**Name: Venkata Lahari Goruputi**

**Role: DevOps Intern**

**Topic: ELK stack set up and EC2 Instance Monitoring in ELK**

**Creating ELK stack using Docker and Monitoring an EC2 Instance using ELK:**

**What is ELK:**

* ELK is an acronym that stands for **Elasticsearch**, **Logstash**, and **Kibana**. These are three open-source projects that are often used together to collect, process, and visualize data in real-time. Together, they form a powerful stack for managing and analyzing log data, often referred to as the ELK Stack.

**Elasticsearch:**

* Elasticsearch is a distributed, RESTful search and analytics engine capable of addressing a growing number of use cases. As the heart of the Elastic Stack, it centrally stores your data so you can discover the expected and uncover the unexpected.
* Elasticsearch is a Search and Analytics Engine.
* It is helpful in Full-text search, real-time search and analytics, scalability, and high availability.

**Logstash:**

* Logstash is a server-side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a "stash" like Elasticsearch.
* It is useful for Data collection, parsing, enrichment, and transformation, support for various input/output plugins.

**Kibana:**

* Kibana is a data visualization and exploration tool used for log and time-series analytics, application monitoring, and operational intelligence use cases. It provides powerful and easy-to-use features like histograms, line graphs, pie charts, and maps.
* It is the Data Visualization tool.
* Interactive charts, dashboards, data exploration tools, and the ability to search and view data stored in Elasticsearch indices.

**How They Work Together:**

**Data Ingestion with Logstash:**

* Logstash collects and processes data from various sources (e.g., logs, metrics, web applications) and sends the processed data to Elasticsearch.

**Data Storage and Search with Elasticsearch:**

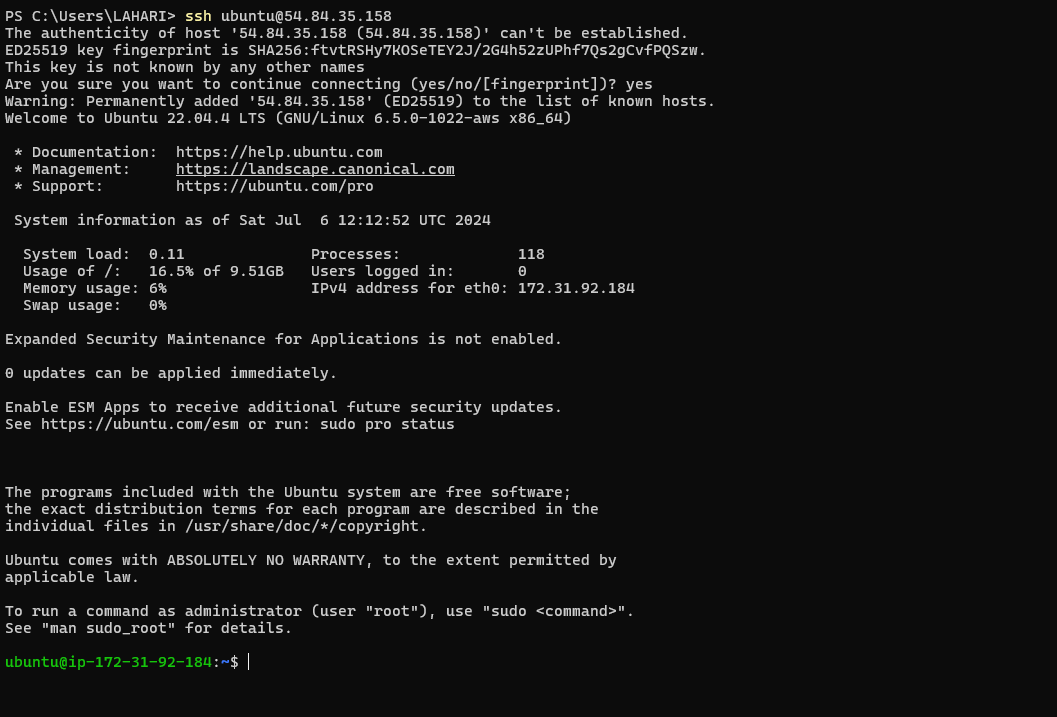
* Elasticsearch stores the data indexed by Logstash and allows for fast search and analysis. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface.

**Data Visualization with Kibana:**

* Kibana connects to Elasticsearch and provides a web interface to visualize the data. Users can create dashboards and perform complex queries to gain insights from the data stored in Elasticsearch.
* Elasticsearch would index and store these logs.
* Logstash would collect logs from different servers.
* Kibana would visualize the log data, allowing system administrators to create dashboards that display server health, performance metrics, and error logs.
* **Set Up Process:**
* As part of ELK set up, we need to create an EC2 instance and install **docker** in it as we are setting up ELK via docker.

[Docker Install Script](https://get.docker.com/)

* I have created an **ubuntu-22.04** with size **t2.large** EC2 instance in AWS.
* Log into it using the command **[ssh ubuntu@<Public-Ip>].**
* After log in to the instance we need to install docker.



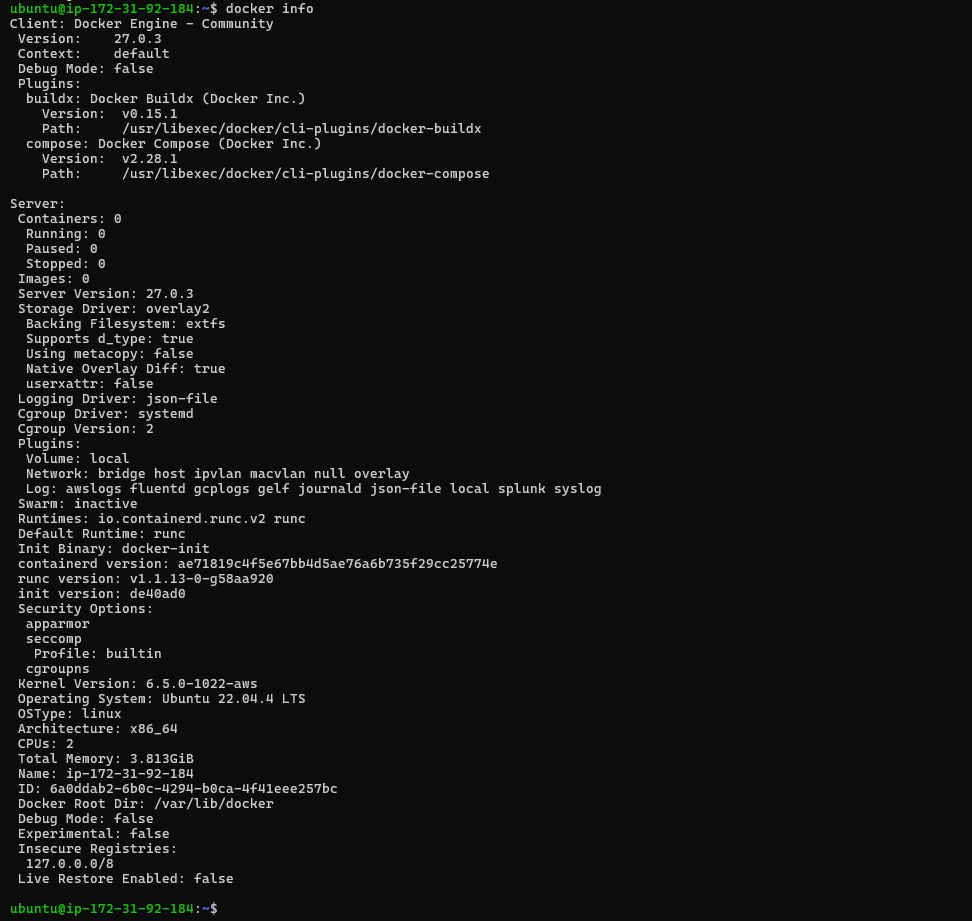
**Steps to install docker:**

curl -fsSL https://get.docker.com -o install-docker.sh

sh install-docker.sh

sudo usermod -aG docker <username-of-the-server>

* After doing the above steps logout of the server and login again and check for docker using **docker info** command, if this returns fine then docker is up and running.

****

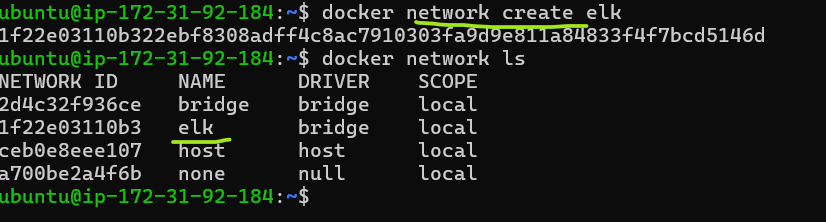
* **ELK stack set up:**
* Create a new docker network to deploy all the ELK containers in one network so that every container in that network can communicate with each other using the container names also.

**Create network in docker:**

**Command:**

**[docker network create elk]**

* The above command will create a network named **elk**. we will deploy all the ELK containers in this network only.

****

**Installing ElasticSearch:**

docker run -d --name elasticsearch --net elk --restart unless-stopped \

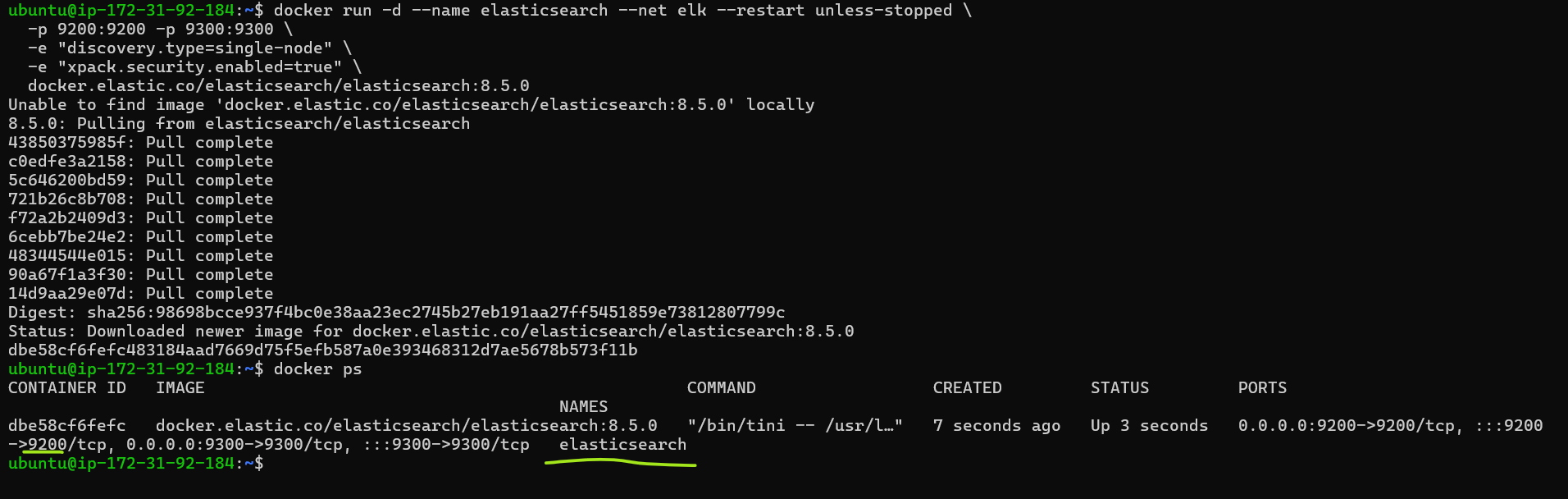
-p 9200:9200 -p 9300:9300 \

-e "discovery.type=single-node" \

-e "xpack.security.enabled=true" \

docker.elastic.co/elasticsearch/elasticsearch:8.5.0

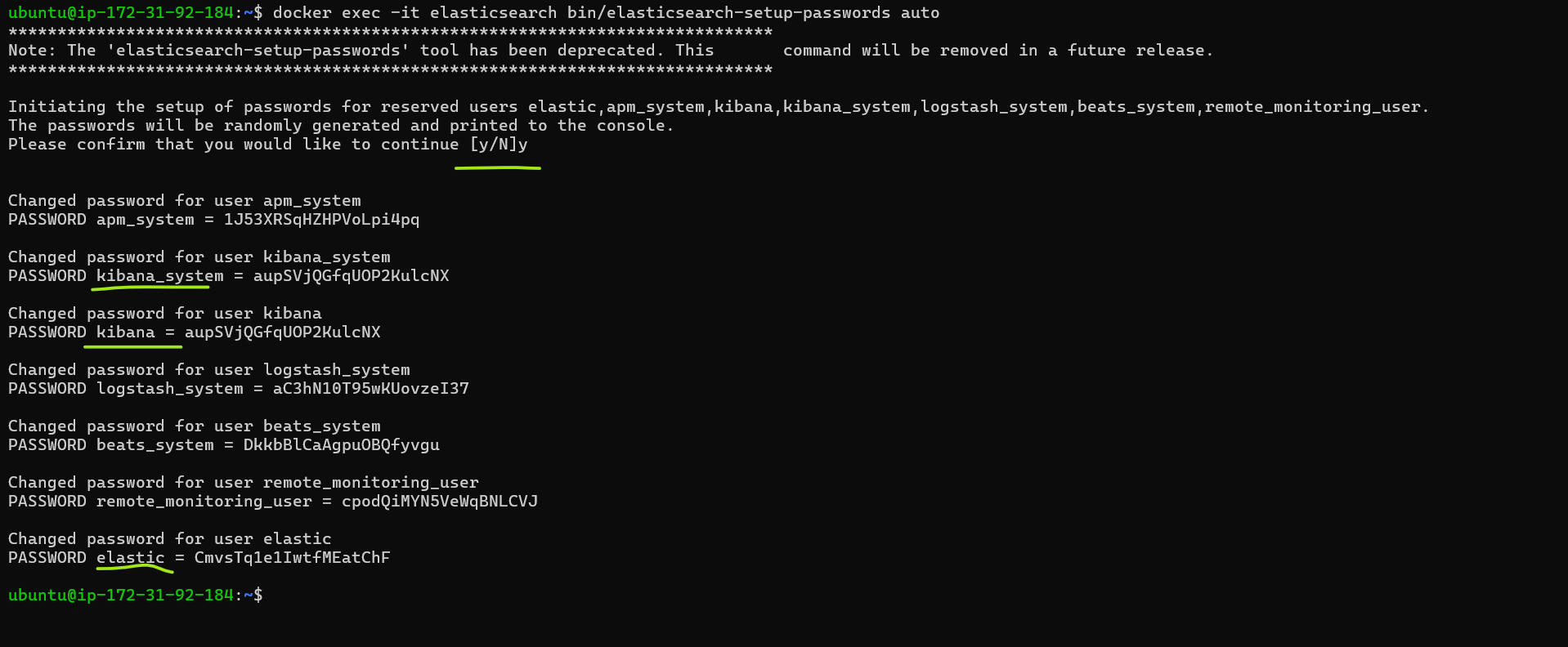
* The above command will deploy elasticsearch container on port **9200** and **9300**.
* Then we need to generate the passwords for all the users which will be required for elastic search.
* Give 2 minutes of time after running the above command so that elastic search will start run.
* Then use the below command to auto generate the user passwords.

****

**Command:**

**[docker exec -it elasticsearch bin/elasticsearch-setup-passwords auto]**

* The above command will auto generate the passwords for the users which looks like as below.



* **Changed password for user apm\_system**

PASSWORD apm\_system = 2jJwtege0YwrBH0Zpirv

* **Changed password for user kibana\_system**

PASSWORD kibana\_system = BGRWL9tNfpFhCvGJglRD

* **Changed password for user kibana**

PASSWORD kibana = BGRWL9tNfpFhCvGJglRD

* **Changed password for user logstash\_system**

PASSWORD logstash\_system = mdLdcNvEo2qWneSMbJ5t

* **Changed password for user beats\_system**

PASSWORD beats\_system = Qpkjit9RXFX5ZeZRJ1UB

* **Changed password for user remote\_monitoring\_user**

PASSWORD remote\_monitoring\_user = X4iu8glhWivgLhkwe1Ly

* **Changed password for user elastic**

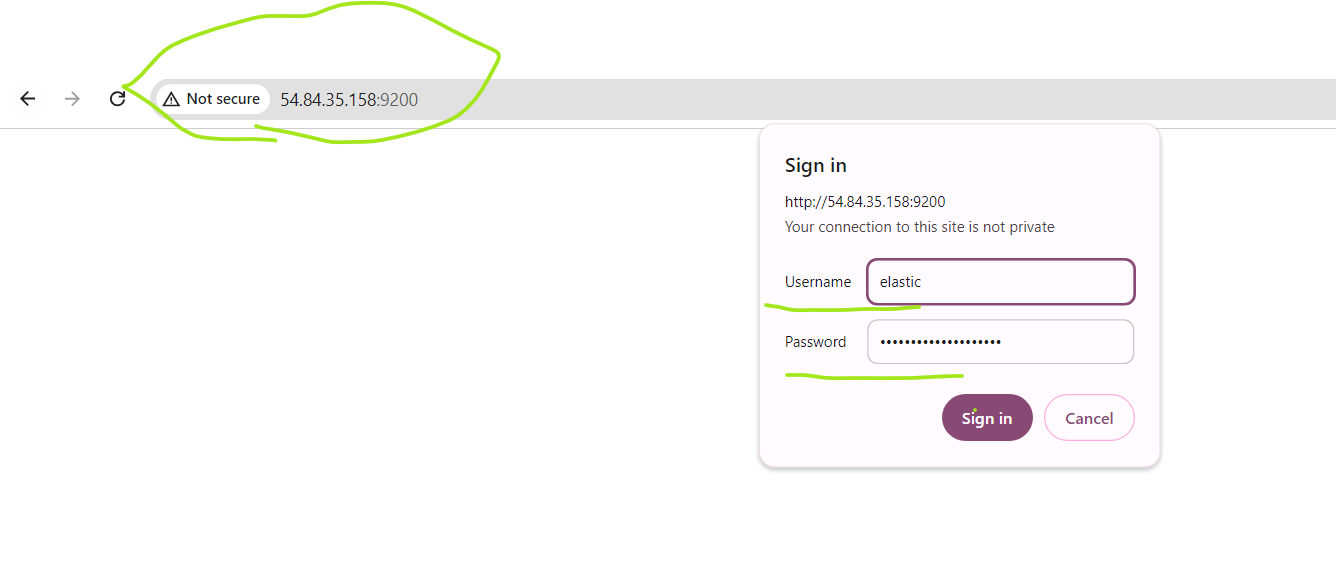
PASSWORD elastic = 5Tm0Vr3n950vghgzNY06

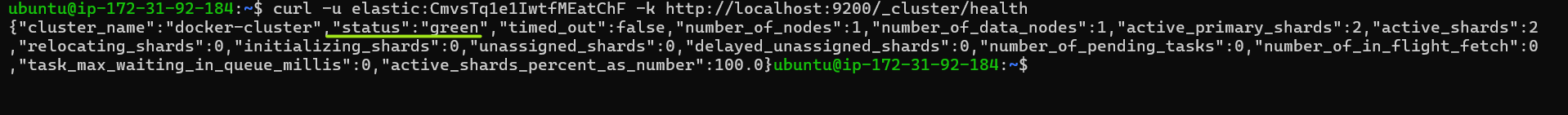
* Then we can access elasticsearch on port **9200** using the **Public-IP** of the server where elasticsearch container is running and then login using **elastic** user and **elastic\_user** password.
* We can also check the cluster status using the below command.

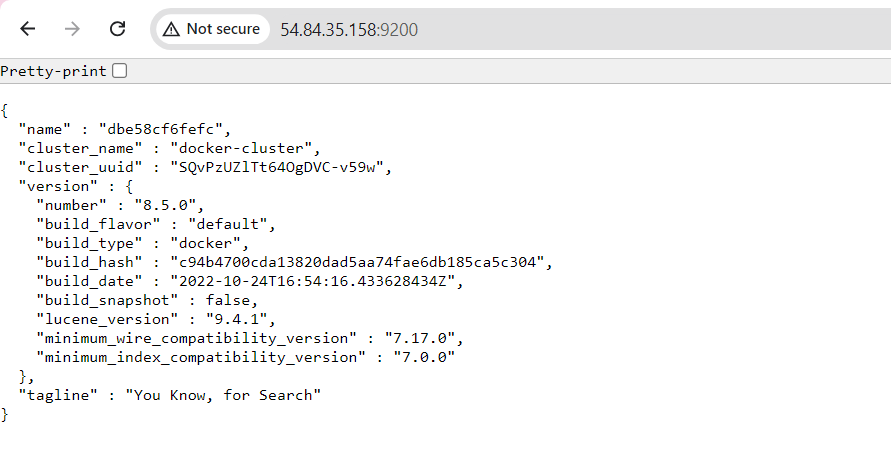
**Command:**

**[`curl -u elastic:elastic\_user\_password -k** [**http://localhost:9200/\_cluster/health`**](http://localhost:9200/_cluster/health%60)**]**

* If the above command returns the status as `green` then the elasticsearch is running fine.



****

****

**Installing Kibana:**

**Note:**

* In the below command replace the **kibana\_password** with the kibana user password which is generated above.

docker run -d --name kibana --net elk --restart unless-stopped \

-p 5601:5601 \

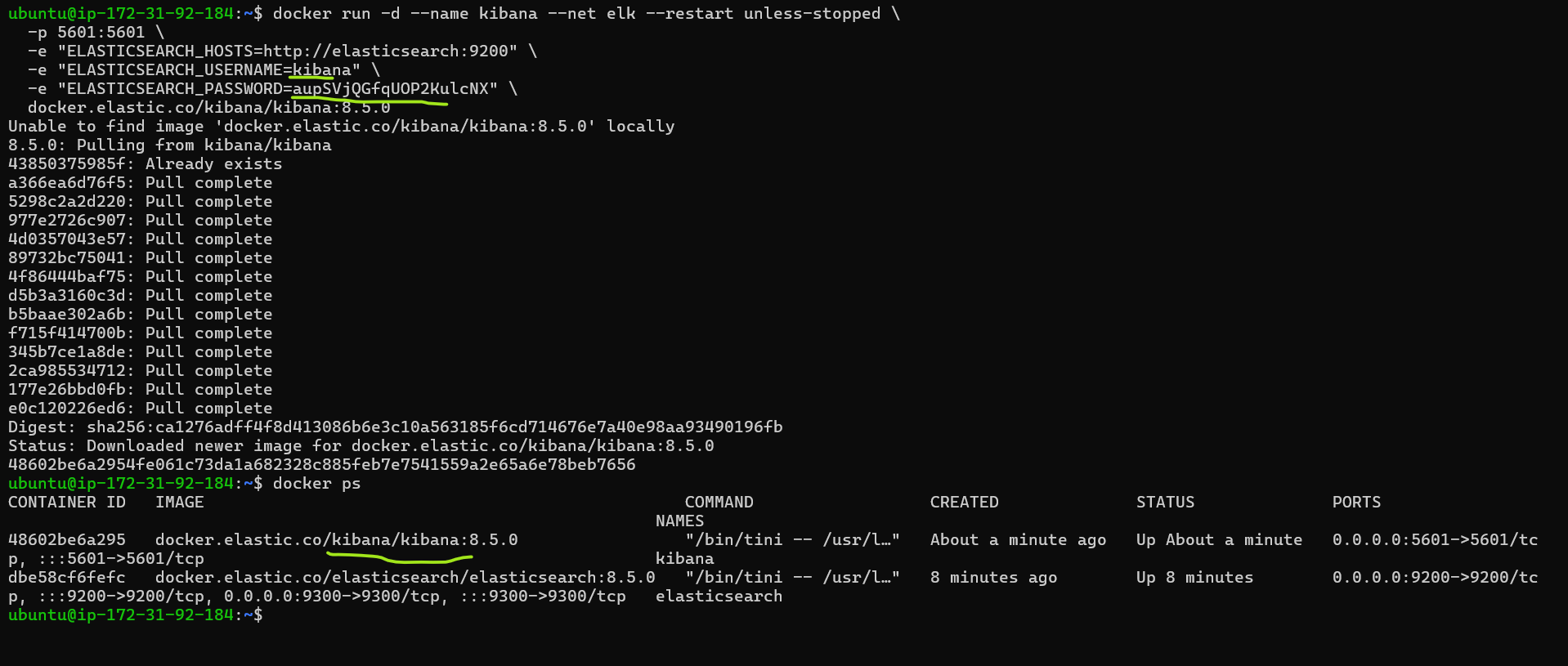
-e "ELASTICSEARCH\_HOSTS=http://elasticsearch:9200" \

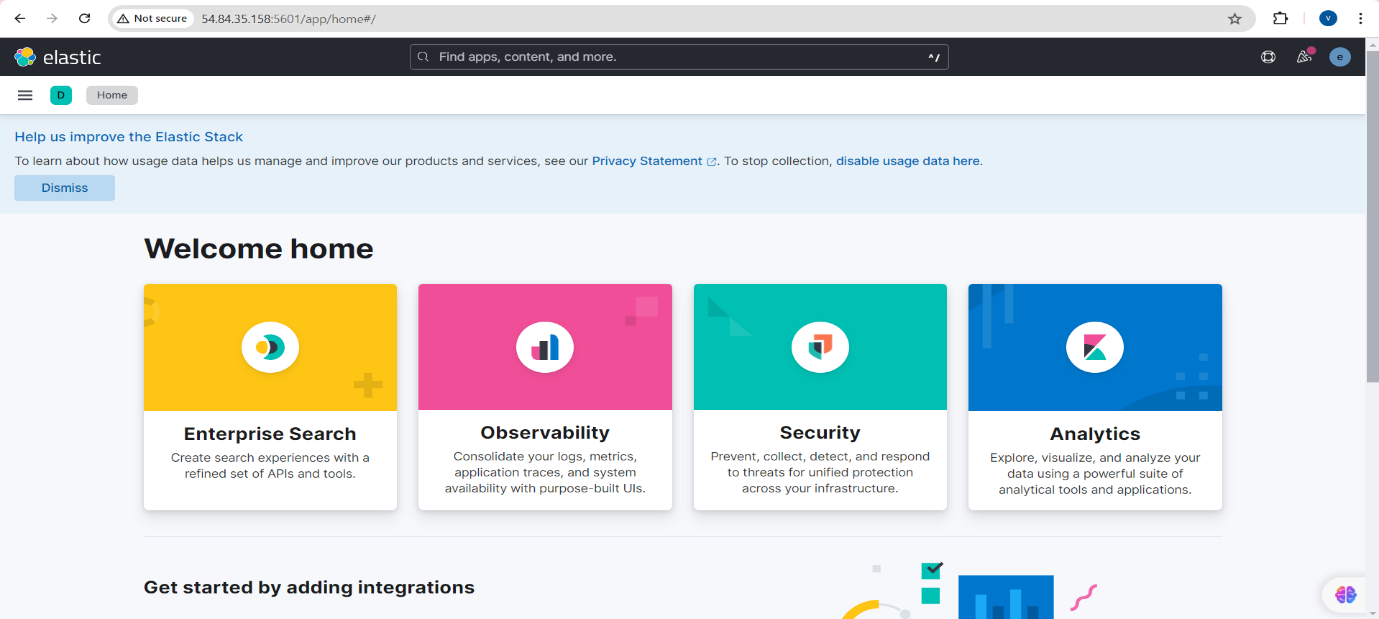
-e "ELASTICSEARCH\_USERNAME=kibana" \

-e "ELASTICSEARCH\_PASSWORD=<kibana\_password>" \

docker.elastic.co/kibana/kibana:8.5.0

* The above command will deploy kibana container on port **5601.**
* Then we can access kibana on port **5601** using the Public-IP of the server then use the kibana username and password to login. Here, we must need to use kibana password only, otherwise we can not access the kibana server, it will stay in server ready state.
* After login we can able to see kibana dashboard. Here, When it is asking for credentials to login we need to pass elastic user credentials because we are accessing kibana as a elastic user.



****

**Installing LogStash:**

* create a file with name **logstash.conf** in the current directory and add the below content to it to let know LogStash about the elasticsearch.

input {

beats {

port => 5044

}

}

filter {

grok {

match => { "message" => "%{SYSLOGLINE}" }

}

date {

match => [ "timestamp", "MMM d HH:mm:ss", "MMM dd HH:mm:ss" ]

}

}

output {

elasticsearch {

hosts => ["http://elasticsearch:9200"]

user => "elastic"

password => " CmvsTq1e1IwtfMEatChF "

index => "syslog"

}

stdout { codec => rubydebug }

}

* To create the file use **sudo vi logstash.conf** command and add the above content and save the file using **:wq!** instruction.
* In the above replace the **elastic\_user\_password** with the **elastic** user password which is generated in the 1st step.
* Then run the below command to deploy LogStash container.

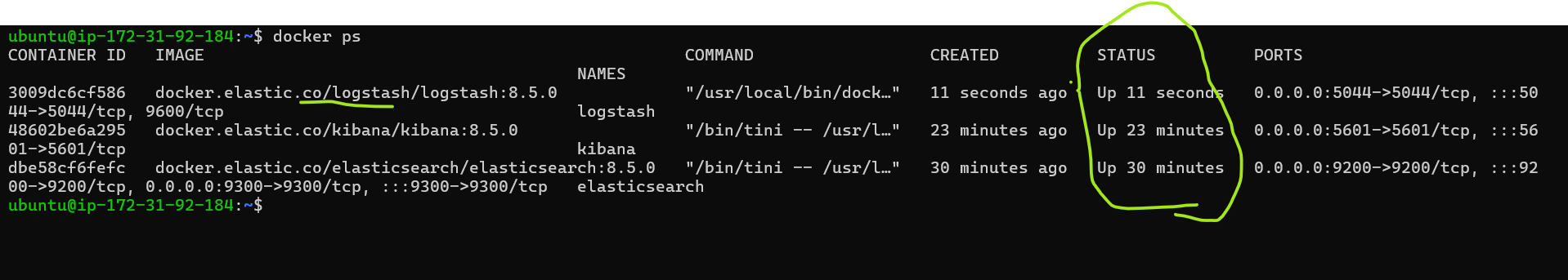
docker run -d --name logstash --net elk --restart unless-stopped \

-v $(pwd)/logstash.conf:/usr/share/logstash/pipeline/logstash.conf \

-p 5044:5044 \

docker.elastic.co/logstash/logstash:8.5.0

* After deploying it check the containers status using **docker ps** (or) **docker container ls** command, if this command shows all three containers status as running then ELK set up done.



**Monitoring EC2 instance using ELK:**

* To monitor an EC2 instance we need to install betas in the server which we wanted to monitor.

**What are beats in ELK:**

* Beats are lightweight data shippers designed to collect and send various types of operational data (logs, metrics, network data, etc.) from different machines to Logstash or Elasticsearch.
* Beats are part of the Elastic Stack, which includes Elasticsearch, Logstash, Kibana, and Beats.
* Each Beat is purpose-built to collect specific types of data, and they are often used to collect data from edge nodes and ship it to a central location for processing and analysis.

**Commonly used beats:**

* File Beat
* Metric Beat
* Heart Beat
* Packet Beat

Here I have taken **File Beat** and **Metric Beat** because,

* **File Beat** is a lightweight shipper for forwarding and centralizing log data. It monitors log files or locations you specify, collects log events, and forwards them to Elasticsearch or Logstash.
* **Metric Beat** collects metrics from the operating system and from services running on the server. It can monitor system-level metrics such as CPU, memory, and network usage, as well as application-level metrics.
* Here, in the server which I wanted to monitor, I'm installing the beats using docker, so I have created another **ubuntu-22.04** EC2 instance of size **t2.micro** and installed docker as above.

**Installing File Beat:**

* Firstly, we need to create a **filebeat.yml** file and insert the below data, which is essential to let know file beat about the elasticsearch details.

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/log/\*.log

- /path/to/your/logs/\*.log

output.elasticsearch:

hosts: ["http://54.84.35.158:9200/ "]

username: "elastic"

password: "elastic\_user\_password"

**Note:**

* Replace the elasticsearch host and elastic user password and save the file.
* To create and open the file with an editor use **sudo vi filebeat.yml** command and add the above content and then save it using **:wq!** instruction.
* Then use the below command which will deploy a file beat container.
* The above command will add the created **filebeat.yml** file to the container as a volume for the container.

docker run -d --name=filebeat \

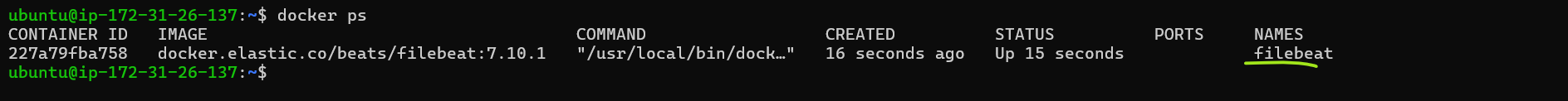
--user=root \

--volume="$(pwd)/filebeat.yml:/usr/share/filebeat/filebeat.yml:ro" \

--volume="/var/log:/var/log:ro" \

--volume="/path/to/your/logs:/path/to/your/logs:ro" \

docker.elastic.co/beats/filebeat:7.10.1 filebeat -e -strict.perms=false

****

**Installing Metric Beat:**

* Firstly, we need to create a **metricbeat.yml** file and insert the below data, which is essential to let know filebeat about the elasticsearch details.

**Note:**

* Replace the **elasticsearch host** and **elastic user password** and save the file.
* To create and open the file with an editor use **sudo vi metricbeat.yml** command.
* Add the below content and then save it using **:wq** instruction in **vi** editor.
* If you are using **nano** editor then use **sudo nano metricbeat.yml** command to create and open the file and save the file using **Ctrl+X** and **Y.**
* Add the below content in the **metricbeat.yml** file.

metricbeat.modules:

- module: system

metricsets:

- cpu

- load

- memory

- network

- process

- process\_summary

- socket\_summary

- uptime

enabled: true

period: 10s

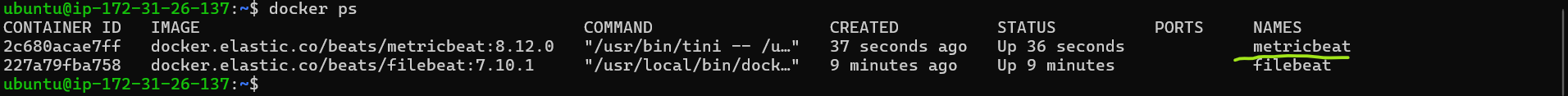
processes: ['.\*']

output.elasticsearch:

hosts: ["http://54.84.35.158:9200/ "]

username: "elastic"

password: " CmvsTq1e1IwtfMEatChF "



docker run -d --name=metricbeat \

--user=root \

--volume="$(pwd)/metricbeat.yml:/usr/share/metricbeat/metricbeat.yml:ro" \

--volume="/sys/fs/cgroup:/hostfs/sys/fs/cgroup:ro" \

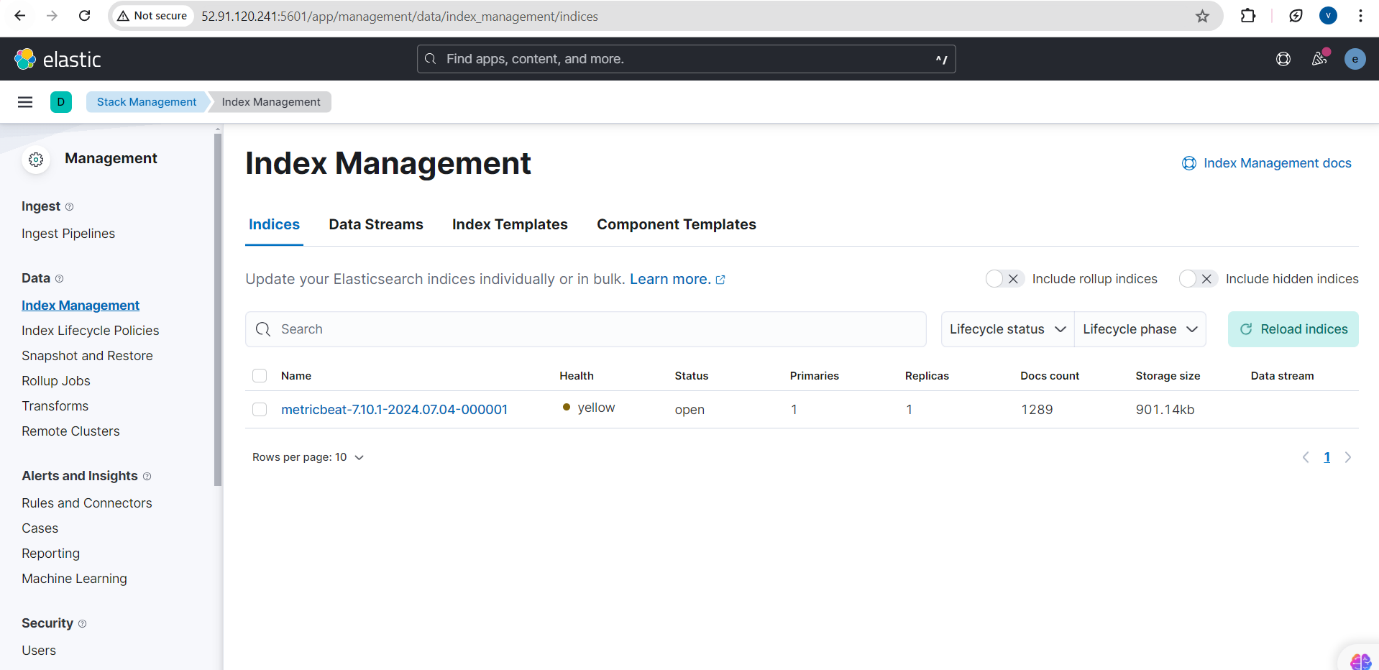
--volume="/proc:/hostfs/proc:ro" \

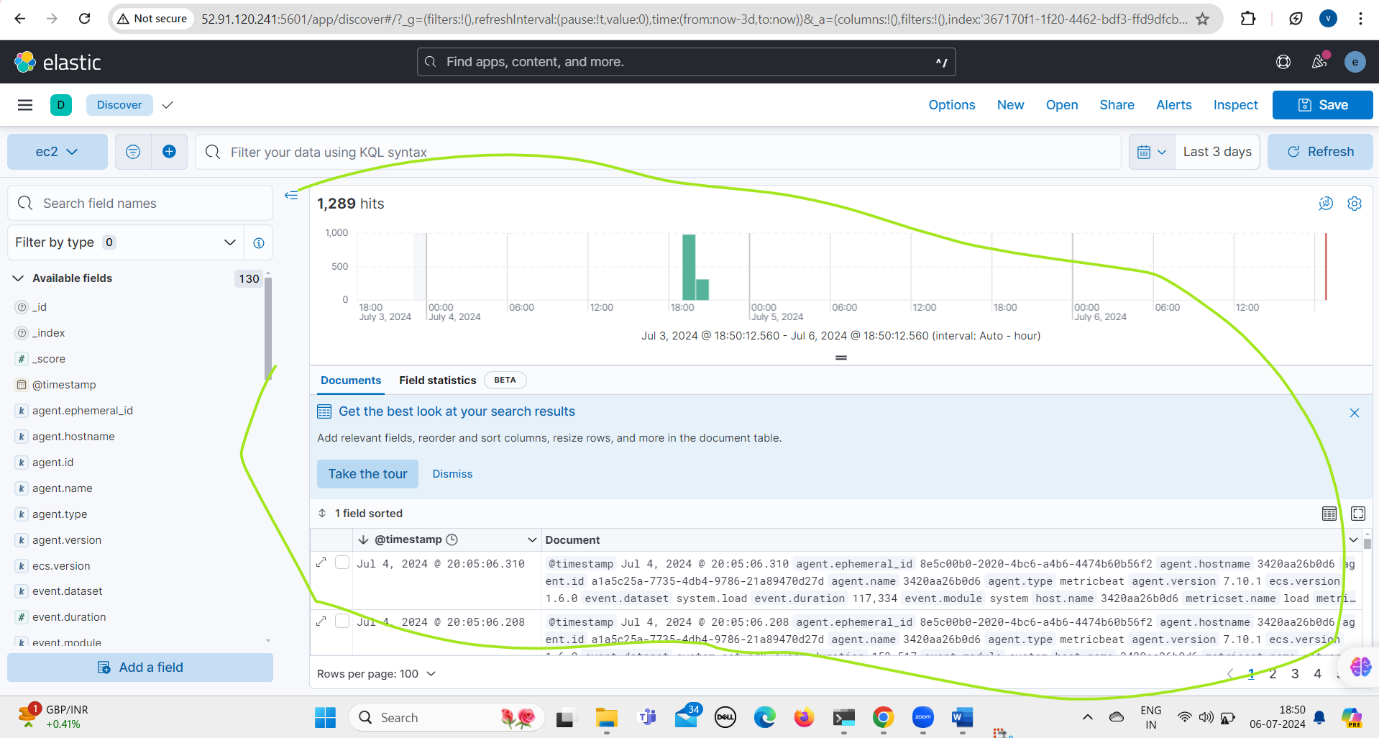
--volume="/:/hostfs:ro" \

docker.elastic.co/beats/metricbeat:8.12.0 \

-system.hostfs=/hostfs

* Then use the above command which will deploy a filebeat container. This command will add the created **metricbeat.yml** file to the container as a volume for the container.
* Then login to kibana dashboard and to Configure Index Patterns > Go to Management > Index Patterns.
* Create index patterns for **filebeat-\*** and **metricbeat-\***.
* Navigate to Discover in Kibana to start exploring logs and metrics collected from your EC2 instance.
* You can also create visualizations and dashboards based on the data.





* This is the process to setup ELK using docker and monitor an EC2 instance using beats.

**Thanks,**

**Lahari G**