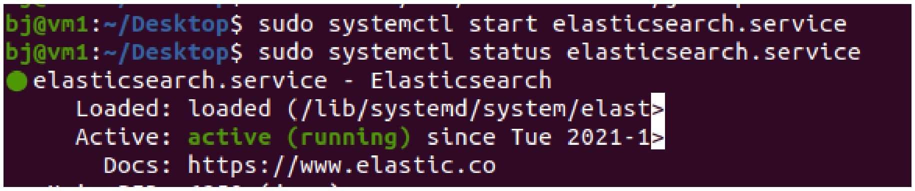
```
discovery.type: single-node
xpack.security.enabled: true
xpack.security.authc.api_key.enabled: true
network.host: 0.0.0.0
```

After then, elastic search was enabled and started with commands:

sudo systemctl enable elasticsearch

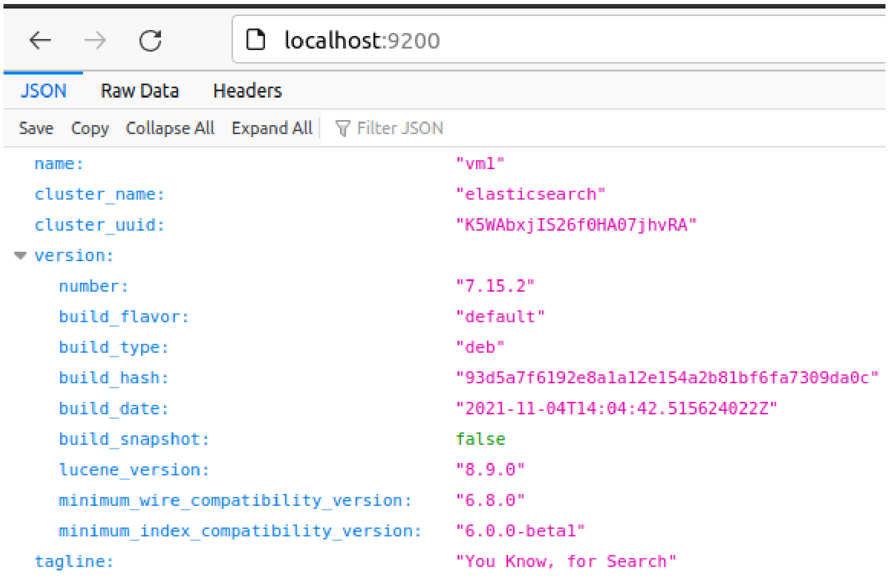
sudo systemctl start elasticsearch

Sudo systemctl status elastic search



```
bj@vm1:~/Desktop$ sudo systemctl start elasticsearch.service
bj@vm1:~/Desktop$ sudo systemctl status elasticsearch.service
● elasticsearch.service - Elasticsearch
     Loaded: loaded (/lib/systemd/system/elast>
     Active: active (running) since Tue 2021-1>
       Docs: https://www.elastic.co
```

If we go to the browser and enter server IP (localhost or 192.168.1.67) with port 9200, we can see this content by default.

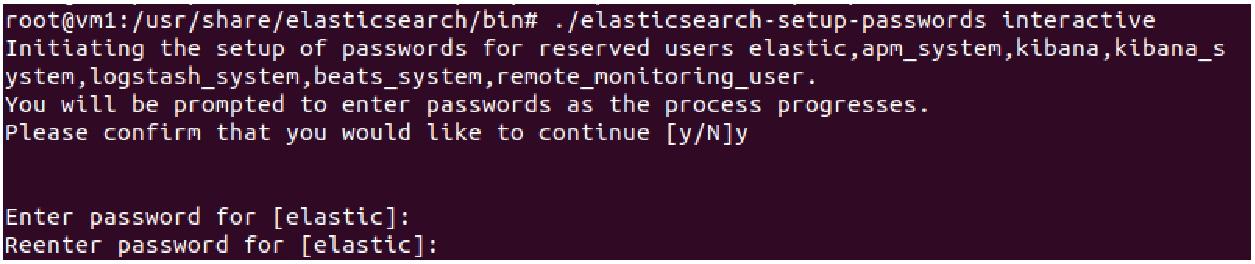


```
name:                                  "vm1"
cluster_name:                          "elasticsearch"
cluster_uuid:                          "K5WAbxjIS26f0HA07jhvRA"
▼ version:
    number:                            "7.15.2"
    build_flavor:                      "default"
    build_type:                        "deb"
    build_hash:                        "93d5a7f6192e8a1a12e154a2b81bf6fa7309da0c"
    build_date:                        "2021-11-04T14:04:42.515624022Z"
    build_snapshot:                    false
    lucene_version:                    "8.9.0"
    minimum_wire_compatibility_version:  "6.8.0"
    minimum_index_compatibility_version: "6.0.0-beta1"
tagline:                               "You Know, for Search"
```

Now we go to /usr/share/elasticsearch/bin and set up a password for authentication when we access the content for elastic search from the browser.

Cd /usr/share/elasticsearch/bin

./elasticsearch-setup-passwords interactive



```
root@vm1:/usr/share/elasticsearch/bin# ./elasticsearch-setup-passwords interactive
Initiating the setup of passwords for reserved users elastic,apm_system,kibana,kibana_s
ystem,logstash_system,beats_system,remote_monitoring_user.
You will be prompted to enter passwords as the process progresses.
Please confirm that you would like to continue [y/N]y


Enter password for [elastic]:
Reenter password for [elastic]:
```

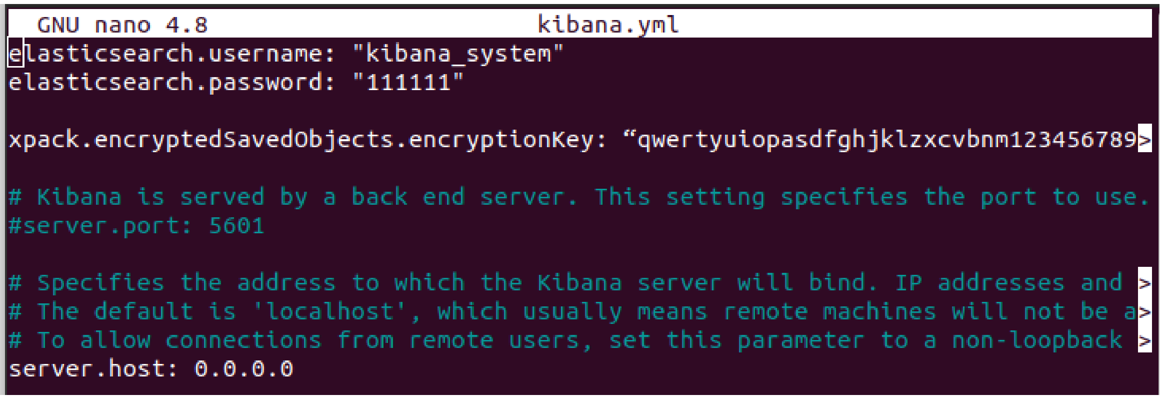
Systemctl restart elasticsearch

Now, in the browser, access to localhost:9200 asks for authentication and only after entering username and password, we can see the elasticsearch page.

Now Kibana is configured:

Cd /etc/kibana/

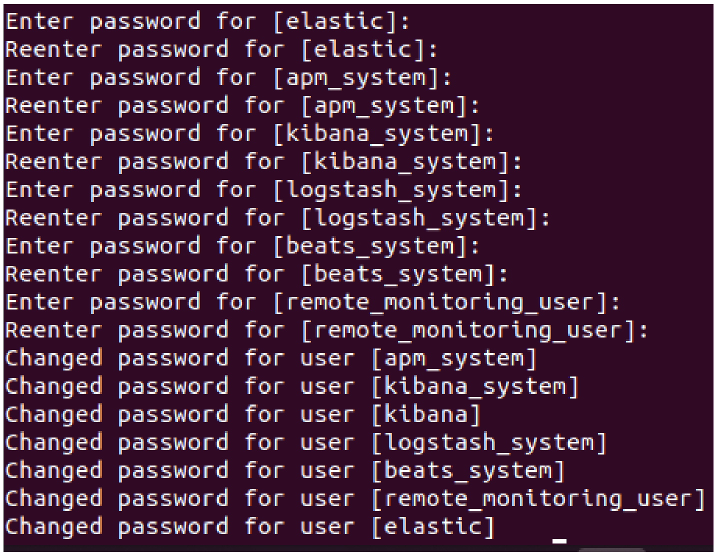
Nano kibana.yml



```
  GNU nano 4.8                           kibana.yml
elasticsearch.username: "kibana_system"
elasticsearch.password: "111111"

xpack.encryptedSavedObjects.encryptionKey: “qwertyuiopasdfghjklzxcvbnm123456789>

# Kibana is served by a back end server. This setting specifies the port to use.
#server.port: 5601

# Specifies the address to which the Kibana server will bind. IP addresses and >
# The default is 'localhost', which usually means remote machines will not be a>
# To allow connections from remote users, set this parameter to a non-loopback >
server.host: 0.0.0.0
```

(*The xpack configuration is for alerting rules which was added later on to set up alerts on slack channel*)

We have to use the username and password for kibana. We can set any user and password which we set before in this page:



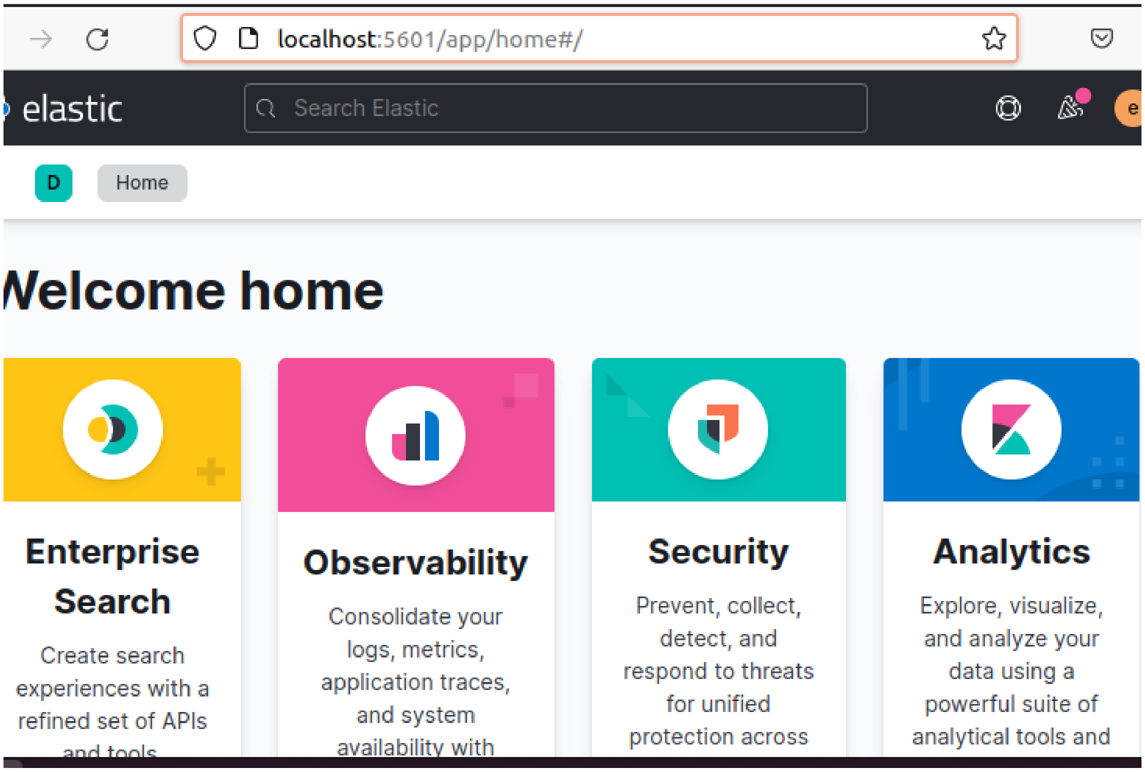
```
Enter password for [elastic]:
Reenter password for [elastic]:
Enter password for [apm_system]:
Reenter password for [apm_system]:
Enter password for [kibana_system]:
Reenter password for [kibana_system]:
Enter password for [logstash_system]:
Reenter password for [logstash_system]:
Enter password for [beats_system]:
Reenter password for [beats_system]:
Enter password for [remote_monitoring_user]:
Reenter password for [remote_monitoring_user]:
Changed password for user [apm_system]
Changed password for user [kibana_system]
Changed password for user [kibana]
Changed password for user [logstash_system]
Changed password for user [beats_system]
Changed password for user [remote_monitoring_user]
Changed password for user [elastic]
```

We should use the valid username and password which we set in the previous step when we created interactive passwords for each user.

Now kibana is restarted:

Systemctl restart kibana

And when we go to browser and enter server ip with port 5601, we can see login page of kibana, and after providing correct credentials, we can see the dashboard of kibana:



Welcome home

Enterprise Search
Create search experiences with a refined set of APIs and tools

Observability
Consolidate your logs, metrics, application traces, and system availability with

Security
Prevent, collect, detect, and respond to threats for unified protection across

Analytics
Explore, visualize, and analyze your data using a powerful suite of analytical tools and

Now in virtualbox, we installed another ubuntu virtual machine and inside it, we configured logstash and metricbeat using following commands:

curl -L -O https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-7.15.2-amd64.deb

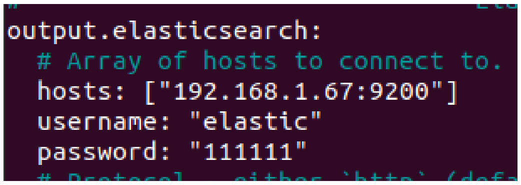
sudo dpkg -i metricbeat-7.15.2-amd64.deb

And configured metricbeat for local environment:

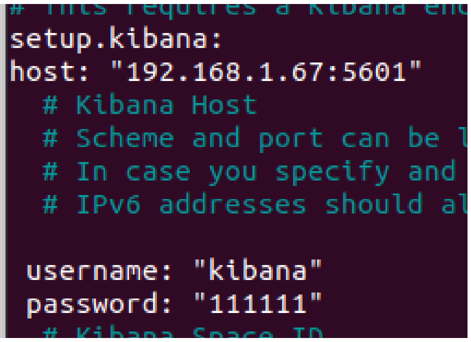
Cd /etc/metricbeat

Nano metricbeat.yml

And inside this file, we should configure elasticsearch and kibana setup as:



```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["192.168.1.67:9200"]
  username: "elastic"
  password: "111111"
  # Protocol - either `http` (defa
```



```
# This requires a Kibana end
setup.kibana:
host: "192.168.1.67:5601"
  # Kibana Host
  # Scheme and port can be l
  # In case you specify and
  # IPv6 addresses should al

 username: "kibana"
 password: "111111"
  # Kibana Space ID
```

Now the modules can be enabled if we desire to use modules of different applications. But we need to monitor system metrics, so we don't need to enable any modules. We only need a system module which is enabled by default.

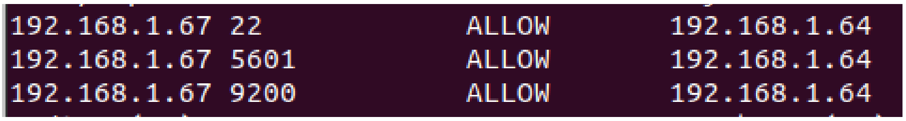
In our host machine, we should allow the incoming requests from our server machine in our firewall on two specific ports, 5601 and 9200.

Sudo ufw allow from 192.168.1.64 to 192.168.1.67 port 5601

Sudo ufw allow from 192.168.1.64 to 192.168.1.67 port 9200

Sudo ufw reload

And we can use ufw status to see if the rules are updated in the table.



```
192.168.1.67 22             ALLOW       192.168.1.64
192.168.1.67 5601           ALLOW       192.168.1.64
192.168.1.67 9200           ALLOW       192.168.1.64
```

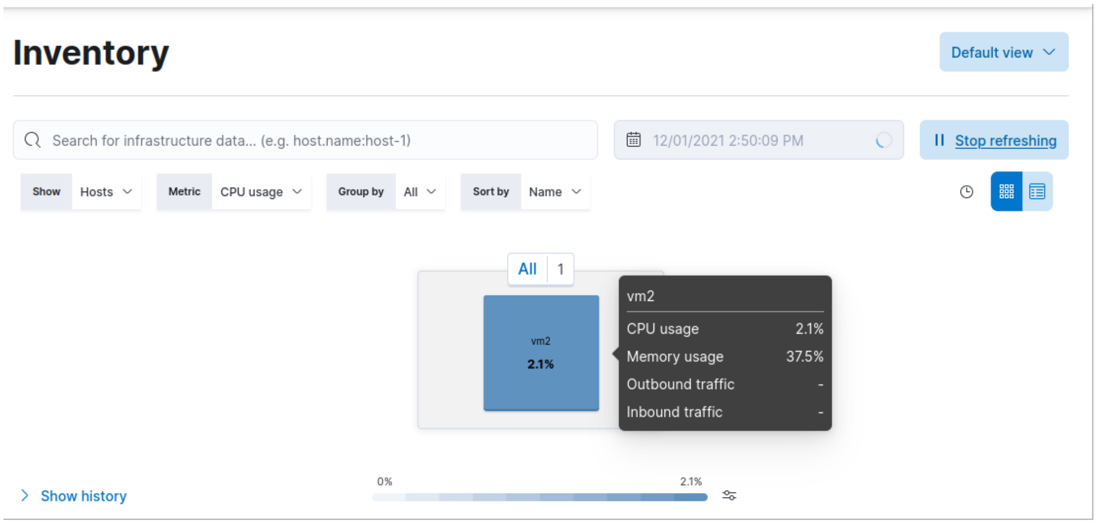
Now metricbeat is restarted using command:

Systemctl restart metricbeat

Now, metricbeat service is started and we can start monitoring by command:

Metricbeat -e

And when we view the kibana dashboard, we can see the metrics for VM2:



vm2

CPU usage 2.1%

Memory usage 37.5%

Outbound traffic -

Inbound traffic -

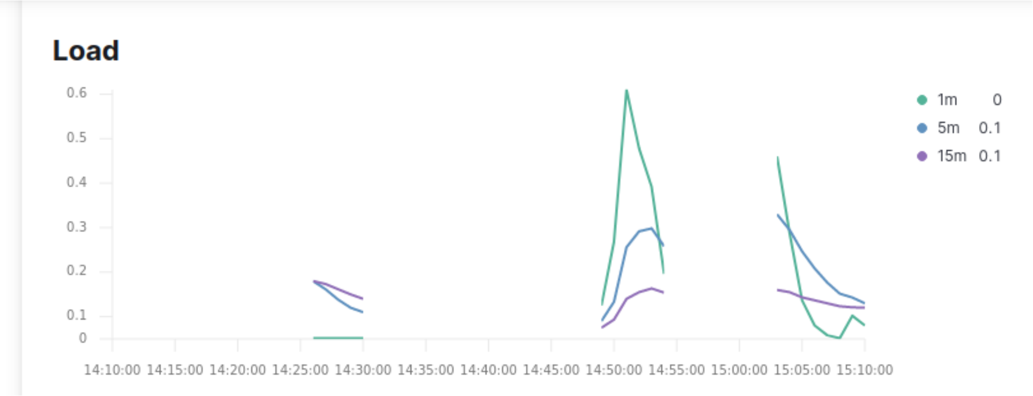
Collect metric from following sources in server1 and send them to elasticsearch. Store them in an index named "server1-metrics".

a. Memory usage

b. Disk usage

c. Load average

1. Create a dashboard in kibana and generate visual report(line graph) for Memory usage and load average of server1 with relation to time



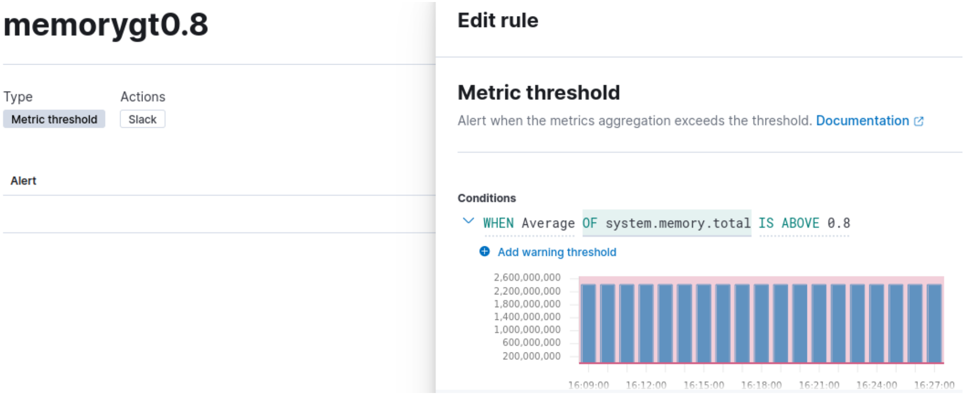
2. Generate alerts through the kibana system for following thresholds

   When memory usage > 80% for the last 2 minutes send an alert to a slack channel

When Disk usage > 70% send alerts to a slack channel

When load average > 1 for last 2 minutes send alert to a slack channel

To set up the alert rules, we can go to stack management and then rules and connectors and create our own rules. For memory consumption, we can set rule as shown below:



memorygt0.8

Type: Metric threshold
Actions: Slack

Alert

Edit rule

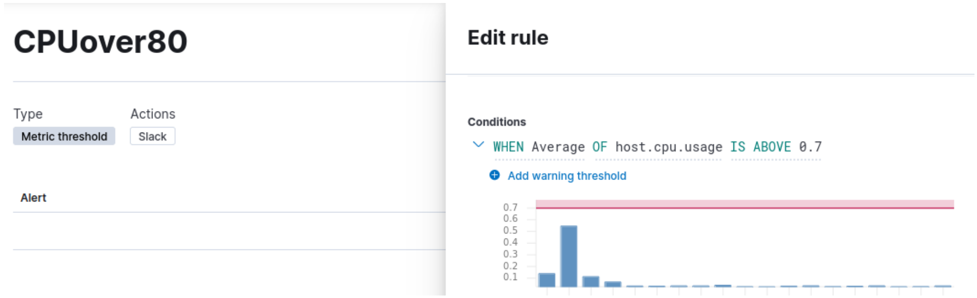
Metric threshold

Alert when the metrics aggregation exceeds the threshold. Documentation

Conditions

WHEN Average OF system.memory.total IS ABOVE 0.8

Similarly, for CPU over 70 percent:



CPUover80

Type: Metric threshold
Actions: Slack

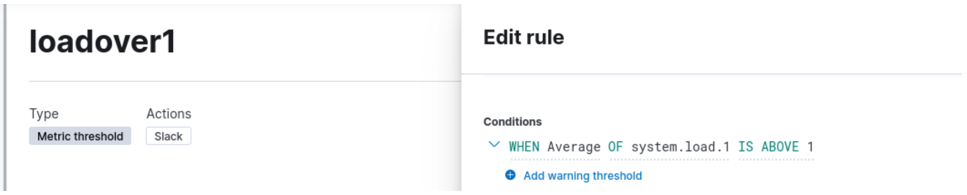
Alert

Edit rule

Conditions

WHEN Average OF host.cpu.usage IS ABOVE 0.7

And for load over 1:



loadover1

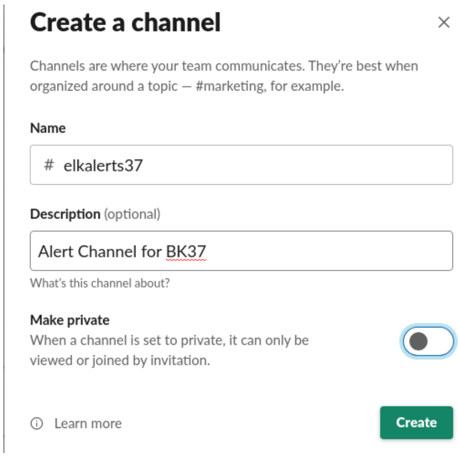
Type: Metric threshold
Actions: Slack

Edit rule

Conditions

WHEN Average OF system.load.1 IS ABOVE 1

To set these alerts in a slack channel, we first create a channel in slack: elkalerts37



Create a channel

Channels are where your team communicates. They're best when organized around a topic — #marketing, for example.

Name

elkalerts37

Description (optional)

Alert Channel for BK37

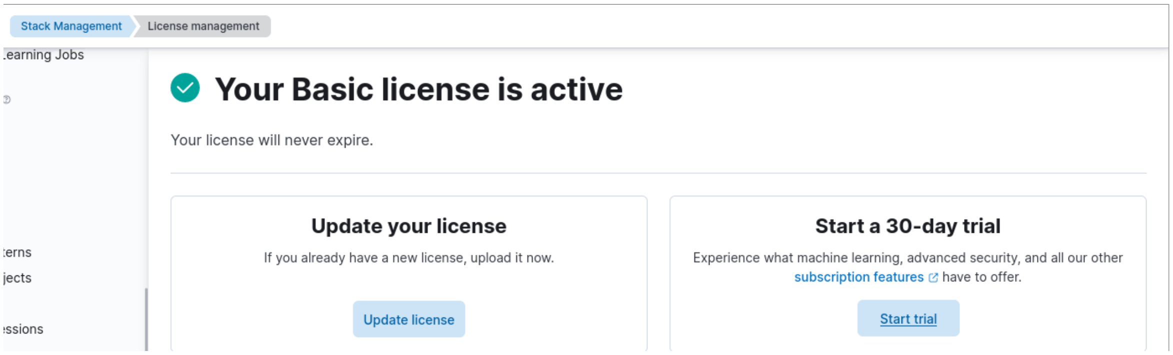
What's this channel about?

Make private

When a channel is set to private, it can only be viewed or joined by invitation.

Now we can start a 30 day trial to test this alerting mechanism from:

stack management >> License management



Your Basic license is active

Your license will never expire.

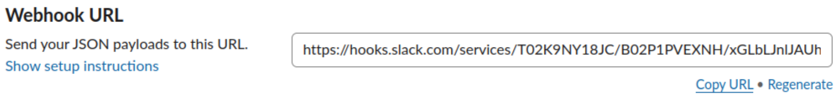
Update your license

If you already have a new license, upload it now.

Start a 30-day trial

Experience what machine learning, advanced security, and all our other subscription features have to offer.

And create a webhook URL for slack channel:

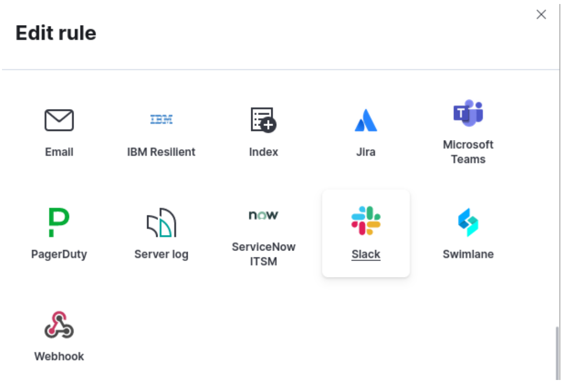


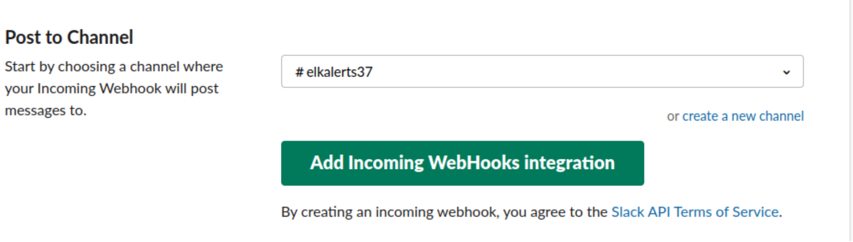
Webhook URL

Send your JSON payloads to this URL.

https://hooks.slack.com/services/T02K9NY18JC/B02P1PVEXNH/xGLbLJnlJAUh

Now when we edit those rules we set previously, we can see slack option as below:





Post to Channel

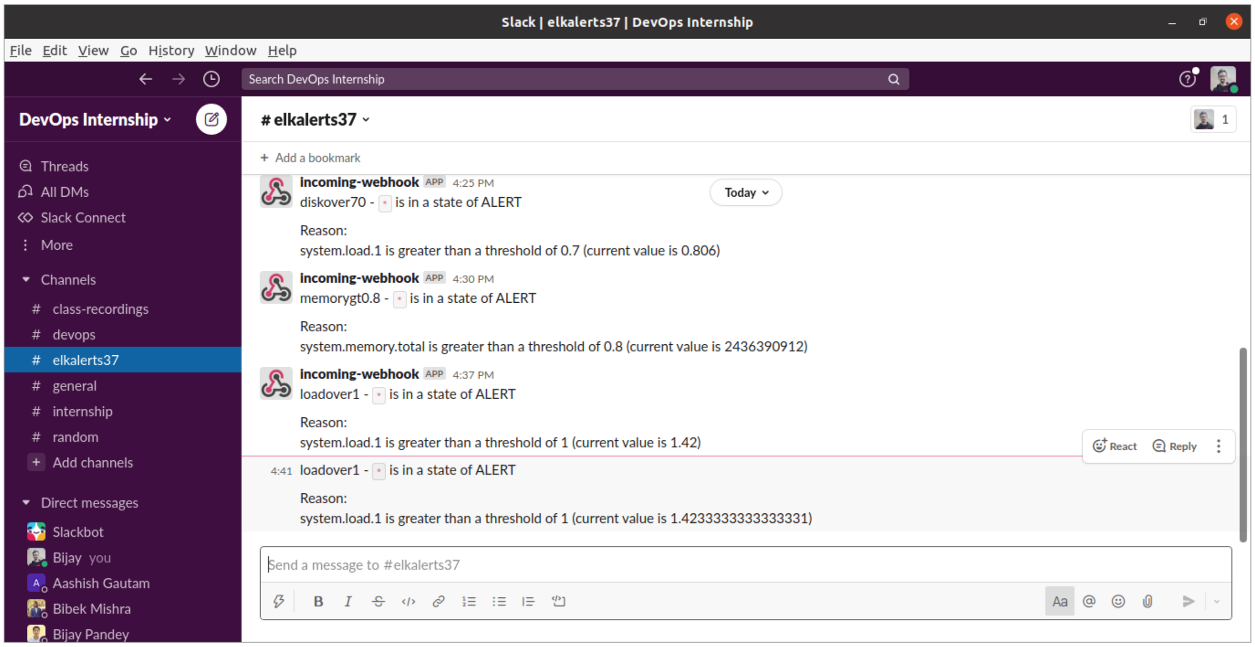
Start by choosing a channel where your Incoming Webhook will post messages to.

elkalerts37

By creating an incoming webhook, you agree to the Slack API Terms of Service.

Incoming webhooks integrations are added in slack.

Here are some of the alerts generated after all these steps:



incoming-webhook APP 4:25 PM
diskover70 - is in a state of ALERT

Reason:
system.load.1 is greater than a threshold of 0.7 (current value is 0.806)

incoming-webhook APP 4:30 PM
memorygt0.8 - is in a state of ALERT

Reason:
system.memory.total is greater than a threshold of 0.8 (current value is 2436390912)

incoming-webhook APP 4:37 PM
loadover1 - is in a state of ALERT

Reason:
system.load.1 is greater than a threshold of 1 (current value is 1.42)

4:41 loadover1 - is in a state of ALERT

Reason:
system.load.1 is greater than a threshold of 1 (current value is 1.4233333333333331)