FACULTAD DE CIENCIA Y TECNOLOGÍA

CENTRO REGIONAL DE USULUTÁN

INGENIERÍA EN SISTEMAS Y REDES INFORMÁTICAS

SEGURIDAD INFORMÁTICA

CICLO II – 2025

ACTIVIDAD:

TALLER SOBRE ELASTICSEARCH.

DOCENTE:

ING. TIMOTEA GUADALUPE MENJIVAR.

ESTUDIANTE:

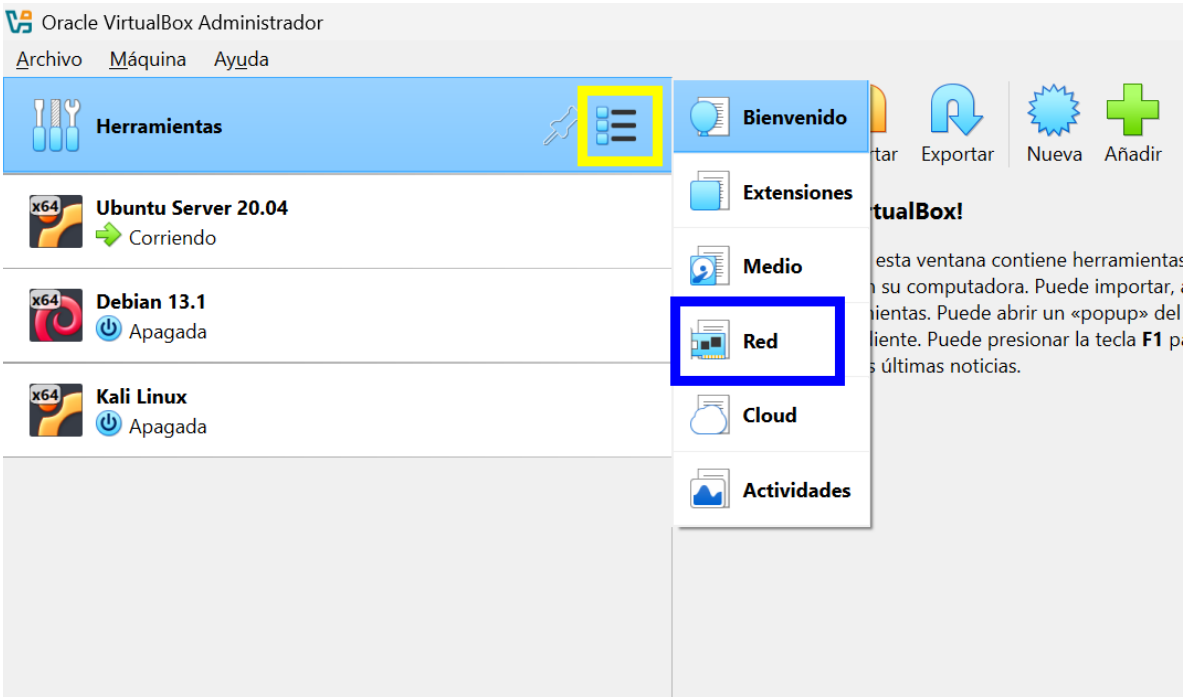
ROMEO ALEXANDER GARCIA CASTILLO USIS000313

FECHA DE ENTREGA:

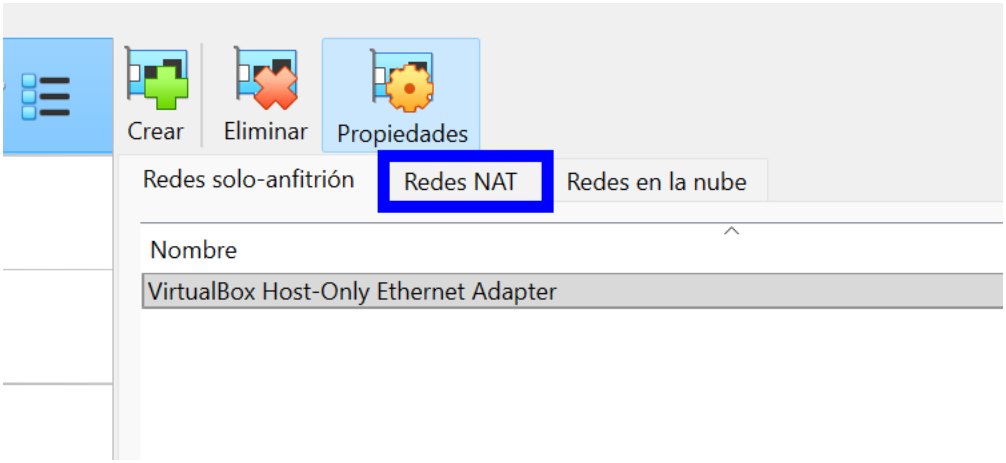
USULUTÁN, 13 DE NOVIEMBRE DE 2025.

Configuramos la red NAT

Para configurar la red nat, debemos ir a la parte del panel principal donde dice herramientas y presionar en el botón que aparece al lado derecho de la opción y seleccionar red:



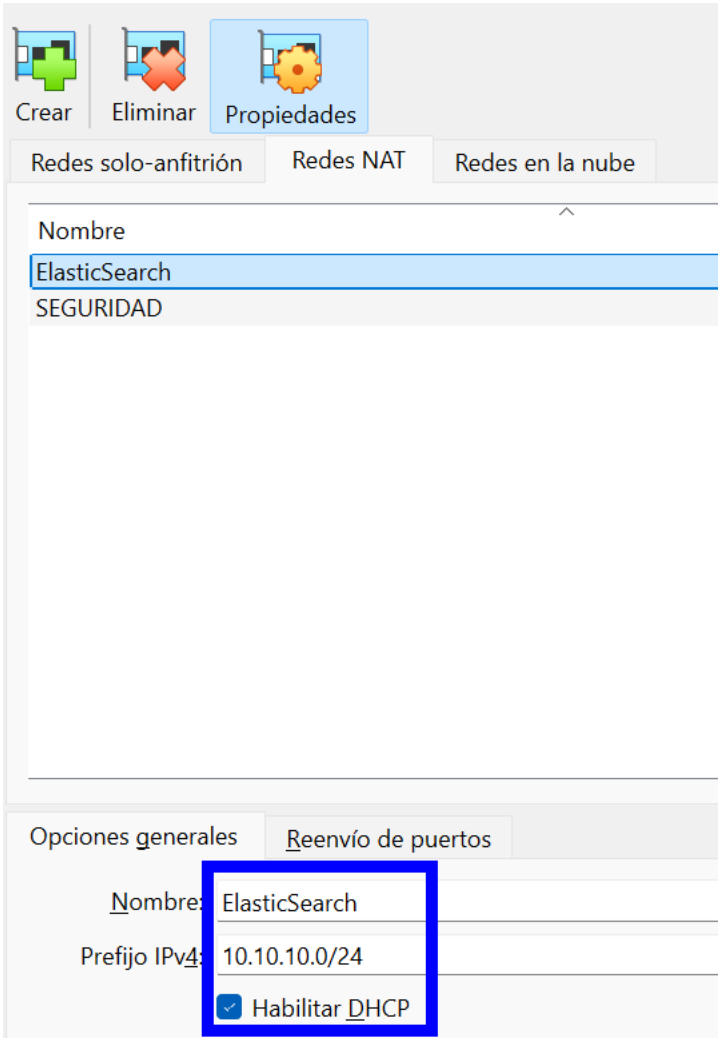
Una vez dentro debemos configurar la red, seleccionamos la que dice: “Redes NAT”:



Una vez dentro debemos configurar la red NAT de la siguiente manera para esta práctica:

Nombre de la red NAT: ElasticSearch

IP: 10.10.10.0/24

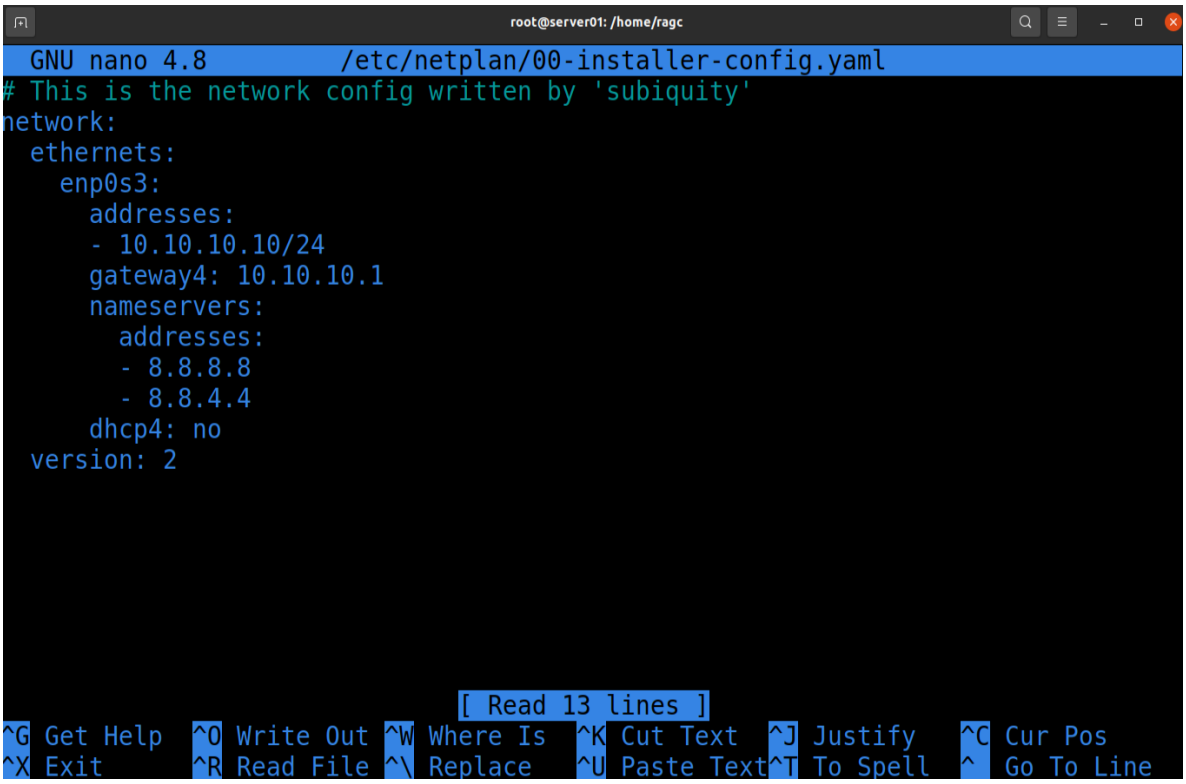


Configuramos una red estática en el servidor Ubuntu Server 20.04:

nano /etc/netplan/00-installer-config.yaml

```
root@server01:/home/ragc# nano /etc/netplan/00-installer-config.yaml
```

En la ruta anterior, debemos configurar de la siguiente manera y guardar la configuración con CTRL + O y luego salir con CTRL + X:



Para guardar la configuración realizada escribimos el comando:

netplan apply

Y si todo sale correctamente simplemente la terminal quedará limpia, pero si existe un error nos dirá el comando erróneo:

```
root@server01:/home/ragc# nano /etc/netplan/00-installer-config.yaml
root@server01:/home/ragc# netplan apply
root@server01:/home/ragc#
```

Verificamos que la configuración de red halla sido aplicada con el comando:

ip a

```
root@server01:/home/ragc# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP gr
oup default qlen 1000
    link/ether 08:00:27:23:41:5c brd ff:ff:ff:ff:ff:ff
    inet 10.10.10.10/24 brd 10.10.10.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe23:415c/64 scope link
       valid_lft forever preferred_lft forever
root@server01:/home/ragc#
```

Vamos actualizar el sistema:

apt-get update && apt-get upgrade -y

```
root@server01:/home/ragc# apt-get update && apt-get upgrade -y
```

Instalamos dependencias necesarias:

sudo apt install apt-transport-https curl gnupg -y

```
root@server01:/home/ragc# sudo apt install apt-transport-https curl gnupg -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
curl is already the newest version (7.68.0-1ubuntu2.25).
curl set to manually installed.
gnupg is already the newest version (2.2.19-3ubuntu2.5).
gnupg set to manually installed.
apt-transport-https is already the newest version (2.0.11).
The following packages were automatically installed and are no longer required:
  libfwupdplugin1 libxmlb1
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@server01:/home/ragc#
```

Agregamos la clave GPG de Elastic.

wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo gpg --dearmor -o /usr/share/keyrings/elasticsearch-keyring.gpg

```
root@server01:/home/ragc# wget -qO - https://artifacts.elastic.co/GPG-KEY-elasti
csearch | sudo gpg --dearmor -o /usr/share/keyrings/elasticsearch-keyring.gpg
root@server01:/home/ragc#
```

NOTA:

Tenemos que saber que, aunque no muestra progreso y solo nos muestra la terminal lista para seguir ingresando comandos, esto es completamente normal, debido a que:

wget -qO: Descarga la clave GPG desde el sitio de Elastic (sin mostrar progreso por el parámetro -q).

El símbolo **|** (nombrado “Pipe”): **envía esa clave directamente** como entrada al siguiente comando.

sudo gpg –dearmor: Convierte la clave a formato binario (.gpg) y la guarda en: /usr/share/keyrings/elasticsearch-keyring.gpg.

El **-q** (quiet) en wget y el uso de **--dearmor** no generan texto de confirmación.

Omitiremos el paso de: **instalación por APT:**

Instalacion por APT

sudo apt-get install apt-transport-https

Esto es por que esta línea sirve para **permitir que APT (el gestor de paquetes de Ubuntu/Debian)** pueda **descargar paquetes desde**

repositorios HTTPS (como el de Elastic). Pero **Ubuntu 20.04 ya incluye** apt-transport-https **por defecto.**

Esto lo hace desde versiones recientes, APT ya soporta HTTPS nativamente, por lo que este paquete ya viene instalado.

Si nosotros queremos comprobar lo mencionado anteriormente, basta con ejecutar el comando siguiente:

dpkg -l | grep apt-transport-https

```
root@server01:/home/ragc# dpkg -l | grep apt-transport-https
ii  apt-transport-https                      2.0.11
    all          transitional package for https support
root@server01:/home/ragc#
```

Mostrandonos:

ii → el paquete **ya está instalado correctamente.**

apt-transport-https → el nombre del paquete.

2.0.11 → versión instalada.

Transitional package for https support → Es un paquete de transición (Ubuntu 20.04 y posteriores ya traen HTTPS integrado en APT).

Agregamos el repositorio de Elasticsearch.

echo "deb [signed-by=/usr/share/keyrings/elasticsearch-keyring.gpg] https://artifacts.elastic.co/packages/9.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-9.x.list

```
root@server01:/home/ragc# echo "deb [signed-by=/usr/share/keyrings/elasticsearch
-keyring.gpg] https://artifacts.elastic.co/packages/9.x/apt stable main" | sudo
tee /etc/apt/sources.list.d/elastic-9.x.list
deb [signed-by=/usr/share/keyrings/elasticsearch-keyring.gpg] https://artifacts.
elastic.co/packages/9.x/apt stable main
root@server01:/home/ragc#
```

NOTA:

Debido al comando **tee,** no imprime mensaje de éxito. Solo genera un mensaje si hay error, en este caso como no hubo errores, se queda en silencio. Lo que indica que **se ejecutó correctamente.**

Verificamos que se creó correctamente:

Ejecutamos el comando:

cat /etc/apt/sources.list.d/elastic-9.x.list

```
root@server01:/home/ragc# cat /etc/apt/sources.list.d/elastic-9.x.list
```

Y nos debe salir la línea siguiente:

```
root@server01:/home/ragc# cat /etc/apt/sources.list.d/elastic-9.x.list
deb [signed-by=/usr/share/keyrings/elasticsearch-keyring.gpg] https://artifacts.
elastic.co/packages/9.x/apt stable main
root@server01:/home/ragc#
```

Confirmando con este resultado que **el archivo del repositorio se creó correctamente** y que contiene la línea exacta que debía tener. Esto significa que mi **Ubuntu server ya tiene agregado el repositorio oficial de elasticsearch.**

INSTALANDO ELASTICSEARCH.

Con el siguiente comando vamos a actualizar la lista de paquetes incluyendo el nuevo repositorio de Elastic y descargaremos e instalaremos Elasticsearch en nuestro sistema.

sudo apt-get update && sudo apt-get install elasticsearch -y

```
root@server01:/home/ragc# sudo apt-get update && sudo apt-get install elasticsearch -y
Hit:1 http://sv.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 https://packages.wazuh.com/4.x/apt stable InRelease
Get:3 https://artifacts.elastic.co/packages/9.x/apt stable InRelease [3.248 B]
Get:4 http://sv.archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB]
Hit:5 http://sv.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:6 http://sv.archive.ubuntu.com/ubuntu focal-security InRelease
Get:7 https://artifacts.elastic.co/packages/9.x/apt stable/main amd64 Packages [19,6 k
B]
Fetched 151 kB in 2s (63,8 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libfwupdplugin1 libxmlb1
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  elasticsearch
0 upgraded, 1 newly installed, 0 to remove and 1 not upgraded.
Need to get 687 MB of archives.
After this operation, 1.316 MB of additional disk space will be used.
Get:1 https://artifacts.elastic.co/packages/9.x/apt stable/main amd64 elasticsearch am
d64 9.2.0 [687 MB]
```

Una vez que termine la instalación debemos ver que se genere la contraseña de elastic.

The generated password for the elastic built-in superuser is : uNtrHMRcgcC8aBf0DwIv

```
-------------------------- Security autoconfiguration information ------------------
-----------

Authentication and authorization are enabled.
TLS for the transport and HTTP layers is enabled and configured.

The generated password for the elastic built-in superuser is : uNtrHMRcgcC8aBf0DwIv

If this node should join an existing cluster, you can reconfigure this with
'/usr/share/elasticsearch/bin/elasticsearch-reconfigure-node --enrollment-token <token
-here>'
```

Ahora debemos realizar los pasos que nos pide el instalador realizar:

```
--------------------------------------------------------------------------------
-----------
### NOT starting on installation, please execute the following statements to configure
 elasticsearch service to start automatically using systemd
 sudo systemctl daemon-reload
 sudo systemctl enable elasticsearch.service
### You can start elasticsearch service by executing
 sudo systemctl start elasticsearch.service
root@server01:/home/ragc#
```

1. Le decimos al sistema que vuelva a leer todos los archivos de configuración de servicios:

   sudo systemctl daemon-reload

```
root@server01:/home/ragc# sudo systemctl daemon-reload
root@server01:/home/ragc#
```

2. Le decimos que inicie elasticsearch automáticamente cada vez que encendamos el sistema.

   sudo systemctl enable elasticsearch.service

```
root@server01:/home/ragc# sudo systemctl enable elasticsearch.service
Created symlink /etc/systemd/system/multi-user.target.wants/elasticsearch.service → /l
ib/systemd/system/elasticsearch.service.
root@server01:/home/ragc#
```

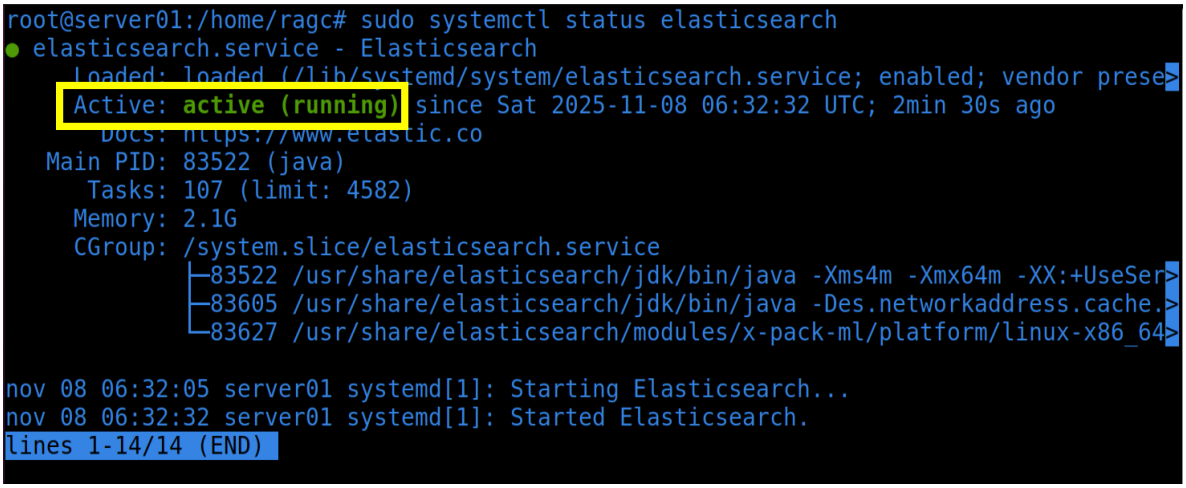
3. Iniciamos el servicio en este momento:

   sudo systemctl start elasticsearch.service

```
root@server01:/home/ragc# sudo systemctl start elasticsearch.service
root@server01:/home/ragc#
```

4. Verificamos el estado actual del servicio:

   sudo systemctl status elasticsearch



NOTA: Para salirnos de esa opción, solo presionamos la letra “q”.

Probando la conexión con cURL.

NOTA:

- Elasicsearch tiene su **propia API REST**, que escucha en el puerto **9200.**

AHORA VAMOS A ASEGURARNOS DE QUE RESPONDE EN EL PUERTO 9200 (POR HTTPS).

Vamos a ejecutar el siguiente comando, utilizando la contraseña generada:

curl -u elastic:'uNtrHMRcgcC8aBf0Dwlv' -k https://localhost:9200

El resultado obtenido nos indica que el servidor ElasticSearch está funcionando correctamente y respondiendo sin errores.

Elasticsearch **ya está activo, autenticado y listo para conectarse con Kibana.**

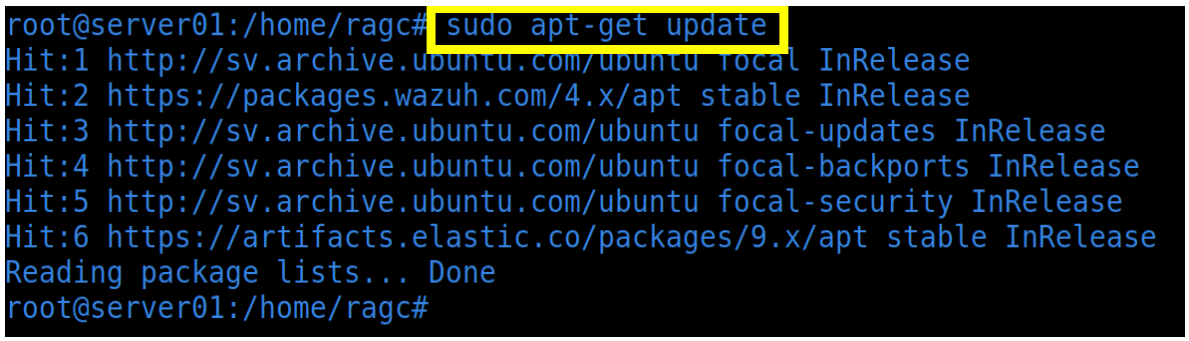
INSTALACIÓN DE KIBANA.

Instalar kibana y dejarlo activo para luego conectarlo con Elasticsearch.

Actualizamos los repositorios.

Esto permite que el sistema reconozca el repositorio de Elastic que ya se agrego durante la instalación de Elasticsearch.

sudo apt-get update



Instalamos kibana.

Lo que haremos es instalar kibina desde el mismo repositorio oficial de Elastic y lo instalará como un servicio en el sistema.

sudo apt-get install kibana -y

```
root@server01:/home/ragc# sudo apt-get install kibana -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libfwupdplugin1 libxmlb1
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  kibana
0 upgraded, 1 newly installed, 0 to remove and 1 not upgraded.
Need to get 360 MB of archives.
After this operation, 1.157 MB of additional disk space will be used.
Get:1 https://artifacts.elastic.co/packages/9.x/apt stable/main amd64 kibana amd
64 9.2.0 [360 MB]
Fetched 360 MB in 40s (8.898 kB/s)
Selecting previously unselected package kibana.
(Reading database ... 267824 files and directories currently installed.)
Preparing to unpack .../kibana_9.2.0_amd64.deb ...
Unpacking kibana (9.2.0) ...
Setting up kibana (9.2.0) ...
Creating kibana group... OK
Creating kibana user... OK
Created Kibana keystore in /etc/kibana/kibana.keystore
root@server01:/home/ragc#
```

Verificamos que el paquete se haya instalado correctamente, para ello usamos el comando:

dpkg -l | grep kibana

```
root@server01:/home/ragc# dpkg -l | grep kibana
ii  kibana                                       9.2.0
    amd64        Explore and visualize your Elasticsearch data
root@server01:/home/ragc#
```

Obteniendo los siguientes resultados:

ii → Significa que el paquete esta instalado y configurado correctamente.

kibana 9.2.0 → Es la versión actual que coincide con la de nuestro Elasticsearch (Perfecto).

Amd64 → Indica la arquitectura de 64 bits.

Descripción → Confirma que es el componente de visualización de Elasticsearch.

Habilitamos Kibana para que inicie automáticamente.

sudo systemctl enable kibana.service

```
root@server01:/home/ragc# sudo systemctl enable kibana.service
Created symlink /etc/systemd/system/multi-user.target.wants/kibana.service → /li
b/systemd/system/kibana.service.
root@server01:/home/ragc#
```

Iniciamos Kibana.

sudo systemctl start kibana.service

```
root@server01:/home/ragc# sudo systemctl start kibana.service
root@server01:/home/ragc#
```

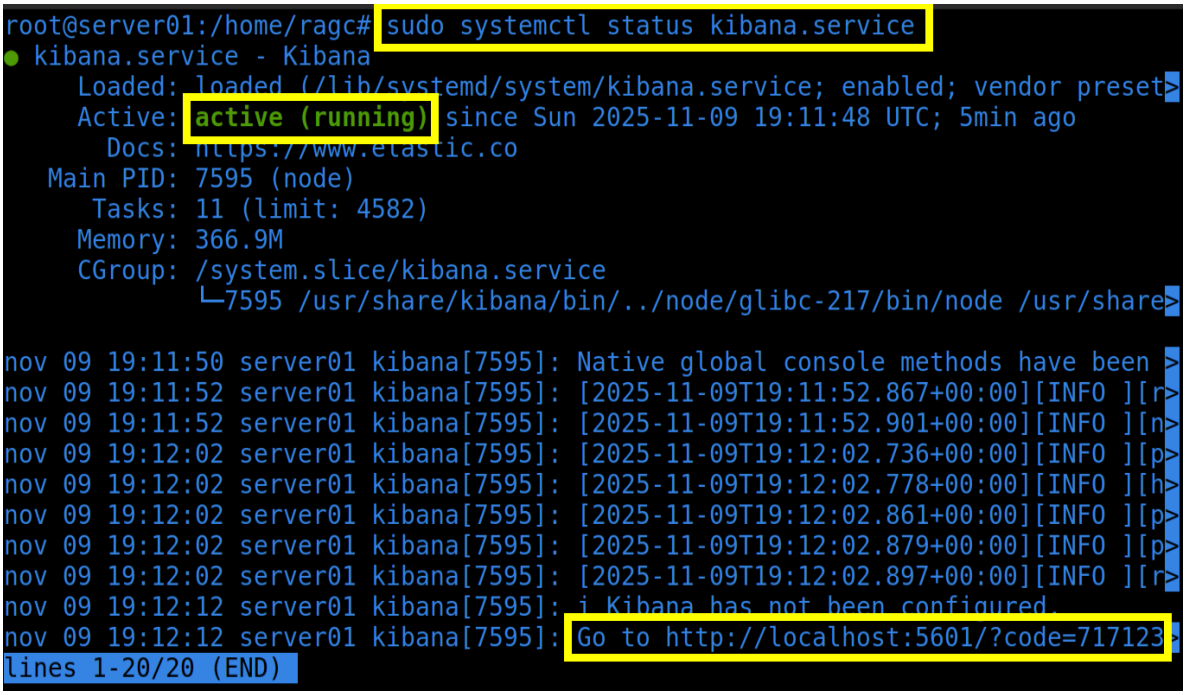
Verificamos el estado del servicio.

sudo systemctl status kibana.service

Podemos ver que el servicio de kibana está corriendo sin problema. Adicional a esto, vemos al final que **kibana ya levantó su servidor local** en el puerto **5601,** pero todavía **no esta configurado para acceder externamente (desde otro equipo).**

Por el momento solo podemos entrar desde el mismo servidor Ubuntu (localhost).

NOTA: Para salir presionamos la letra “q”.



CONFIGURAR ELASTICSEARCH PARA RECIBIR DATOS.

Abrimos el archivo de configuración, para ello ejecutamos el comando:

sudo nano /etc/elasticsearch/elasticsearch.yml

```
root@server01:/home/ragc# sudo nano /etc/elasticsearch/elasticsearch.yml
```

Estando en el archivo de configuración debemos modificar de la siguiente manera:

cluster.name: elasticprueba

node.name: server01

network.host: 10.10.10.10

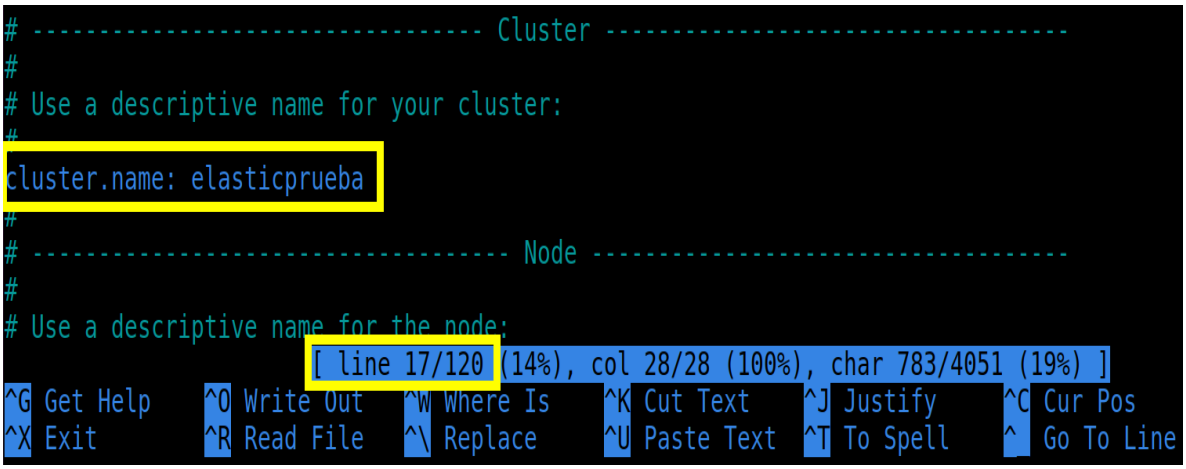
http.port: 9200

discovery.type: single-node

Quedando de la siguiente manera:

cluster.name: elasticprueba

Este es el nombre del grupo de servidores. Esta línea del archivo de configuración se modifica en esta parte, quedando de la siguiente manera:



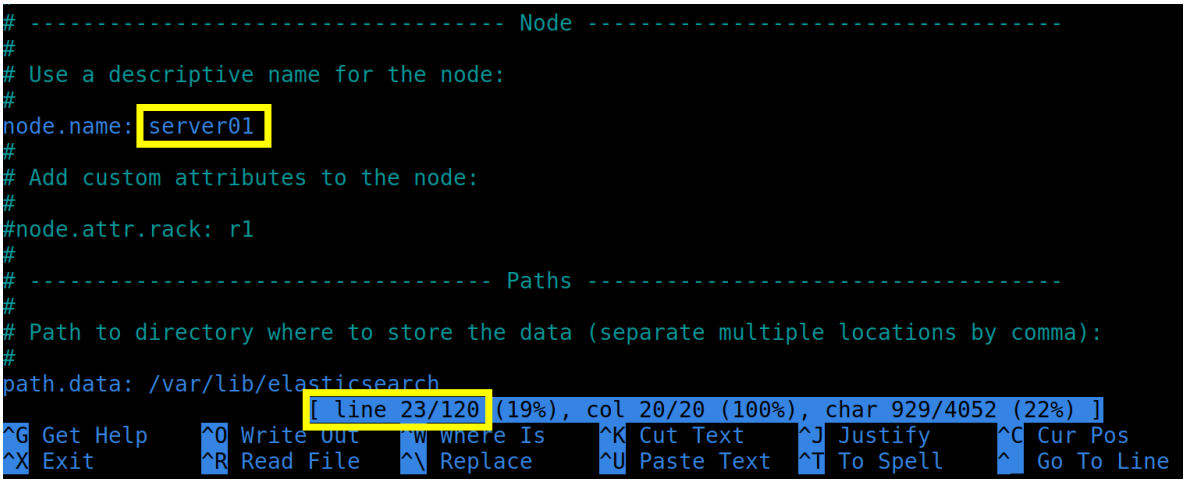
Siendo el número 17 de la fila, si nosotros queremos verificar el número de fila, primero debemos buscar en la herramienta nano esa parte del código, para ello solo presionamos **CTRL + W**, que es similar a utilizar la búsqueda en pdf o Word con control más F, luego posicionar el cursor sobre la línea que queremos verificar y presionar el comando **CTRL + C** y nos mostrará la posición actual del cursor, tanto en fila como columna.

node.name: server01

Parte opcional, pero recomendado.

Este identifica el nodo (cada nodo representa un servidor), se puede usar el hostname de nuestro servidor. De esta forma se reconoce el servidor que se esta trabajando dentro del sistema de elasticsearch y, por extensión, dentro de **Kibana** cuando empecemos a visualizar los datos.

Esta línea es la número 23 de la configuración.



network.host

Es la ip estática de nuestro servidor. Siendo la misma en este caso la que se configuro anteriormente.

```
# ---------------------------------- Network -----------------------------------
#
# By default Elasticsearch is only accessible on localhost. Set a different
# address here to expose this node on the network:
#
network.host: 10.10.10.10
#
```

http.port

En este se configura el puerto por defecto de Elasticsearch el cual es el puerto 9200.

```
# By default Elasticsearch listens for HTTP traffic on the first free port it
# finds starting at 9200. Set a specific HTTP port here:
#
http.port: 9200
#
# For more information, consult the network module documentation.
#
```

```
  GNU nano 4.8                        /etc/elasticsearch/elasticsearch.yml
# Enable security features
xpack.security.enabled: true

xpack.security.enrollment.enabled: true

# Enable encryption for HTTP API client connections, such as Kibana, Logstash, and Agents
xpack.security.http.ssl:
  enabled: true
  keystore.path: certs/http.p12

# Enable encryption and mutual authentication between cluster nodes
xpack.security.transport.ssl:
  enabled: true
  verification_mode: certificate
  keystore.path: certs/transport.p12
  truststore.path: certs/transport.p12
# Create a new cluster with the current node only
# Additional nodes can still join the cluster later
cluster.initial_master_nodes: ["server01"]
# Allow HTTP API connections from anywhere
# Connections are encrypted and require user authentication
http.host: 10.10.10.10

# Allow other nodes to join the cluster from anywhere
# Connections are encrypted and mutually authenticated
#transport.host: 0.0.0.0
```

Guardamos la configuración con CTRL + O y salimos con CTRL + X

Lo siguiente es reiniciar elasticsearch y verificar el estado del mismo.

Reiniciamos elasticsearch.

sudo systemctl restart elasticsearch

```
root@server01:/home/ragc# sudo systemctl restart elasticsearch
root@server01:/home/ragc#
```

Verificamos que este activo.

sudo systemctl status elasticsearch

```
root@server01:/home/ragc# sudo systemctl status elasticsearch
● elasticsearch.service - Elasticsearch
     Loaded: loaded (/lib/systemd/system/elasticsearch.service; enabled; vendor preset: enabled)
     Active: active (running) since Mon 2025-11-10 01:57:05 UTC; 9min ago
       Docs: https://www.elastic.co
   Main PID: 6778 (java)
      Tasks: 95 (limit: 4582)
     Memory: 2.3G
     CGroup: /system.slice/elasticsearch.service
             ├─6778 /usr/share/elasticsearch/jdk/bin/java -Xms4m -Xmx64m -XX:+UseSerialGC -Dcli.name=serve>
             ├─6858 /usr/share/elasticsearch/jdk/bin/java -Des.networkaddress.cache.ttl=60 -Des.networkadd>
             └─6880 /usr/share/elasticsearch/modules/x-pack-ml/platform/linux-x86_64/bin/controller

nov 10 01:56:39 server01 systemd[1]: Stopped Elasticsearch.
nov 10 01:56:39 server01 systemd[1]: Starting Elasticsearch...
nov 10 01:57:05 server01 systemd[1]: Started Elasticsearch.
lines 1-15/15 (END)
```

curl -u elastic:'uNtrHMRcgcC8aBf0Dwlv' -k https://10.10.10.10:9200

```
root@server01:/home/ragc# curl -u elastic:'uNtrHMRcgcC8aBf0DwIv' -k https://10.10.10.10:9200
{
  "name" : "server01",
  "cluster_name" : "elasticprueba",
  "cluster_uuid" : "4qzO6pV0TZKNQ4w53_NF4w",
  "version" : {
    "number" : "9.2.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "25d88452371273dd27356c98598287b669a03eae",
    "build_date" : "2025-10-21T10:06:21.288851013Z",
    "build_snapshot" : false,
    "lucene_version" : "10.3.1",
    "minimum_wire_compatibility_version" : "8.19.0",
    "minimum_index_compatibility_version" : "8.0.0"
  },
  "tagline" : "You Know, for Search"
}
root@server01:/home/ragc#
```

El resultado del JSON indica que **todo está funcionando correctamente.**

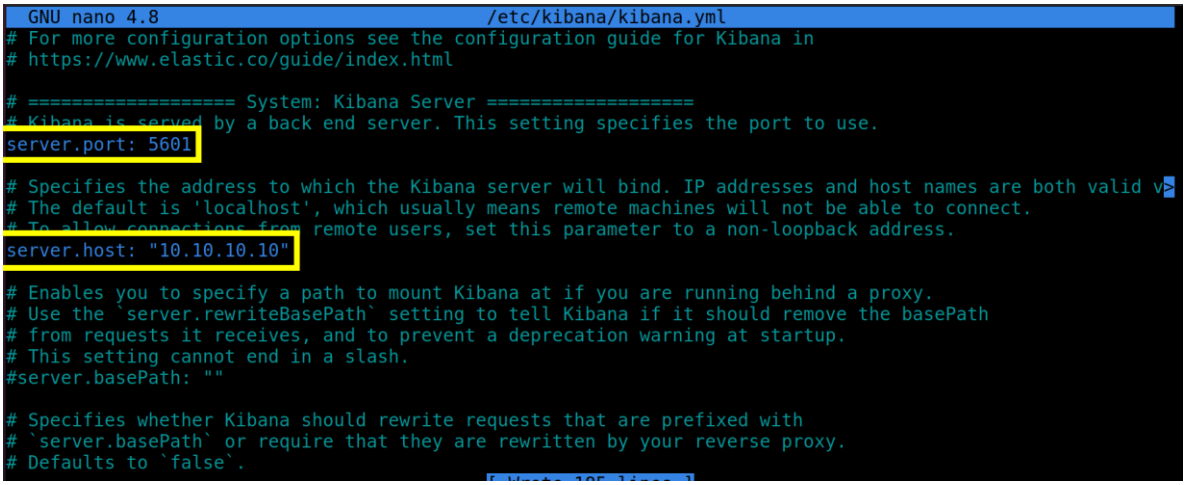
CONFIGURACIÓN DE KIBANA PARA MOSTRAR DATOS.

Vamos a configurar para que se mantenga el puerto 5601 y pondremos la misma ip de elasticsearch, luego iniciar/habilitar servicios y validar el acceso a la web.

Abrimos el config de Kibana.

sudo nano /etc/kibana/kibana.yml

```
root@server01:/home/ragc# sudo nano /etc/kibana/kibana.yml
```



Reiniciamos los servicios de elastic:

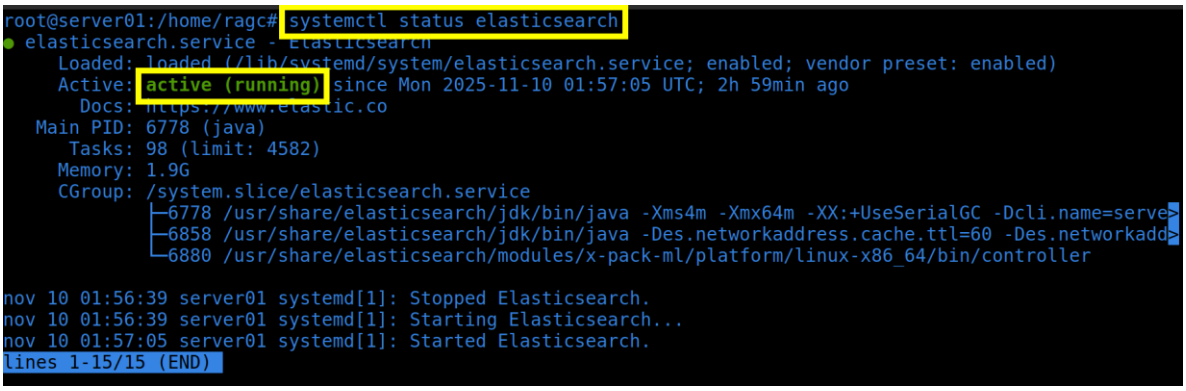
systemctl start elasticsearch

```
root@server01:/home/ragc# systemctl start elasticsearch
root@server01:/home/ragc#
```

systemctl enable elasticsearch

```
root@server01:/home/ragc# systemctl enable elasticsearch
root@server01:/home/ragc#
```

systemctl status elasticsearch



Iniciar los servicios de Kibana:

systemctl start kibana

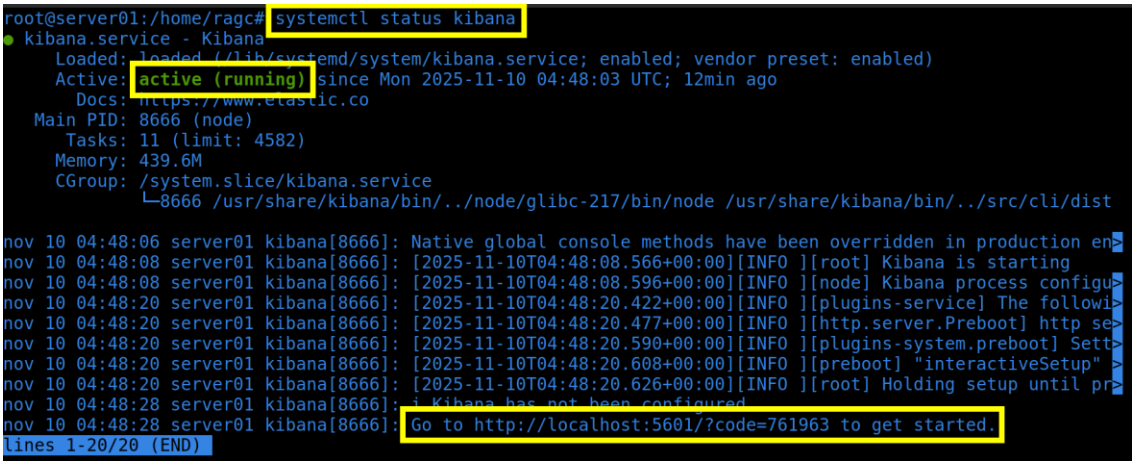
```
root@server01:/home/ragc# systemctl start kibana
root@server01:/home/ragc#
```

systemctl enable kibana

```
root@server01:/home/ragc# systemctl enable kibana
root@server01:/home/ragc#
```

systemctl status kibana

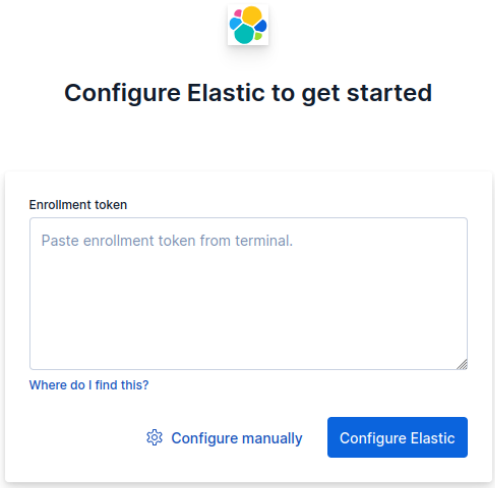
En el estatus de Kibana, estará el link para acceder desde el navegador, o también podemos acceder desde la dirección ipkibana:puertokibana **10.10.10.10:5601 o localhost:5601**



Siendo el link el siguiente: http://localhost:5601/?code=761963

Ahora vamos a obtener el nombre del clúster, el nombre del nodo, la versión de elasticsearch:

Al abrir en el navegador debemos entrar con la ip del servidor 10.10.10.10:5601



Ahora crearemos el token, primero se genera con el archivo **elasticsearch-create-enrollment-token**, que se encuentra en:

cd /usr/share/elasticsearch/bin/

```
ragc@server01:~$ cd /usr/share/elasticsearch/bin/
```

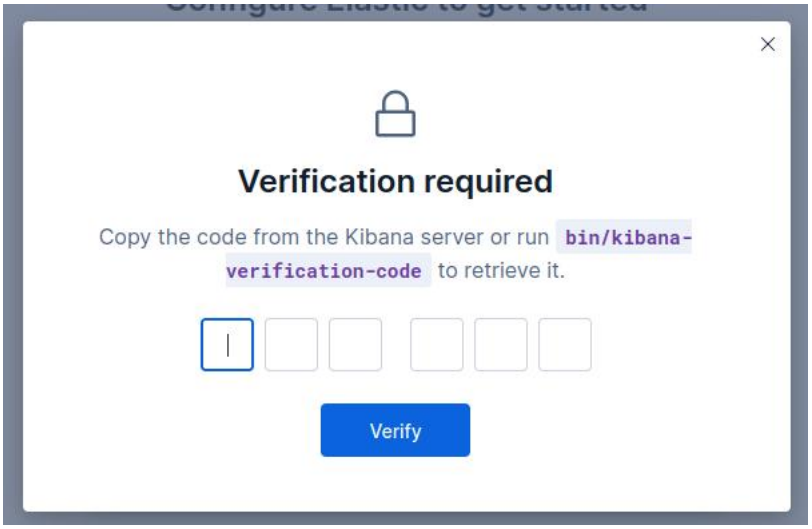
Y se ejecuta con:

./elasticsearch-create-enrollment-token --scope kibana

```
root@server01:/usr/share/elasticsearch/bin# ./elasticsearch-create-enrollment-token --scope kibana
eyJ2ZXIiOiI4LjE0LjAiLCJhZHIiOlsiMTAuMTAuMTAuMTA6OTIwMCJdLCJmZ3IiOiIzM2VjNjRkZDc1YTUzYzZmMTJkYWIzZjhmZWI5MGU
3ZTIxNmI3OWNhMzliYzg1OTc5MTBkNDVkZmRkNzliZjNlIiwia2V5Ijoic1ZSQmJKb0JVVUhTMzdFaVFCTko6ZTFwc1BCQ3NlMTlPX3VGSF
B2WHVBQSJ9
root@server01:/usr/share/elasticsearch/bin#
```

Este enrollment-token lo copiamos y lo pegamos en el navegador

Nos pide una verificación



Nos pasamos a la siguiente ruta:

cd /usr/share/kibana/bin

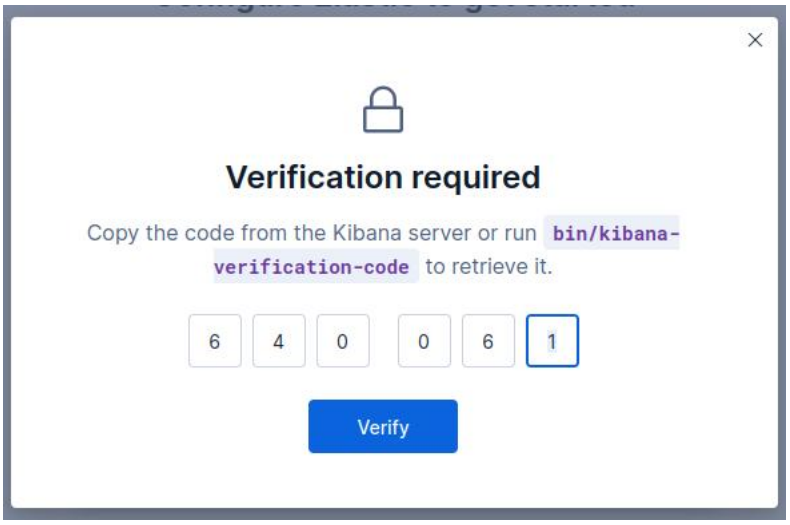
```
root@server01:/usr/share/elasticsearch/bin# cd /usr/share/kibana/bin
root@server01:/usr/share/kibana/bin#
```

Ahora ejecutamos.

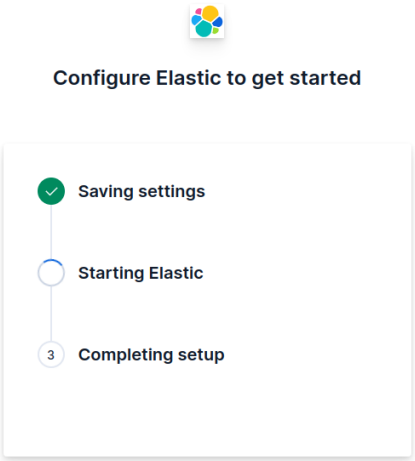
./kibana-verification-code

```
root@server01:/usr/share/kibana/bin# ./kibana-verification-code
```

```
root@server01:/usr/share/kibana/bin# ./kibana-verification-code
Your verification code is:  640 061
root@server01:/usr/share/kibana/bin#
```



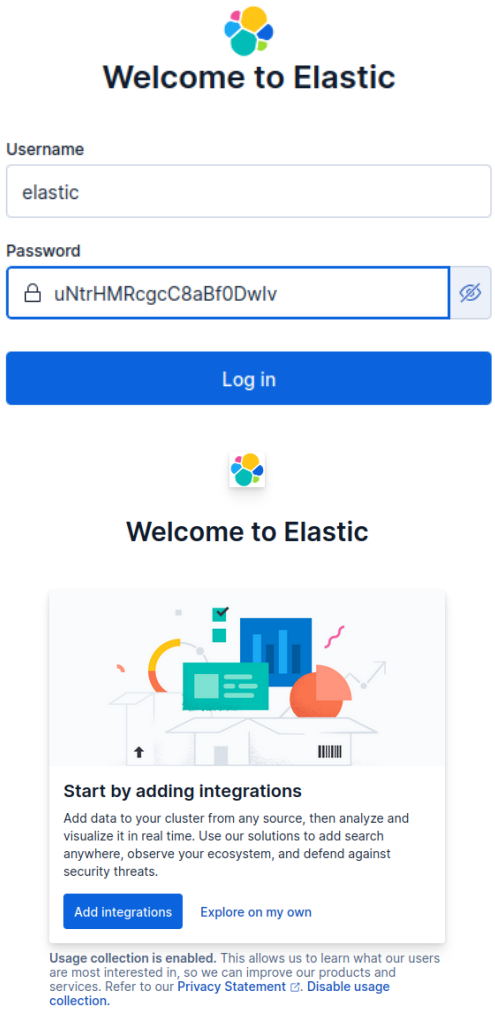
Y se iniciarán todos los servicios.



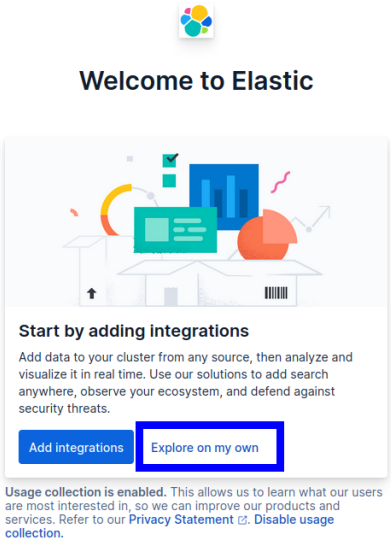
El usuario es superusuario creado por defecto, este es **elastic.**

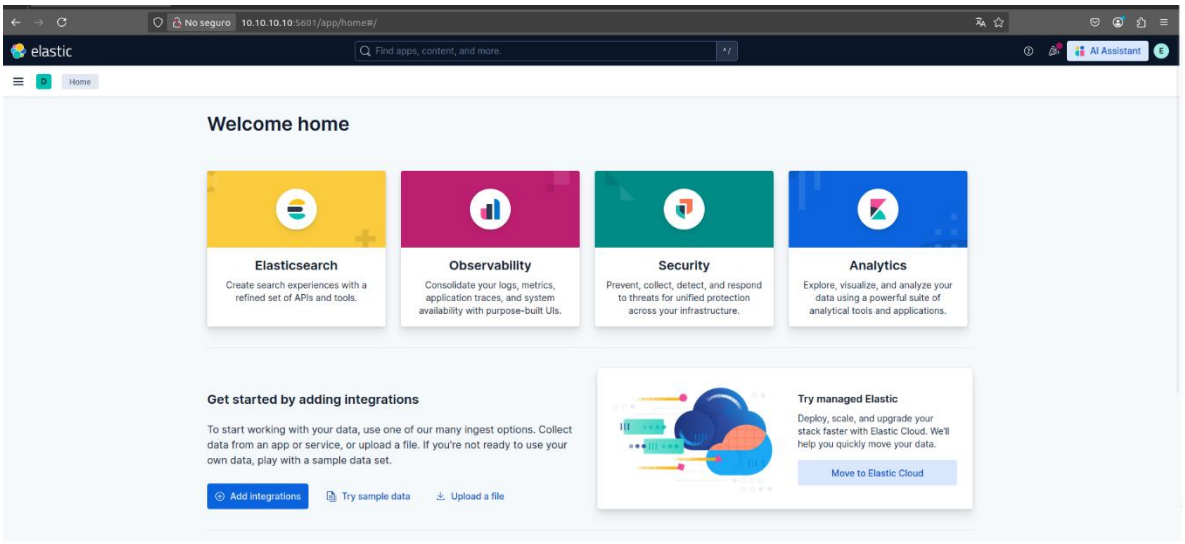
La contraseña es el que generamos al inicio.

```
1 Superusuario predeterminado: elastic
2
3 The generated password for the elastic built-in
  superuser is : uNtrHMRcgcC8aBf0DwIv
```

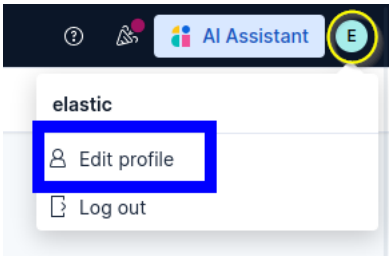


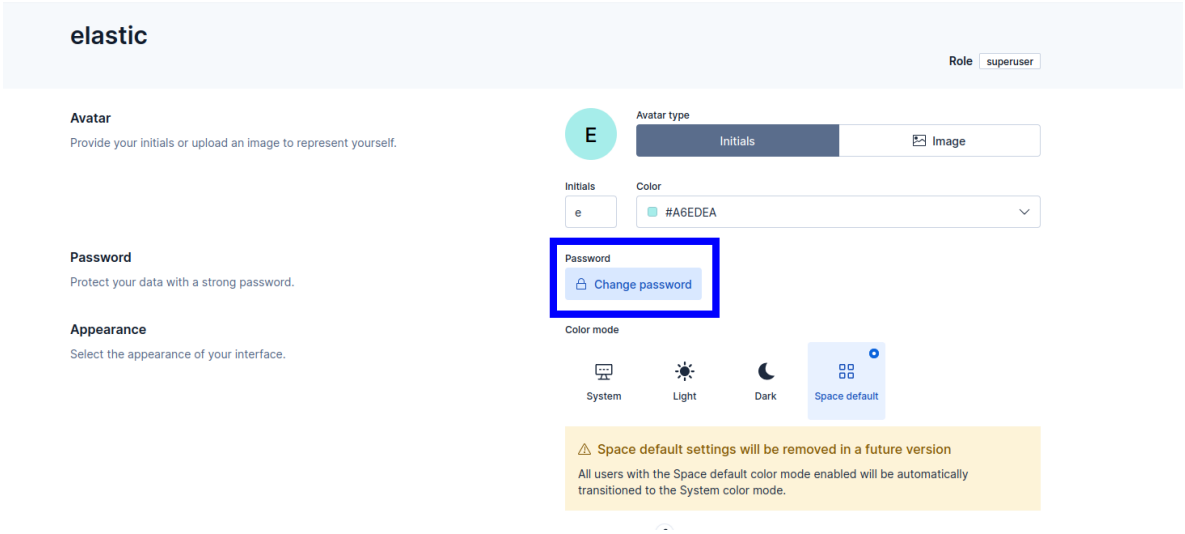
Tenemos que presionar el botón “Explore on my own” para pasar al **panel principal de kibana.**



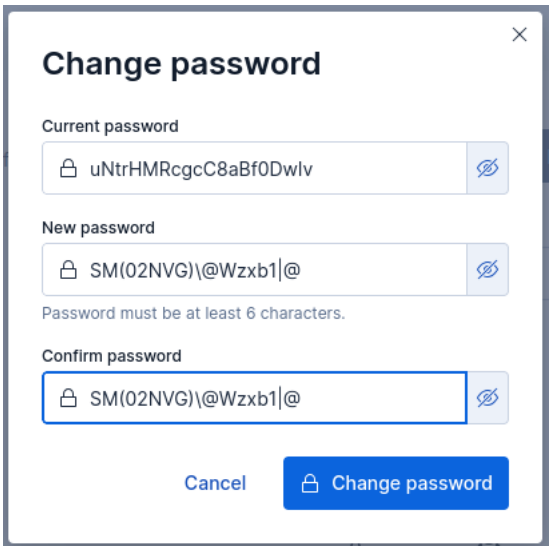


Ahora vamos a cambiar la contraseña, para ello debemos ir a **Edit profile,** en la esquina superior derecha.





Nos aparece una ventana para cambiar la contraseña:



Para ver la versión de elastic, lo hacemos con el siguiente comando:

curl -k -u elastic https://10.10.10.10:9200

```
root@server01:/home/ragc# curl -k -u elastic https://10.10.10.10:9200
Enter host password for user 'elastic':
{
  "name" : "server01",
  "cluster_name" : "elasticprueba",
  "cluster_uuid" : "4qzO6pV0TZKNQ4w53_NF4w",
  "version" : {
    "number" : "9.2.0",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "25d88452371273dd27356c98598287b669a03eae",
    "build_date" : "2025-10-21T10:06:21.288851013Z",
    "build_snapshot" : false,
    "lucene_version" : "10.3.1",
    "minimum_wire_compatibility_version" : "8.19.0",
    "minimum_index_compatibility_version" : "8.0.0"
  },
  "tagline" : "You Know, for Search"
}
root@server01:/home/ragc#
```

Para ver la licencia de elastic, lo hacemos con el siguiente comando:

curl -u elastic -X GET "https://10.10.10.10:9200/_license?pretty" -k

```
root@server01:/home/ragc# curl -u elastic -X GET "https://10.10.10.10:9200/_license?pretty" -k
Enter host password for user 'elastic':
{
  "license" : {
    "status" : "active",
    "uid" : "e2281317-b33d-4bf2-99f3-5bc044e8bd53",
    "type" : "basic",
    "issue_date" : "2025-11-08T06:32:34.595Z",
    "issue_date_in_millis" : 1762583554595,
    "max_nodes" : 1000,
    "max_resource_units" : null,
    "issued_to" : "elasticsearch",
    "issuer" : "elasticsearch",
    "start_date_in_millis" : -1
  }
}
root@server01:/home/ragc#
```

CONFIGURAMOS EL CLIENTE CON FILEBEAT.

Configuramos el cliente con filebeat para que envíe al servidor elastic los logs, el cliente y elastic deben estar bajo la misma red NAT.

Descargamos el instalador de Filebeat.

curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.18.3-amd64.deb

```
root@romeoDebian:/home/ragc# curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.
18.3-amd64.deb
```

sudo dpkg -i filebeat-8.18.3-amd64.deb

```
ragc@romeoDebian:~$ sudo dpkg -i filebeat-8.18.3-amd64.deb
[sudo] contraseña para ragc:
Seleccionando el paquete filebeat previamente no seleccionado.
(Leyendo la base de datos ... 142589 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar filebeat-8.18.3-amd64.deb ...
Desempaquetando filebeat (8.18.3) ...
Configurando filebeat (8.18.3) ...
```

sudo apt install rsyslog -y

```
ragc@romeoDebian:~$ sudo apt install rsyslog -y
Installing:
  rsyslog

Installing dependencies:
  libestr0  libfastjson4  liblognorm5

Paquetes sugeridos:
  rsyslog-doc       rsyslog-mongodb        rsyslog-hiredis     rsyslog-docker       | rsyslog-gnutls
  rsyslog-mysql     rsyslog-elasticsearch  rsyslog-snmp        rsyslog-clickhouse   rsyslog-gssapi
  | rsyslog-pgsql   rsyslog-kafka          rsyslog-kubernetes  rsyslog-openssl      rsyslog-relp

Summary:
  Upgrading: 0, Installing: 4, Removing: 0, Not Upgrading: 0
  Download size: 863 kB
  Space needed: 2,338 kB / 4,064 MB available
```

sudo nano /etc/filebeat/filebeat.yml

```
ragc@romeoDebian:~$ sudo nano /etc/filebeat/filebeat.yml
ragc@romeoDebian:~$
```

Comentamos las líneas:

filebeat.inputs

enabled: true

paths:

- /var/log/*.log

- /var/log/auth.log

- /var/log/syslog

```
  GNU nano 8.4                          /etc/filebeat/filebeat.yml
###################### Filebeat Configuration Example #########################

# This file is an example configuration file highlighting only the most common
# options. The filebeat.reference.yml file from the same directory contains all the
# supported options with more comments. You can use it as a reference.
#
# You can find the full configuration reference here:
# https://www.elastic.co/guide/en/beats/filebeat/index.html

# For more available modules and options, please see the filebeat.reference.yml sample
# configuration file.

# ============================== Filebeat inputs ===============================

#filebeat.inputs:

# Each - is an input. Most options can be set at the input level, so
# you can use different inputs for various configurations.
```

Configuramos las direcciones del servidor de elastic.

setup.kibana:

host: http://10.10.10.10:5601

```
# ================================== Kibana ===================================

# Starting with Beats version 6.0.0, the dashboards are loaded via the Kibana API.
# This requires a Kibana endpoint configuration.
setup.kibana:

  # Kibana Host
  # Scheme and port can be left out and will be set to the default (http and 5601)
  # In case you specify and additional path, the scheme is required: http://localhost:5601/path
  # IPv6 addresses should always be defined as: https://[2001:db8::1]:5601
  host: "http://10.10.10.10:5601"
```

Como no usaremos certificados, colocamos la siguiente línea para saltar el certificado en cada petición.

```
# ---------------------------- Elasticsearch Output ----------------------------
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["https://10.10.10.10:9200"]

  # Performance preset - one of "balanced", "throughput", "scale",
  # "latency", or "custom".
  preset: balanced

  # Protocol - either `http` (default) or `https`.
  #protocol: "https"

  # Authentication credentials - either API key or username/password.
  #api_key: "id:api_key"
  username: "elastic"
  password: "admin123"
  ssl.verification_mode: none
```

Probamos conexión.

- **sudo filebeat test config -c /etc/filebeat/filebeat.yml**
- **sudo filebeat test output**

```
ragc@romeoDebian:~$ sudo filebeat test config -c /etc/filebeat/filebeat.yml
Config OK
ragc@romeoDebian:~$ sudo filebeat test output
elasticsearch: https://10.10.10.10:9200...
  parse url... OK
  connection...
    parse host... OK
    dns lookup... OK
    addresses: 10.10.10.10
```

Desbloqueamos el puerto 9200, ya que es posible que este bloqueado por el firewall del servidor

En el servidor:

sudo ufw status

```
ragc@server01:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
443/tcp                    ALLOW       Anywhere
1514/tcp                   ALLOW       Anywhere
55000/tcp                  ALLOW       Anywhere
5601/tcp                   ALLOW       Anywhere
1515/tcp                   ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
443/tcp (v6)               ALLOW       Anywhere (v6)
1514/tcp (v6)              ALLOW       Anywhere (v6)
55000/tcp (v6)             ALLOW       Anywhere (v6)
5601/tcp (v6)              ALLOW       Anywhere (v6)
1515/tcp (v6)              ALLOW       Anywhere (v6)

ragc@server01:~$
```

El firewall esta activo y no se ve el 9200 en la lista, por lo tanto debemos agregarlo con la regla:

sudo ufw allow 9200/tcp

sudo ufw reload

```
ragc@server01:~$ sudo ufw allow 9200/tcp
Rule added
Rule added (v6)
ragc@server01:~$ sudo ufw reload
Firewall reloaded
ragc@server01:~$
```

Luego confirmamos:

sudo ufw status

```
ragc@server01:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
443/tcp                    ALLOW       Anywhere
1514/tcp                   ALLOW       Anywhere
55000/tcp                  ALLOW       Anywhere
5601/tcp                   ALLOW       Anywhere
1515/tcp                   ALLOW       Anywhere
9200/tcp                   ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
443/tcp (v6)               ALLOW       Anywhere (v6)
1514/tcp (v6)              ALLOW       Anywhere (v6)
55000/tcp (v6)             ALLOW       Anywhere (v6)
5601/tcp (v6)              ALLOW       Anywhere (v6)
1515/tcp (v6)              ALLOW       Anywhere (v6)
9200/tcp (v6)              ALLOW       Anywhere (v6)
```

```
ragc@romeoDebian:~$ sudo filebeat test output
[sudo] contraseña para ragc:
elasticsearch: https://10.10.10.10:9200...
  parse url... OK
  connection...
    parse host... OK
    dns lookup... OK
    addresses: 10.10.10.10
    dial up... OK
  TLS...
    security... WARN server's certificate chain verification is disabled
    handshake... OK
    TLS version: TLSv1.3
    dial up... OK
  talk to server... OK
  version: 9.2.0
ragc@romeoDebian:~$
```

HABILITAMOS EL MODULO SYSTEM EN EL CLIENTE. EN ESTE CASO DEBIAN 10.10.10.11

HABILITA EL MÓDULO SYSTEM.

sudo filebeat modules enable system

```
ragc@romeoDebian:~$ sudo filebeat modules enable system
Enabled system
ragc@romeoDebian:~$
```

sudo filebeat modules enable auditd

```
ragc@romeoDebian:~$ sudo filebeat modules enable auditd
Enabled auditd
ragc@romeoDebian:~$
```

CONFIGURAMOS EL MÓDULO SYSTEM.

sudo nano /etc/filebeat/modules.d/system.yml

```
ragc@romeoDebian:~$ sudo nano /etc/filebeat/modules.d/system.yml
```

```
  GNU nano 8.4                    /etc/filebeat/modules.d/system.yml *
# Module: system
# Docs: https://www.elastic.co/guide/en/beats/filebeat/8.18/filebeat-module-system.html

- module: system
  # Syslog
  syslog:
    enabled: true

    # Set custom paths for the log files. If left empty,
    # Filebeat will choose the paths depending on your OS.
    #var.paths:

    # Use journald to collect system logs
    #var.use_journald: false

  # Authorization logs
  auth:
    enabled: true
```

CONFIGURAMOS MODULO AUDIT.

sudo nano /etc/filebeat/modules.d/auditd.yml

```
ragc@romeoDebian:~$ sudo nano /etc/filebeat/modules.d/auditd.yml
```

```
  GNU nano 8.4                          /etc/filebeat/modules.d/auditd.yml
# Module: auditd
# Docs: https://www.elastic.co/guide/en/beats/filebeat/8.18/filebeat-module-auditd.html

- module: auditd
  log:
    enabled: true

    # Set custom paths for the log files. If left empty,
    # Filebeat will choose the paths depending on your OS.
    #var.paths:
```

REINICIAMOS KIBANA EN EL SERVIDOR DE ELASTICSEARCH.

sudo systemctl restart kibana

```
ragc@server01:~$ sudo systemctl restart kibana
ragc@server01:~$
```

REINICIAMOS LOS SERVICIOS DE FILEBEAT EN EL CLIENTE.

sudo systemctl enable filebeat

```
ragc@romeoDebian:~$ sudo systemctl enable filebeat
Synchronizing state of filebeat.service with SysV service script with /usr/lib/systemd/systemd-sysv-insta
ll.
Executing: /usr/lib/systemd/systemd-sysv-install enable filebeat
Created symlink '/etc/systemd/system/multi-user.target.wants/filebeat.service' → '/usr/lib/systemd/system
/filebeat.service'.
ragc@romeoDebian:~$
```

Este mensaje indica que Filebeat ya fue habilitado correctamente para que se inicie automáticamente cada vez que se inicie Debian.

sudo systemctl start filebeat

```
ragc@romeoDebian:~$ sudo systemctl start filebeat
ragc@romeoDebian:~$
```

sudo systemctl restart filebeat

```
ragc@romeoDebian:~$  sudo systemctl restart filebeat
ragc@romeoDebian:~$
```

Habilitamos los filesets del módulo system con estos comandos:

sudo filebeat modules enable system

```
ragc@romeoDebian:~$ sudo filebeat modules enable system
Module system is already enabled
ragc@romeoDebian:~$
```

Verificamos que este activo

sudo filebeat modules list

```
ragc@romeoDebian:~$ sudo filebeat modules list
Enabled:
auditd
system
```

EJECUTAMOS LOS SIGUIENTES COMANDOS POR SEPARADO (EN EL CLIENTE).

Esto es para cargas las gráficas y los módulos en elastic, solo es necesario una vez.

El comando carga los módulos de filebeat en kibana, ejecutarlos en el cliente.

Comando 1:

sudo filebeat setup --pipelines

```
ragc@romeoDebian:~$ sudo filebeat setup --pipelines
Loaded Ingest pipelines
```

Confirmar que los filesets están realmente habilitados.

grep enabled /etc/filebeat/modules.d/system.yml

```
ragc@romeoDebian:~$ grep enabled /etc/filebeat/modules.d/system.y
    enabled: true
    enabled: true
          enabled: true
ragc@romeoDebian:~$
```

Cargar correctamente los pipelines.

**sudo filebeat setup --pipelines --modules system **

**-M "system.syslog.enabled=true" **

-M "system.auth.enabled=true"

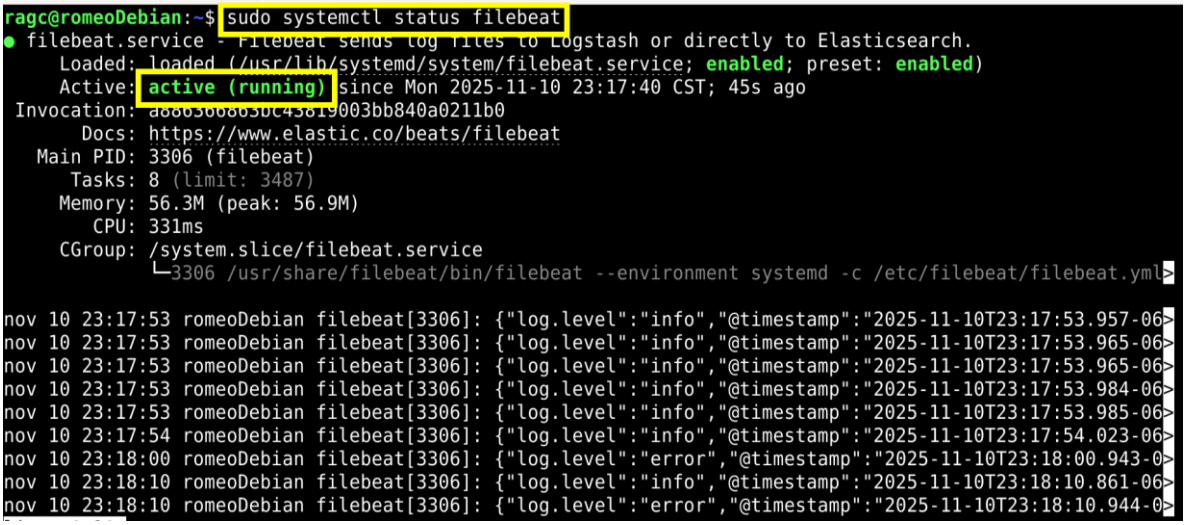
```
ragc@romeoDebian:~$ sudo filebeat setup --pipelines --modules system \
  -M "system.syslog.enabled=true" \
  -M "system.auth.enabled=true"
Loaded Ingest pipelines
ragc@romeoDebian:~$
```

Reiniciamos y confirmamos que corre bien.

sudo systemctl restart filebeat

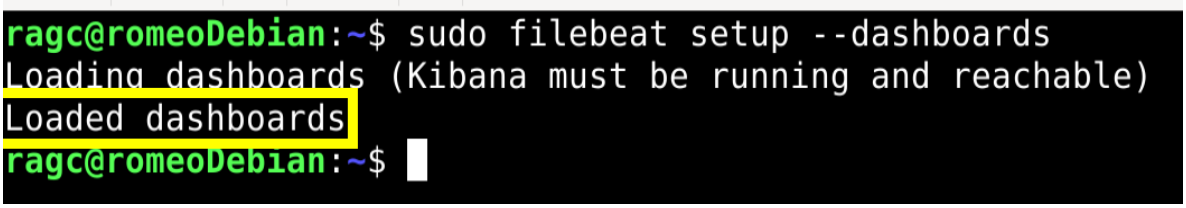
```
ragc@romeoDebian:~$ sudo systemctl restart filebeat
ragc@romeoDebian:~$
```

sudo systemctl status filebeat

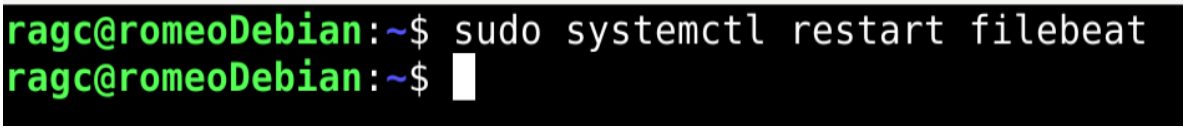


sudo filebeat setup –dashboards

Este comando depende de los recursos y del internet.



sudo systemctl restart filebeat

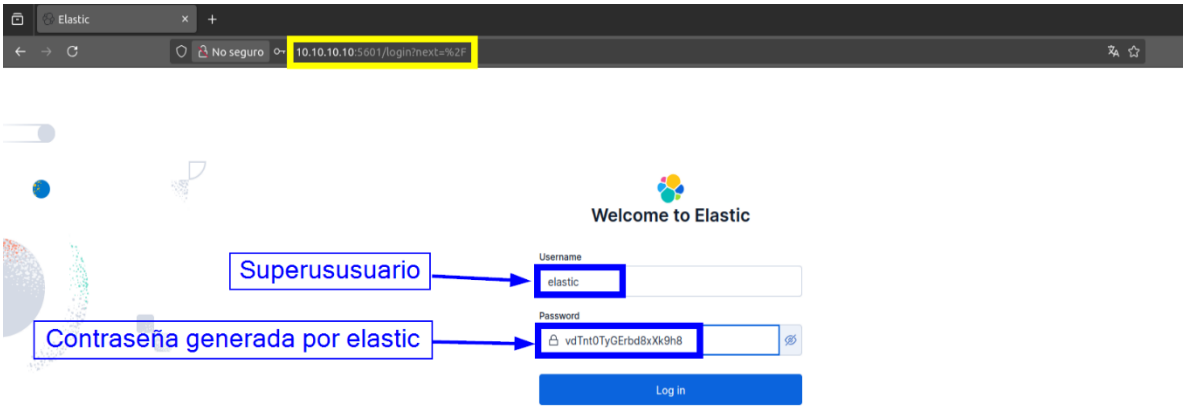


sudo systemctl status filebeat

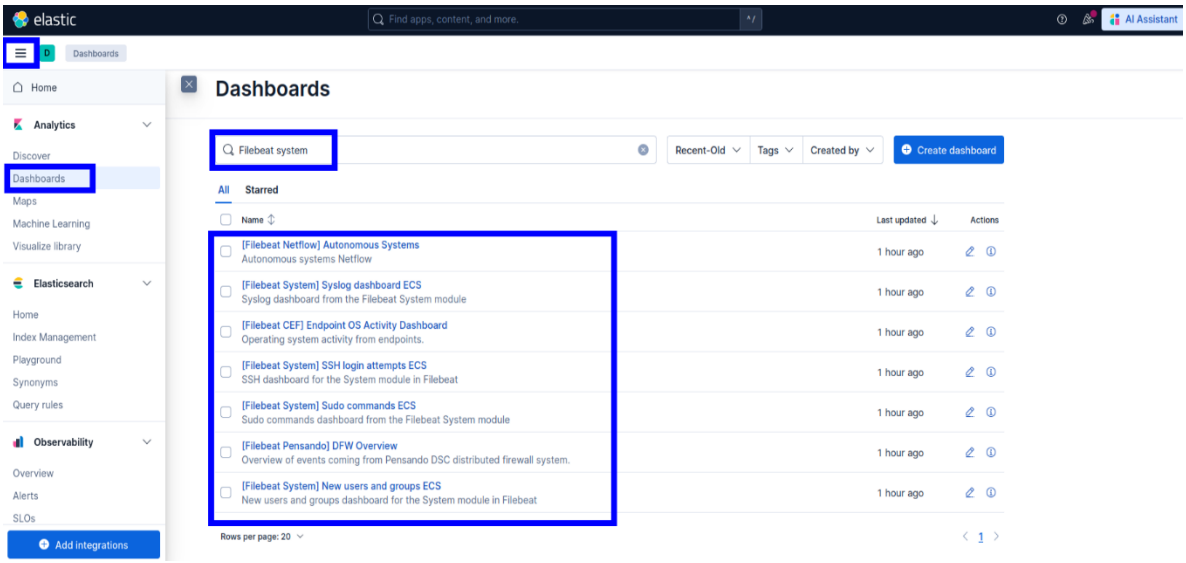
```
ragc@romeoDebian:~$ sudo systemctl status filebeat
● filebeat.service - Filebeat sends log files to Logstash or directly to Elasticsearch.
     Loaded: loaded (/usr/lib/systemd/system/filebeat.service; enabled; preset: enabled)
     Active: active (running) since Tue 2025-11-11 10:17:07 CST; 1min 54s ago
 Invocation: c02ef0ec7e5d47ee8c2d4259a8370c10
       Docs: https://www.elastic.co/beats/filebeat
   Main PID: 2187 (filebeat)
      Tasks: 8 (limit: 3487)
     Memory: 57M (peak: 59.1M)
        CPU: 635ms
     CGroup: /system.slice/filebeat.service
             └─2187 /usr/share/filebeat/bin/filebeat --environment systemd -c /etc/filebeat/filebeat.yml

nov 11 10:17:47 romeoDebian filebeat[2187]: {"log.level":"error","@timestamp":"2025-11-11T10:17:47.876-0>
nov 11 10:17:57 romeoDebian filebeat[2187]: {"log.level":"error","@timestamp":"2025-11-11T10:17:57.893-0>
nov 11 10:18:07 romeoDebian filebeat[2187]: {"log.level":"info","@timestamp":"2025-11-11T10:18:07.783-06>
nov 11 10:18:07 romeoDebian filebeat[2187]: {"log.level":"error","@timestamp":"2025-11-11T10:18:07.896-0>
nov 11 10:18:17 romeoDebian filebeat[2187]: {"log.level":"error","@timestamp":"2025-11-11T10:18:17.897-0>
nov 11 10:18:27 romeoDebian filebeat[2187]: {"log.level":"error","@timestamp":"2025-11-11T10:18:27.898-0>
nov 11 10:18:37 romeoDebian filebeat[2187]: {"log.level":"info","@timestamp":"2025-11-11T10:18:37.790-06>
nov 11 10:18:37 romeoDebian filebeat[2187]: {"log.level":"error","@timestamp":"2025-11-11T10:18:37.900-0>
nov 11 10:18:47 romeoDebian filebeat[2187]: {"log.level":"error","@timestamp":"2025-11-11T10:18:47.940-0>
```

Ahora desde el navegador (en el servidor) e ingresamos las credenciales generadas.

http://10.10.10.10:5601



Una vez iniciado, buscamos en el panel izquierdo el apartado Dashboard y ponemos en el buscador: **Filebeat system**



NOTA:

Si vemos todo tal como se ve en la imagen, entonces indica que **Filebeat y Kibana ya están correctamente integrados**.

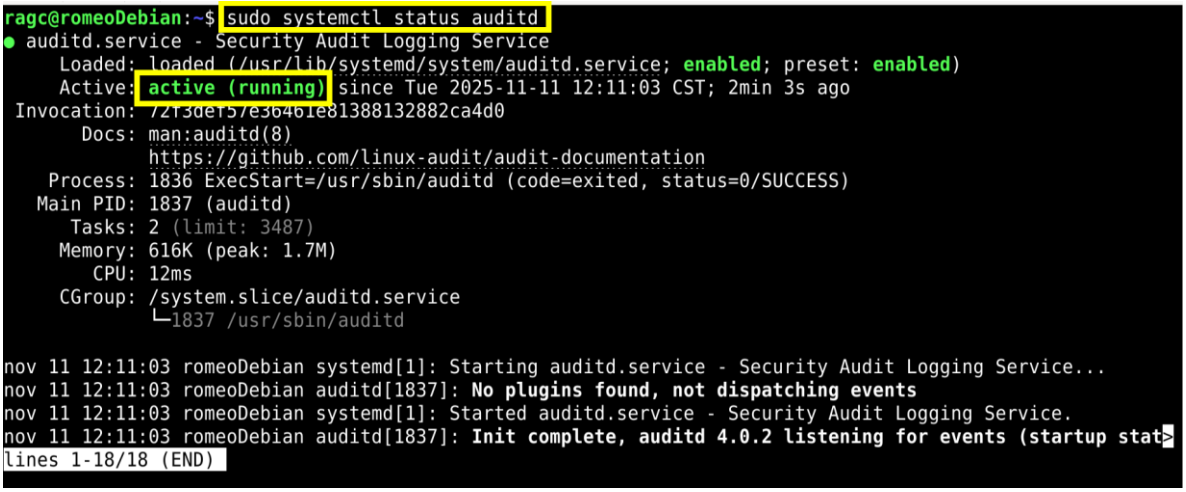
Instalamos el paquete de auditd en el cliente Debian.

sudo apt update && sudo apt install auditd -y

```
ragc@romeoDebian:~$ sudo apt update && sudo apt install auditd -y
Obj:1 http://security.debian.org/debian-security trixie-security InRelease
Obj:2 http://deb.debian.org/debian trixie InRelease
Des:3 http://deb.debian.org/debian trixie-updates InRelease [47.3 kB]
Obj:4 http://deb.debian.org/debian bookworm InRelease
Descargados 47.3 kB en 0s (101 kB/s)
Todos los paquetes están actualizados.
Installing:
  auditd

Installing dependencies:
  libauparse0t64

Paquetes sugeridos:
  audispd-plugins
```

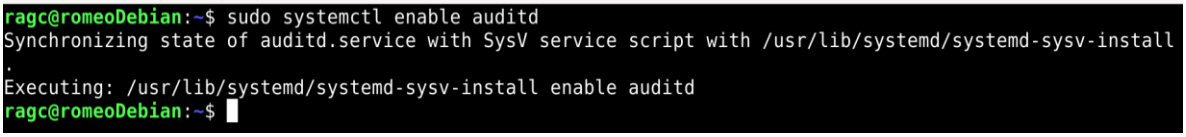
Verificamos que el servicio este activo.

sudo systemctl status auditd



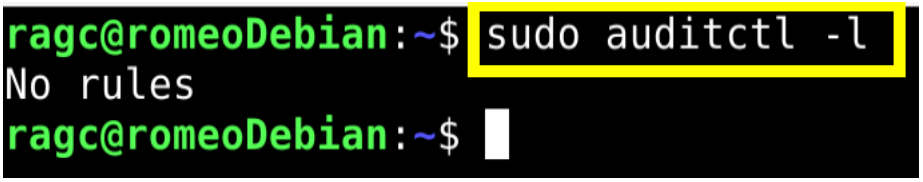
Configuramos para que trabaje automáticamente al iniciar el sistema operativo:

sudo systemctl enable auditd



Listamos las reglas:

sudo auditctl -l



Creamos el archivo de reglas.

sudo nano /etc/audit/rules.d/reglas.rules

```
ragc@romeoDebian:~$ sudo nano /etc/audit/rules.d/reglas.rules
```

Pegamos todos los comandos:

-w /etc/passwd -p wa -k monitoreo-passwd

-w /etc/shadow -p wa -k monitoreo-shadow

-w /etc/group -p wa -k monitoreo-group

-w /etc/gshadow -p wa -k monitoreo-gshadow

–w /etc/sudoers -p wa -k monitoreo-sudoers

-w /etc/ssh/sshd_config -p wa -k monitoreo-sshd_config

-w /usr/bin/ssh -p x -k monitoreo-ssh

-w /usr/bin/scp -p x -k monitoreo-scp

-w /usr/bin/rsync -p x -k monitoreo-rsync

-w /usr/bin/sftp -p x -k monitoreo-sftp

-w /usr/bin/nc -p x -k monitoreo-netcat

-w /usr/bin/nmap -p x -k monitoreo-nmap

-w /usr/bin/curl -p x -k monitoreo-curl

-w /usr/bin/wget -p x -k monitoreo-wget

-w /usr/bin/ftp -p x -k monitoreo-ftp

-w /usr/bin/smbclient -p x -k monitoreo-smbclient

-w /usr/bin/sudo -p x -k monitoreo-sudo

-w /usr/bin/passwd -p x -k monitoreo-passwd-cmd

-w /usr/sbin/useradd -p x -k monitoreo-useradd

-w /usr/sbin/userdel -p x -k monitoreo-userdel

-w /usr/sbin/usermod -p x -k monitoreo-usermod

-w /var/log -p wa -k monitoreo-logs

-w /home/cliente1/Escritorio -p rwxa -k monitoreo-escritorio

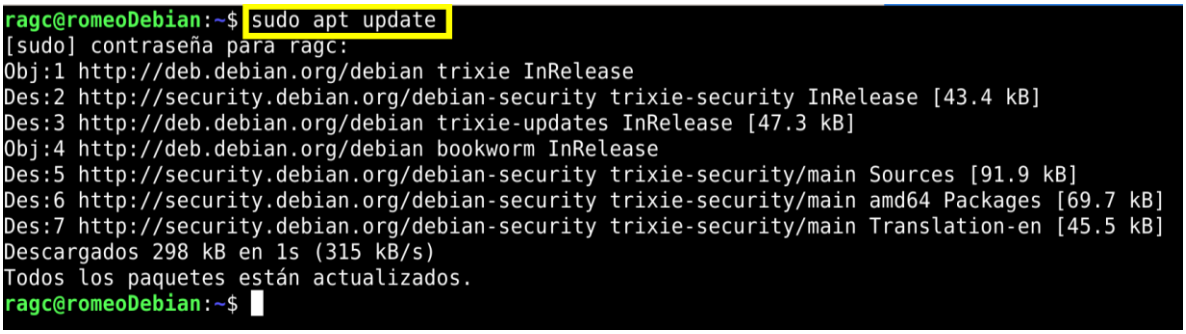
-a always,exit -F arch=b64 -S connect -F a0=0x2 -F key=network_connect

-a always,exit -F arch=b32 -S connect -F a0=0x2 -F key=network_connect

```
  GNU nano 8.4                          /etc/audit/rules.d/reglas.rules *
# Archivos críticos
-w /etc/passwd -p wa -k monitoreo-passwd
-w /etc/shadow -p wa -k monitoreo-shadow
-w /etc/group -p wa -k monitoreo-group
-w /etc/gshadow -p wa -k monitoreo-gshadow
-w /etc/sudoers -p wa -k monitoreo-sudoers
-w /etc/ssh/sshd_config -p wa -k monitoreo-sshd_config

# Comandos importantes
-w /usr/bin/ssh -p x -k monitoreo-ssh
-w /usr/bin/scp -p x -k monitoreo-scp
-w /usr/bin/rsync -p x -k monitoreo-rsync
-w /usr/bin/sftp -p x -k monitoreo-sftp
-w /usr/bin/nc -p x -k monitoreo-netcat
-w /usr/bin/nmap -p x -k monitoreo-nmap
-w /usr/bin/curl -p x -k monitoreo-curl
-w /usr/bin/wget -p x -k monitoreo-wget
-w /usr/bin/ftp -p x -k monitoreo-ftp
```

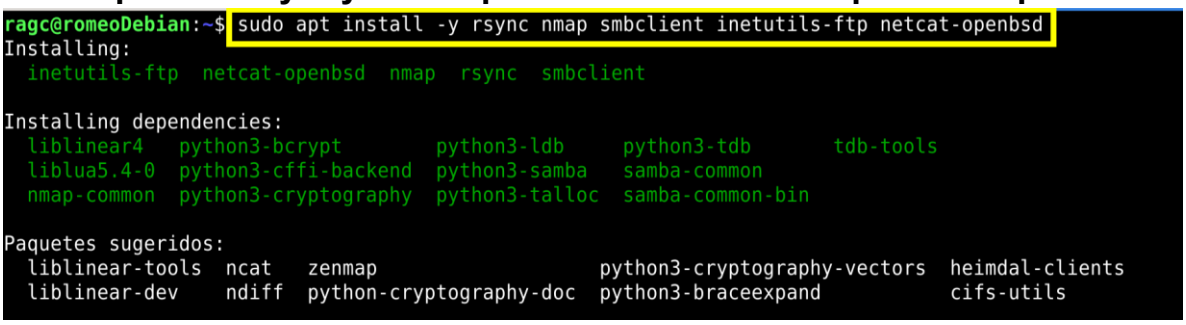
Vamos a instalar repositorios.

sudo apt update



Instalamos binarios faltantes:

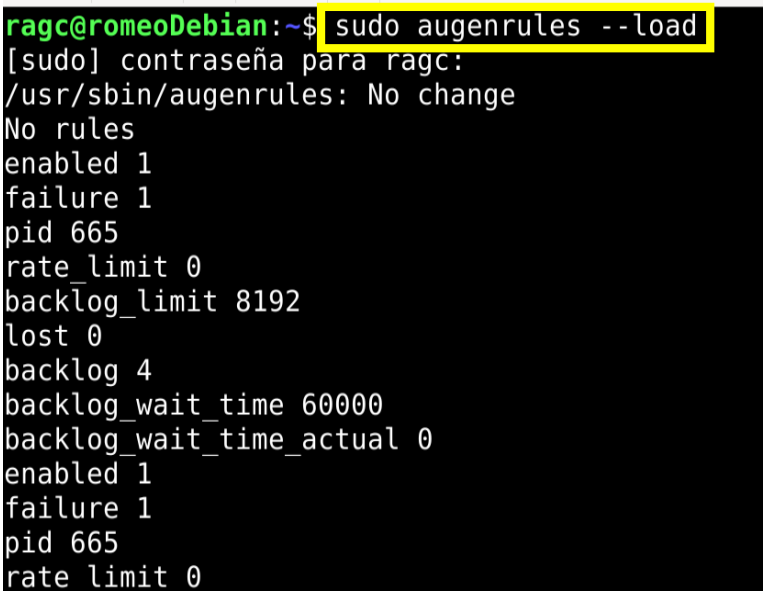
sudo apt install -y rsync nmap smbclient inetutils-ftp netcat-openbsd



Cargamos las reglas

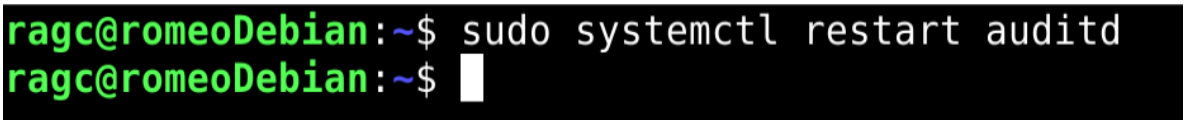
sudo augenrules --load

Compila todas las reglas de /etc/audit/rules.d/.



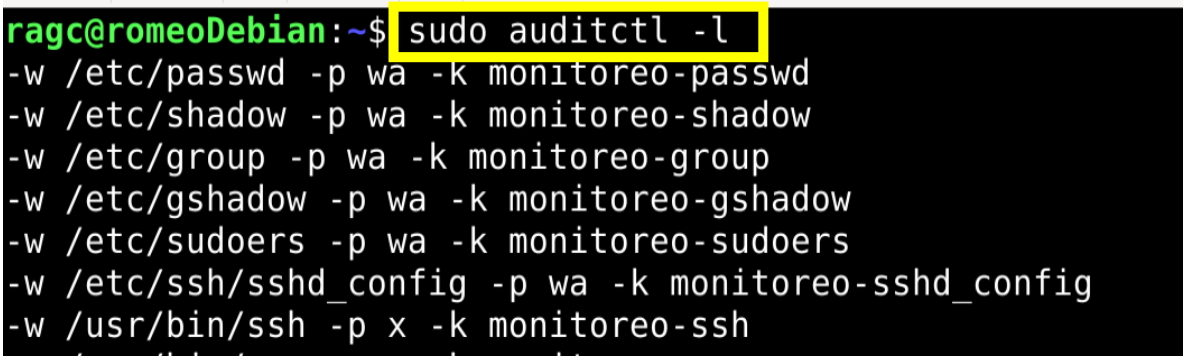
sudo systemctl restart auditd

Reinicia el servicio y aplica las nuevas reglas.



sudo auditctl -l

Lista las reglas activas.



Reiniciamos servicios de Audit.

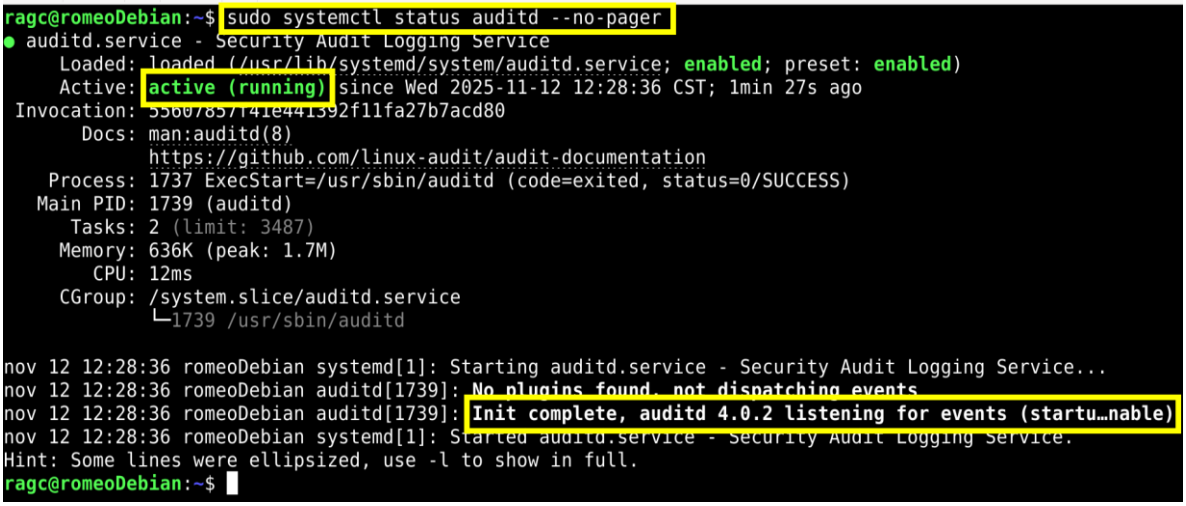
sudo systemctl restart auditd

```
ragc@romeoDebian:~$ sudo systemctl restart auditd
ragc@romeoDebian:~$
```

Verificamos que todo este activo.

sudo systemctl status auditd --no-pager

Se le agrega “—no-pager” para Muéstrame toda la salida completa de una vez, sin pausas ni paginación.



Podemos ver que se inicializó sin errores y está escuchando los eventos del kernel.

AGREGAR UNA RED FIREWALL PARA RECONOCIMIENTOS DE PING.

Implementamos la red firewall para el reconocimiento de ping. (EN EL SERVIDOR).

Este comando agrega una regla al firewall para registrar (no bloquear) todos los intentos de ping (ICMP echo-request) que lleguen al sistema, guardándolos en los archivos de log con el prefijo “PING:”.

sudo iptables -A INPUT -p icmp --icmp-type echo-request -j LOG --log-prefix "PING: "

Con esto cada que alguien haga un ping a mi máquina se verá una línea en los logs, similar a esta:

Nov 12 12:45:10 romeoDebian kernel: [12345.678901] PING: IN=eth0 OUT= MAC=... SRC=10.10.10.15 DST=10.10.10.10 LEN=84 ...

NOTA: Esto no bloquea el ping, solo lo registra — justo como pide la guía (para monitoreo).

Prueba de funcionalidad.

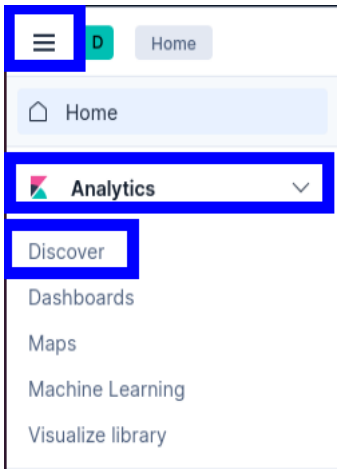
Vamos al servidor Ubuntu server 20.04 y en el navegador accedemos al kibana del servidor.

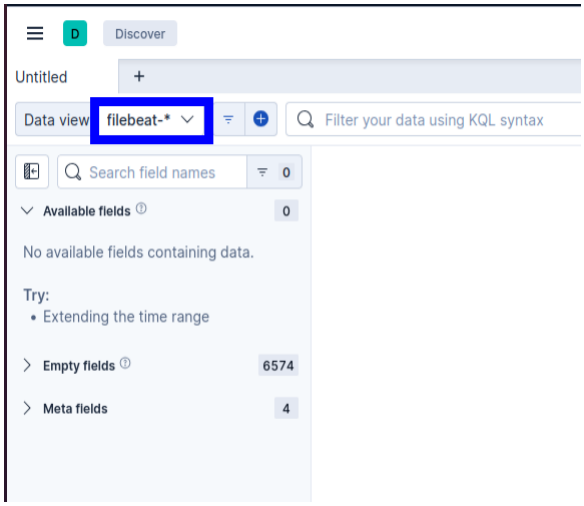
http://10.10.10.10:5601

Lo que estamos comprobando es que Filebeat (ya configurado y enviando logs del sistema) esté transmitiendo datos correctamente hacia Elasticsearch y que Kibana los esté visualizando.

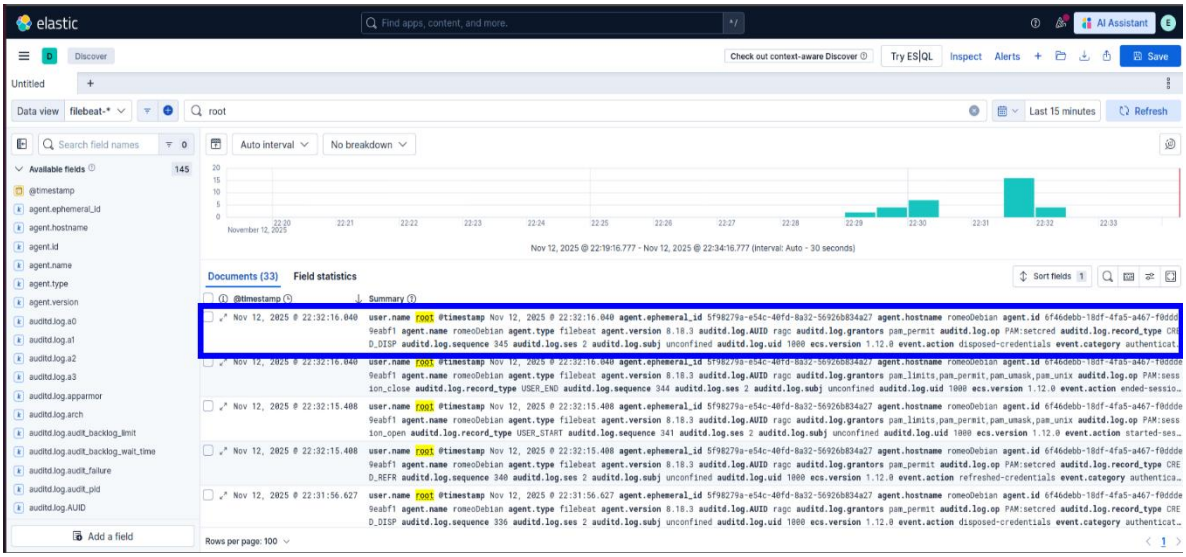
Módulo system.

Ir al menú de la izquierda, en el apartado de analytics, discover, en dataview tendrá que aparecer el grupo filebeat-*





Generaremos un log de inicio de sesión, en el cliente que estamos monitoreando, acceder como usuario root, y buscamos el log en kibana.



Modulo Audit.

Vamos a probar que las reglas de Audit funcionan.