



APUNTES 07

SIEM ELK

INCIDENTES DE CIBERSEGURIDAD

ALBA MOREJÓN GARCÍA

2024/2025

Ciberseguridad en Entornos de las Tecnologías de la Información

ÍNDICE

1. Escenario de Trabajo SIEM.
 - 1.1. Premisas para la Práctica SIEM.
 - 1.2. Instalación de OpenJDK.
 - 1.3. Instalación de Elasticsearch.
 - 1.3.1. Descarga y Edición del Fichero de Configuración.
 - 1.3.2. Limitación del Uso de la Memoria RAM.
 - 1.3.3. Arranque y Chequeo de Status.
 - 1.3.4. Uso de Nmap para Comprobar Acceso a Elasticsearch.
 - 1.4. Instalación de Logstash.
 - 1.4.1. Descarga, Instalación y Limitación del Uso de la RAM.
 - 1.4.2. Edición del Fichero de Configuración, Arranque y Comprobación del Status.
 - 1.4.3. Revisión del Log de Arranque y Parada de la Aplicación.
 - 1.5. Instalación de Kibana.
 - 1.5.1. Descarga de Node.js.
 - 1.5.2. Procedimiento de Instalación.
 - 1.5.3. Configuración de Kibana.
 - 1.5.4. Sustitución de Node.js por su Versión Correcta.
 - 1.5.5. Edición de un Fichero de Servicio y Arranque.
 - 1.6. Configuración SIEM.
 - 1.6.1. Arranque del SIEM.
 - 1.6.2. Pipelines en Logstash.
 - 1.7. Visualización SIEM.
 - 1.7.1. Arranque de Kibana desde el Navegador.
 - 1.7.2. Configuración de Kibana.
 - 1.7.3. Manejo de Kibana.
 - 1.7.4. Creación de un Histograma.
 - 1.7.5. Creación de un Tablero.

Gestión de Eventos e Información de Seguridad

La detección IDS es el primer estadio de tratamiento de la información en un SOC. La información recolectada en esta etapa deberá ser filtrada, clasificada y almacenada en las siguientes etapas de tratamiento, mediante las herramientas adecuadas.

Hecho esto, dará comienzo la etapa final, que se desarrollará de forma continua una vez iniciada, esto es, el análisis de información para detectar patrones de ataque y sacar las conclusiones correspondientes de cara a la Prevención de Incidentes de Ciberseguridad.

Esta unidad complementa las tareas efectuadas en la unidad anterior, esto es, Detección IDS, por lo que se partirá con Snort instalado, configurado y funcionando en la máquina del laboratorio.

Así pues, se añadirán a la máquina el resto de componentes necesarios para disponer de un SIEM completo y operativo (se recomienda utilizar como mínimo una Raspberry Pi 4B 4GB como escenario de trabajo).

1.- ESCENARIO DE TRABAJO SIEM.

Caso práctico

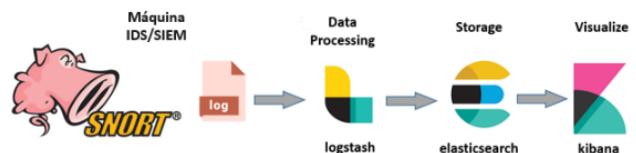
Para implementar el SIEM se instalarán en Raspbian los componentes Linux del Stack ELK, y se visualizará en un PC el dashboard para monitorización de información y eventos de seguridad. No será preciso instalar ningún SW en el PC, sólo hará falta disponer de un navegador. El sistema resultante permitirá efectuar las labores de Detección de Intrusos, Prevención de Ataques, monitorización de intrusiones, preparación de Cuadros de Mando, Gestión de Alarmas, etc., que son las que se efectúan habitualmente en un SOC.

Esta unidad conecta con las tareas efectuadas en la unidad anterior, esto es, Detección IDS, por lo que se partirá con Snort instalado, configurado y funcionando en la máquina DMZ1.

En esta sesión se añadirán a la máquina el resto de componentes necesarios para disponer de un SIEM completo y operativo (se recomienda utilizar como mínimo una Raspberry Pi 4B 4GB como escenario de trabajo).

Para ello, se instalarán en Raspbian los componentes Linux del Stack ELK, y se visualizará en un PC el dashboard para monitorización de información y eventos de seguridad (no se precisa instalar ningún SW en el PC, sólo hará falta disponer de un navegador).

En resumidas cuentas, el sistema resultante permitirá efectuar las labores de Detección de Intrusos, Prevención de Ataques, monitorización de intrusiones, preparación de Cuadros de Mando, Gestión de Alarmas, etc., que son las que se efectúan habitualmente en un SOC.



1.1.- PREMISAS PARA LA PRÁCTICA SIEM.

Premisas de Partida – Snort instalado y operativo

En primer lugar y para evitar problemas durante la instalación del SIEM, nos debemos asegurar de que el fichero de alertas de Snort tenga todos los permisos:

```
sudo chmod 777 /var/log/snort_alerts.log
```

En la sesión anterior ya incluimos las reglas de detección de ICMP y TCP para levantar alertas de acceso ping y ssh, por lo que pudimos comprobar que el fichero de alertas va creciendo adecuadamente según la progresión de los ataques.

Antes de proseguir con la instalación del SW del SIEM, es conveniente comprobar también que se actualiza la fecha y hora del fichero de log interno binario de Snort:

```
ls -l /var/log/snort/snort.log
```

Efectuadas estas comprobaciones, detenemos Snort antes de proseguir con la instalación del SIEM:

```
sudo systemctl stop snort
```

1.2.- INSTALACIÓN DE OPENJDK.

Para comenzar, el stack ELK requiere la instalación del Open Java Development Kit, al menos en su versión 8, junto con un par de librerías adicionales (no obstante, se deberá chequear la versión del OpenJDK requerida por la versión de los productos de ELK que se vayan a instalar en cada caso):

```
sudo apt install openjdk-8-jdk libjffi-java libjffi-jni
```

NOTA IMPORTANTE: Al ejecutar los comandos y transcribir los textos indicados a los ficheros correspondientes, deberemos poner especial atención en las comillas, como ya comentamos en la sesión anterior. Deberán figurar en todos los lugares indicados y no deberán ser las comillas tipográficas que en ocasiones insertan las aplicaciones de edición de textos.

1.3.- INSTALACIÓN DE ELASTICSEARCH.

Elasticsearch es el corazón del SIEM.

Se trata de una base de datos tipo NoSQL, esto es, Not Only SQL. Estas bases de datos están preparadas para almacenar cualquier tipo de información de forma inmediata, aunque su formato no cuadre exactamente con la estructura interna. Esta flexibilidad unida a su capacidad para manejar grandes cantidades de datos, hacen que estos gestores de bases de datos sean muy apropiados para el ámbito de Big Data.

En el caso del SIEM su misión será almacenar toda la información relacionada con los incidentes detectados, para poder analizarla adecuadamente y extraer rápidamente las conclusiones necesarias de cara a la prevención.

1.3.1.- DESCARGA Y EDICIÓN DEL FICHERO DE CONFIGURACIÓN.

Descargamos el paquete Elasticsearch, que indexará y almacenará en su base de datos nuestras alertas de seguridad y nuestros logs. Lo haremos con wget y lo instalaremos con dpkg:

```
sudo wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.5.4.deb
sudo dpkg -i elasticsearch-6.5.4.deb
```

Editamos el fichero de configuración de elasticsearch y asignamos los siguientes valores a las variables:

```
sudo nano /etc/elasticsearch/elasticsearch.yml
cluster.name: siem.sky.net
node.name: node01.siem.sky.net
network.host: 192.168.1.21
discovery.type: single-node
xpack.ml.enabled: false
```

1.3.2.- LIMITACIÓN DEL USO DE LA MEMORIA RAM.

Configuramos elasticsearch para que utilice un máximo de 256 MB de RAM, lo cual resulta más que suficiente para un ejercicio de laboratorio:

```
sudo nano /etc/elasticsearch/jvm.options
-Xms256m
-Xmx256m
```

1.3.3.- ARRANQUE Y CHEQUEO DE STATUS.

Arrancamos Elasticsearch y comprobamos su estado:

```
sudo systemctl start elasticsearch
sudo systemctl status elasticsearch
```

```
pi@DMZ1: ~
pi@DMZ1:~ $ sudo nmap 192.168.1.21 -p9200
Starting Nmap 7.70 ( https://nmap.org ) at 2021-09-09 16:29 CEST
Nmap scan report for DMZ1 (192.168.1.21)
Host is up (0.0001s latency).

PORT      STATE SERVICE
9200/tcp  open  wap-wsp

Nmap done: 1 IP address (1 host up) scanned in 0.50 seconds
```

1.3.4.- USO DE NMAP PARA COMPROBAR ACCESO A ELASTICSEARCH.

Finalmente, instalamos nmap para chequear si Elasticsearch está a la escucha en el puerto adecuado:

```
sudo apt-get install nmap
```

Si todo está correctamente instalado y la aplicación está corriendo, Elasticsearch deberá estar a la escucha en el puerto 9200:

```
sudo nmap 192.168.1.21 -p9200
```

Además se podrá comprobar la operatividad de elasticsearch desde el PC, utilizando la dirección IP y el puerto 9200 en cualquier navegador:

```
http://192.168.1.21:9200
```

1.4.- INSTALACIÓN DE LOGSTASH.

Logstash es una herramienta de tipo ETL, esto es, Extract, Translate & Load.

Su misión es filtrar los datos crudos procedentes del IDS e insertar la información seleccionada en la base de datos Elasticsearch.

Como se verá a continuación, esta misión se lleva a cabo mediante entidades denominadas pipelines.

1.4.1.- DESCARGA, INSTALACIÓN Y LIMITACIÓN DEL USO DE LA RAM.

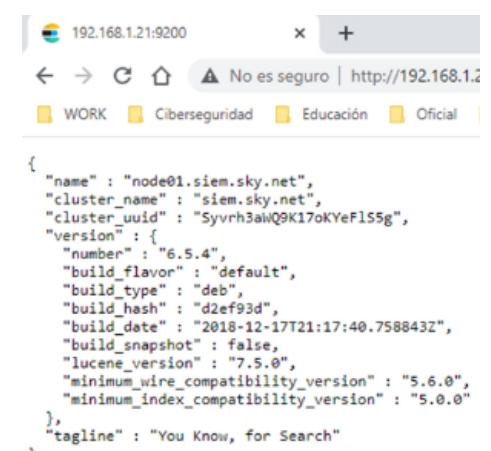
Antes de pasar a la instalación de logstash, paramos elasticsearch:

```
sudo systemctl stop elasticsearch
```

Logstash es la aplicación que analizará gramaticalmente las alertas y los logs de Snort, filtrando y estructurando la información antes de insertarla en la base de datos de Elasticsearch.

Se instala de una forma similar a elasticsearch, limitando igualmente la RAM utilizada en función de la CPU de que se disponga (recomendados también 256 MB):

```
sudo wget https://artifacts.elastic.co/downloads/logstash/logstash-6.5.4.deb
sudo dpkg -i logstash-6.5.4.deb
sudo nano /etc/logstash/jvm.options
-Xms256m
-Xmx256m
```



1.4.2.- EDICIÓN DEL FICHERO DE CONFIGURACIÓN, ARRANQUE Y COMPROBACIÓN DEL STATUS.

Editamos el fichero de configuración de logstash y asignamos los siguientes valores (atención a la dirección IP, que en este caso va entre comillas dobles):

```
sudo nano /etc/logstash/logstash.yml
node.name: node01.siem.sky.net
http.host: "192.168.1.21"
xpack.monitoring.enabled: false
xpack.management.enabled: false
```

Finalmente, arrancamos logstash, chequeamos su log de arranque para ver si todo ha ido bien, y lo detenemos antes de proseguir con la instalación de kibana:

```
sudo systemctl start logstash
sudo systemctl status logstash
```

1.4.3.- REVISIÓN DEL LOG DE ARRANQUE Y PARADA DE LA APLICACIÓN.

Esperamos 2 minutos y chequeamos el log de arranque:

```
sudo tail -f /var/log/logstash/logstash-plain.log
```

Se observarán varios mensajes de error notificando que aún falta completar la configuración, no obstante, si aparece el mensaje "Successfully started Logstash API endpoint", esto indicará que todo ha ido bien.

Antes de proseguir, detenemos logstash y comprobamos que está parado:

```
sudo systemctl stop logstash
sudo systemctl status logstash
```

Deberá aparecer que está "inactive" o "failed"

1.5.- INSTALACIÓN DE KIBANA.

En el stack ELK, Kibana es la herramienta que se utiliza para construir cuadros de mando interactivos (dashboards) y monitorizar las notificaciones y las alarmas.

Kibana 6.5.4 depende de Node.js 8.14. Esta versión de Node.js no es la que incluye de serie la instalación de Kibana. A continuación veremos cómo sustituirla.

IMPORTANTE: Kibana no puede convivir con otras aplicaciones que ocupen el puerto 80 en la misma máquina, por ejemplo, Apache. Si las hubiera, sería necesario desinstalarlas antes de proseguir con la instalación de Kibana.

1.5.1.- DESCARGA DE NODE.JS.

Situarse en el directorio home del usuario sudoer (pi): cd

Descargar la versión mencionada de Node.js siguiendo detalladamente los pasos que se indican a continuación.

Tras este procedimiento, el ejecutable quedará situado en el directorio correcto.

```
sudo wget https://nodejs.org/dist/v8.14.0/node-v8.14.0-linux-armv7l.tar.xz
sudo tar -xvf node-v8.14.0-linux-armv7l.tar.xz node-v8.14.0-linux-armv7l/bin/node
cd node-v8.14.0-linux-armv7l/bin
sudo cp ./node /usr/local/bin/
```

1.5.2.- PROCEDIMIENTO DE INSTALACIÓN.

Instalar Kibana mediante el siguiente procedimiento:

```
sudo wget https://artifacts.elastic.co/downloads/kibana/kibana-6.5.4-linux-x86_64.tar.gz
sudo mkdir /usr/share/kibana/
sudo tar -xvf kibana-6.5.4-linux-x86_64.tar.gz --strip 1 --directory
/usr/share/kibana/
```

1.5.3.- CONFIGURACIÓN DE KIBANA.

Configurar Kibana editando el fichero siguiente:

```
sudo nano /usr/share/kibana/config/kibana.yml
```

Incluyendo las siguientes variables en su interior (no borrar las comillas):

```
server.port: 80
server.host: "192.168.1.21"
server.name: node01.siem.sky.net
elasticsearch.url: "http://192.168.1.21:9200"
logging.dest: /var/log/kibana.log
```

```
pi@DMZ1:~ $ ls -l
total 480700
drwxr-xr-x 2 pi pi 4096 jun 18 08:32 arduino-clli
-rw-r--r-- 1 pi pi 88 jun 7 20:52 ARRANCAR_MINISOC
drwxr-xr-x 2 pi pi 4096 jun 18 08:32 arduino-clli
-rw-r--r-- 1 pi pi 33 sep 7 11:18 deactivate_wlan
drwxr-xr-x 2 pi pi 4096 may 7 17:07 Desktop
drwxr-xr-x 2 pi pi 4096 may 7 17:07 Documents
drwxr-xr-x 2 pi pi 4096 may 7 17:07 Downloads
drwxr-xr-x 1 root root 113259798 dic 19 2018 elasticsearch-6.5.4.deb
-rw-r--r-- 1 root root 206631363 dic 19 2018 kibana-6.5.4-linux-x86_64.tar.gz
drwxr-xr-x 2 pi pi 4096 jun 7 21:03 logstash-6.5.4.deb
drwxr-xr-x 2 pi pi 4096 jun 7 17:07 logstash-6.5.4.deb
drwxr-xr-x 3 root root 4096 jun 7 17:47 node-v8.14.0-linux-armv7l
drwxr-xr-x 1 root root 10207552 nov 27 2018 node-v8.14.0-linux-armv7l.tar.xz
-rw-r--r-- 1 pi pi 86 jun 7 21:03 PARAR_MINISOC
drwxr-xr-x 2 pi pi 4096 may 7 17:07 Pictures
drwxr-xr-x 2 pi pi 4096 may 7 17:07 Public
drwxr-xr-x 2 pi pi 4096 may 7 17:07 Templates
drwxr-xr-x 1 pi pi 4096 may 7 17:07 Videos
```

1.5.4.- SUSTITUCIÓN DE NODE.JS POR SU VERSIÓN CORRECTA.

Desactivar el Node.js que trae incluido Kibana, e indicarle que use la versión descargada ad hoc:

```
sudo mv /usr/share/kibana/node/bin/node /usr/share/kibana/node/bin/node.bak
sudo ln -s /usr/local/bin/node /usr/share/kibana/node/bin/node
```

```
pi@DMZ1:~ $ cd /usr/share/kibana/node/bin
pi@DMZ1:/usr/share/kibana/node/bin $ ls -l node*
lrwxrwxrwx 1 root root 19 jun 7 18:00 node -> /usr/local/bin/node
-rwxrwxr-x 1 pi pi 34986105 nov 27 2018 node.bak
```

1.5.5.- EDICIÓN DE UN FICHERO DE SERVICIO Y ARRANQUE.

Escribir un fichero de servicio "Systemd" para arrancar y parar Kibana:
 sudo nano /etc/systemd/system/kibana.service

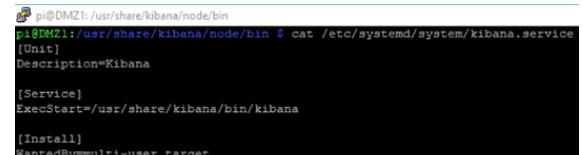
Incluir este contenido en el fichero de servicio:

```
[Unit]
Description=Kibana
[Service]
ExecStart=/usr/share/kibana/bin/kibana
[Install]
WantedBy=multi-user.target
```

Arrancar Kibana y comprobar su status:

```
sudo systemctl start kibana
sudo systemctl status kibana
```

Deberá aparecer que está "active (running)"



```
pi@DMZ1: /usr/share/kibana/node/bin $ cat /etc/systemd/system/kibana.service
[Unit]
Description=Kibana
[Service]
ExecStart=/usr/share/kibana/bin/kibana
[Install]
WantedBy=multi-user.target
```

1.6.- CONFIGURACIÓN SIEM.

Una vez que se han instalado todas las piezas de la torre ELK y se ha efectuado su configuración de bajo nivel, se puede dar paso al proceso de configuración funcional.

En él se verá que la configuración de Elasticsearch es prácticamente inexistente, mientras que Logstash requerirá definir pipelines y Kibana requerirá definir métricas y tableros.

1.6.1.- ARRANQUE DEL SIEM.

Reiniciar el host:

```
sudo reboot
```

Arrancar Elasticsearch siempre ANTES que kibana:

```
sudo systemctl start elasticsearch
sudo systemctl status elasticsearch
```

Arrancar Kibana:

```
sudo systemctl start kibana
sudo systemctl status kibana
```

Comprobar que está arrancado Elasticsearch mediante el navegador del PC (devolverá su lista de características en formato JSON):

```
http://192.168.1.21:9200
```

Esperar 10 minutos antes de comprobar que está arrancado Kibana, introduciendo su URL en el navegador del PC (arrancará la Home de Kibana):

```
http://192.168.1.21
```

Arrancar el resto de las aplicaciones del SOC y comprobar que quedan activas:

```
sudo systemctl start logstash
sudo systemctl status logstash
sudo systemctl start snort
sudo systemctl status snort
```

1.6.2.- PIPELINES EN LOGSTASH.

Configuración de logstash para capturar el log de Snort, procesar sus mensajes y grabarlos en Elasticsearch.

Para esto hay que definir una tubería de logstash (pipeline), con entrada, filtrado y salida hacia elasticsearch, en el fichero /etc/logstash/conf.d/logstash.conf

Las tuberías o pipelines tienen:

- Entrada. Indicación del punto de captura de datos, que puede ser un fichero local o remoto, un punto de acceso a un protocolo de comunicaciones, etc.

- Filtrado. La información de los logs y de los ficheros de trabajo está estructurada de forma muy variada, por lo que hay que utilizar la herramienta de filtrado Grok para formatearla antes de insertarla en la base de datos. Esta herramienta está incluida dentro de Logstash.

- Salida. Logstash puede producir salidas en varios formatos de ficheros, o bien, grabarla directamente en una base de datos NoSQL, como es el caso de Elasticsearch.

- Entrada al Pipeline

Log de Snort que leerá Logstash:

```
/var/log/snort_alerts.log
```

Como resultado de la configuración efectuada en la sesión anterior, en el fichero /etc/snort/rules/local.rules tendremos las reglas siguientes:

```
# Regla para detectar un ping (ICMP)
alert icmp any any -> $HOME_NET any (msg:"¡Trafico ICMP!"; sid:3000001;)

#Regla para detectar un ssh (TCP y puerto 22)
alert tcp any any -> any 22 (msg: "Acceso SSH"; sid:3000002;)
```

Editamos pues el fichero /etc/logstash/conf.d/logstash.conf.

Creamos la entrada a Logstash, para captura del fichero de log de Snort.

```
input {
  file {
    path => ["'/var/log/snort_alerts.log"]
    start_position => beginning
  }
}
```

- Filtrado Grok en el Pipeline

Se efectuará el filtrado de la información entrante al pipeline con la potente herramienta Grok. Aunque la construcción de los patrones Grok se revisará a continuación con varios ejemplos, en este momento estudiaremos un patrón sencillo, con la sintaxis siguiente:

```
filter {
  grok {
    match => {"message" => "%{GREEDYDATA:cadena}"}
  }
}
```



Este patrón es el más simple de todos, pues captura una línea completa del log de Snort, la asigna a variable "cadena" como un string, y la inserta en la base de datos de Elasticsearch.

Existe abundante documentación descriptiva de la sintaxis de los Grok Patterns.

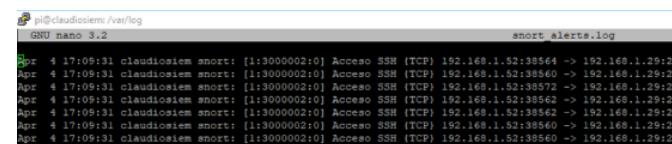
Además, en Kibana hay un depurador Grok que permite ejecutar reglas de forma unitaria sobre una línea de log de muestra y comprobar si las reglas funcionan correctamente.

kibana -> Dev Tools -> Grok Debugger

- Salida del Pipeline hacia Elasticsearch

Definición de la salida del pipeline, que en este caso será el índice "logstash" en la base de datos Elasticsearch:

```
output {
  elasticsearch {
    hosts => [ "http://192.168.1.21:9200" ]
    index => "logstash"
  }
}
```



- Comprobación del Funcionamiento del Pipeline

Para comprobar el funcionamiento del pipeline recién creado, lanzaremos un Ping y un ataque SSH desde otra máquina en sendos terminales, de forma que se registren los eventos asociados a dichos ataques.

Desde kibana se podrá comprobar que Elasticsearch está ingiriendo la información formateada que le envía logstash a través de la tubería creada.

Para ello se deberá ir a Dev Tools, seleccionar Console y lanzar el comando siguiente para ver el contenido del índice logstash:

POST logstash/_search

Se obtendrá una respuesta similar a la de la figura adjunta.



- **Patrón Grok Genérico para un String**

```
Sep 29 17:37:38 claudiosiem snort: [1:3000002:0] Acceso SSH {TCP} 192.168.1.64:55317 -> 192.168.1.29:22
%{GREEDYDATA:cadena}
{
  "cadena": "Sep 29 17:37:38 claudiosiem snort: [1:3000002:0] Acceso SSH {TCP} 192.168.1.64:55317 -> 192.168.1.29:22"
}
```

Nota. GREEDYDATA. Patrón “Codicioso” – No filtra nada. Captura TODO y lo guarda en una variable

- **Patrón Grok para Extraer Varios Campos**

```
Sep 29 17:37:38 claudiosiem snort: [1:3000002:0] Acceso SSH {TCP} 192.168.1.64:55317 -> 192.168.1.29:22
%{MONTH:mes}%{SPACE}%{MONTHDAY:dia}%{SPACE}%{HOUR:hora}: %{MINUTE:minutos}: %{SECOND:segundos}%
GREEDYDATA:resto_mensaje
{
  "hora": "17",
  "resto_mensaje": " claudiosiem snort: [1:3000002:0] Acceso SSH {TCP} 192.168.1.64:55317 -> 192.168.1.29:22",
  "segundos": "38",
  "mes": "Sep",
  "minutos": "37",
  "dia": "29"
}
```

- **Patrón Grok Literal para Extraer Todos los Campos**

```
Sep 29 17:37:38 claudiosiem snort: [1:3000002:0] Acceso SSH {TCP} 192.168.1.64:55317 -> 192.168.1.29:22
%{MONTH:mes}%{SPACE}%{MONTHDAY:dia}%{SPACE}%{HOUR:hora}: %{MINUTE:minutos}: %{SECOND:segundos}%
PACE}%{WORD:nombre_maquina}%{SPACE}%{WORD:origen_log}: %{SPACE}.%{NUMBER:id_1}: %{NUMBER:id_2}: %N
UMBER:id_3}.%{SPACE}%{WORD:accion}%{SPACE}%{WORD:tipo_sesion}%{SPACE}.%{WORD:protocolo}.%{SPACE}%
IP:ip_origen}: %{NUMBER:puerto_origen}....%{IP:ip_destino}: %{NUMBER:puerto_destino}
{ "accion": "Acceso", "nombre_maquina": "claudiosiem", "ip_destino": "192.168.1.29", "ip_origen": "192.168.1.64", "hora":
"17", "minutos": "37", "origen_log": "snort", "protocolo": "TCP", "puerto_origen": "55317", "tipo_sesion": "SSH",
"puerto_destino": "22", "segundos": "38", "mes": "Sep", "id_1": "1", "dia": "29", "id_3": "0", "id_2": "3000002"}
```

- **Patrón con Comodines (Puntos)**

```
Sep 29 17:37:38 claudiosiem snort: [1:3000002:0] Acceso SSH {TCP} 192.168.1.64:55317 -> 192.168.1.29:22
%{MONTH:mes}.%{MONTHDAY:dia}.%{HOUR:hora}.%{MINUTE:minutos}.%{SECOND:segundos}.%{WORD:nombre_maq
uina}.%{WORD:origen_log}...%{NUMBER:id_1}.%{NUMBER:id_2}.%{NUMBER:id_3}..%{WORD:accion}.%{WORD:tipo_se
sion}..%{WORD:protocolo}..%{IP:ip_origen}.%{NUMBER:puerto_origen}....%{IP:ip_destino}:%{NUMBER:puerto_destino}
{ "accion": "Acceso", "nombre_maquina": "claudiosiem", "ip_destino": "192.168.1.29", "ip_origen": "192.168.1.64", "hora":
"17", "minutos": "37", "origen_log": "snort", "protocolo": "TCP", "puerto_origen": "55317", "tipo_sesion": "SSH",
"puerto_destino": "22", "segundos": "38", "mes": "Sep", "id_1": "1", "dia": "29", "id_3": "0", "id_2": "3000002"}
```

- **Patrón Grok con Timestamp y Greedydata**

```
Sep 29 17:37:38 claudiosiem snort: [1:3000002:0] Acceso SSH {TCP} 192.168.1.64:55317 -> 192.168.1.29:22
%{SYSLOGTIMESTAMP:fecha}.%{WORD:maquina}%{GREEDYDATA:mensaje} %{IP:dir_a}:%{INT:port_a}....%{IP:dir_b}:%{I
NT:port_b}
{ "fecha": "Sep 29 17:37:38", "dir_b": "192.168.1.29", "port_a": "55317", "port_b": "22", "maquina": "claudiosiem",
"mensaje": " snort: [1:3000002:0] Acceso SSH {TCP} ", "dir_a": "192.168.1.64"}
```

- **Pipeline para Inyección de Snort a Kibana**

Para continuar con nuestro laboratorio y diseñar un dashboard en Kibana, asignaremos el siguiente contenido al fichero /etc/logstash/conf.d/logstash.conf, antes de proseguir:

```
input {
  file {
    path => ["/var/log/snort_alerts.log"]
    start_position => beginning
  }
  filter {
    grok {
      match => {"message" =>
"%{SYSLOGTIMESTAMP:fecha_y_hora}.%{WORD:maquina}.%{WORD:nombre_ids}[%{INT:proceso}]..%{GREEDYDATA:
mensaje}.%{WORD:protocolo}.%{IP:dir_a}....%{IP:dir_b}"}
      match => {"message" =>
"%{SYSLOGTIMESTAMP:fecha_y_hora}.%{WORD:maquina}.%{WORD:nombre_ids}[%{INT:proceso}]..%{GREEDYDATA:
mensaje}.%{WORD:protocolo}.%{IP:dir_a}:%{INT:puerto_a}....%{IP:dir_b}:%{INT:puerto_b}"}
    }
    output {
      elasticsearch {
        hosts => [ "http://192.168.1.21:9200" ]
        index => "logstash"
      }
    }
  }
}
```

1.7.- VISUALIZACIÓN SIEM.

A continuación monitorizaremos con Kibana la información que Logstash ha capturado de Snort, filtrado con Grok, e insertado en la base de datos de Elasticsearch.

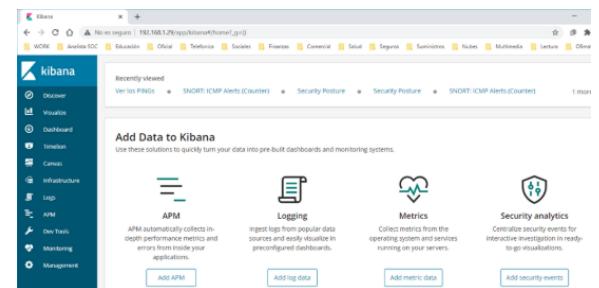
Para ello, localizaremos la información en Elasticsearch a partir del índice creado por Logstash, añadiremos métricas basadas en dicho índice, y finalmente crearemos tableros basados en estas métricas.

Hecho esto, generaremos tráfico ICMP y SSH desde otra máquina para ver cómo se detectan dichos ataques y cómo se muestran en el tablero correspondiente.

1.7.1.- ARRANQUE DE KIBANA DESDE EL NAVEGADOR.

Arrancamos un navegador en el PC y escribimos la dirección de la máquina DMZ1 (sin especificar ningún puerto).

Si el stack ELK está correctamente arrancado, se visualizará la Home de Kibana.



1.7.2.- CONFIGURACIÓN DE KIBANA.

Para la configuración de Kibana, usaremos principalmente las siguientes secciones del menú de la página web:

Discover. En esta sección se localizan los índices creados en la base de datos. En nuestro caso, el índice que ha creado y que alimenta nuestro pipeline se denomina "logstash".

Visualize. Sección para la creación de métricas. Aquí definiremos métricas para Ping y para SSH. Programaremos además un refresco periódico (por ejemplo, 5 segundos), porque en caso contrario el tablero sólo efectuará una primera consulta al índice y se detendrá.

Dashboard. Sección para diseño de Tableros o Cuadros de Mando, en función de las métricas definidas en el paso anterior.

1.7.3.- MANEJO DE KIBANA.

El índice "logstash" que hemos creado ya contiene la historia de todos los ataques efectuados desde el momento de su creación.

1. Mediante la opción "Discover" Se abrirá una pantalla en la que podremos ver el índice "logstash" creado en la base de datos Elasticsearch.

2. Seleccionando dicho índice, podremos ver también la información que se está registrando en él a partir del log de Snort.

3. Para crear una visualización a partir de los datos del índice, seleccionamos la opción "Visualize" y pulsamos el botón "Create a visualization"

4. Para empezar, crearemos una visualización sencilla, tipo métrica.

5. Para ello, pulsaremos el botón "Metric".

6. Seleccionamos el índice "logstash"

7. Creamos un contador, para lo cual seleccionaremos la opción "Count".

8. Por el momento hay creado ningún filtro, por lo que se visualizará el recuento de todos los eventos registrados en el índice hasta el momento, independientemente de su naturaleza.

9. Vamos a crear dos contadores, uno para ICMP y otro para eventos TCP, por lo que los denominaremos "Contador PINGs" y "Contador SSH".

10. Para filtrar los eventos ICMP o TCP entre todos los eventos registrados, procedemos a crear un filtro.

11. Para ello, pulsamos la opción "Add a filter".

12. En este caso usaremos parte del texto del mensaje para seleccionar los eventos, por lo que pulsaremos la opción "mensaje.keyword".

13. A continuación, pulsaremos el operador "is".

14. Al pulsar en el campo de contenidos, nos mostrará los diversos contenidos recibidos hasta el momento en el campo "Mensaje".

15. Seleccionaremos el contenido correspondiente en el momento de crear cada uno de los dos contadores para detectar eventos ICMP/ping y TCP/SSH.

16. Hecho esto, quedará guardado el filtro en cuestión dentro de la visualización.

17. Para que empiece a funcionar el filtro, tendremos que pulsar la opción "Refresh", o bien, activar un intervalo de refresco automático, por ejemplo, de 5 segundos.

18. ATENCIÓN: Si el botón "Save" aparece inhibido, pulsar la flechita de "play" y se activará.

19. Pulsamos el botón "Save" para guardar los cambios efectuados.

20. Asignamos un nombre a la visualización creada y pulsamos "Confirm Save".

1.7.4.- CREACIÓN DE UN HISTOGRAMA.

1. Pulsar "Visualize"
2. Pulsar "+" para añadir una visualización
3. Seleccionar "Vertical Bar"
4. Pulsar "logstash"
5. Pulsar la flechita azul de "Y-Axis Count"
6. Se abrirá un submenú, en el que seleccionaremos "Count" como "Aggregation" (generalmente vendrá seleccionado por defecto)
 7. Pondremos nombre al eje Y en el campo "Custom Label" (Número de PINGs / Número de SSHs)
 8. Pulsaremos "Add a Filter" y añadiremos el filtro de la forma habitual
 9. En el submenú "Buckets" pulsaremos X-Axis
 10. En "Aggregation" seleccionaremos la opción "Date Histogram", con Field="@timestamp" e intervalo automático.
 11. Tras lo anterior, pulsaremos la flechita azul de "play"
 12. Si el resultado es satisfactorio se pulsará "Save" en la barra de menú superior (se puede ver la gráfica interactiva si se ataca con ping o con ssh, en función del filtro elegido).

1.7.5.- CREACIÓN DE UN TABLERO.

1. Pulsar "Dashboard"
2. Pulsar "Add" en la barra de menú superior para añadir visualizaciones al tablero
3. Seleccionar secuencialmente todas las visualizaciones deseadas
4. Cerrar el submenú
5. Redimensionar y reposicionar las visualizaciones a gusto del usuario
6. Pulsar "Save" y asignarle un nombre al tablero
7. A partir de este momento, las visualizaciones se podrán mostrar en pantalla de forma independiente, al seleccionarlas dentro de la opción "Visualize", no obstante, lo normal es crear un tablero de monitorización.
8. Vamos a la opción "Dashboard", pulsamos el botón "Create new dashboard" y creamos un tablero denominado "Tablero Principal".
9. A continuación, creamos dos nuevas visualizaciones gráficas con los mismos filtros anteriores y con formato de histograma, totalizando cada 5 minutos.
10. Pulsamos el nombre "Tablero Principal" y añadimos las cuatro visualizaciones, esto es, los dos contadores y los dos histogramas.
11. A partir de ese momento, el tablero quedará operativo y veremos los cambios en tiempo real.

TEST I

1- ¿Cuál es la opción del menú de Kibana que sirve para visualizar los índices de Elasticsearch y la información contenida en ellos?:

- a. Discover.
- b. Visualize.
- c. Dashboard.

2- ¿Cuál es la misión de Kibana en el SOC?:

- a. Filtrado de la información de los logs.
- b. Monitorización de la información.
- c. Detección y Prevención de Intrusiones.
- d. Almacenamiento de la información.

3- ¿En qué módulo del SOC se definen los Pipelines?:

- a. En Kibana.
- b. En Elasticsearch.
- c. En Logstash.

4- ¿En qué módulo del SOC está ubicado el Grok Debugger?:

- a. En Logstash
- b. En Elasticsearch.
- c. En Kibana.

5- ¿Cuál es la misión de Logstash en el SOC?:

- a. Monitorización de la información.
- b. Filtrado de la información de los logs.
- c. Detección y Prevención de Intrusiones.
- d. Almacenamiento de la información.

6- ¿En qué entorno se basa Kibana?:

- a. Node.js.
- b. IBM WebSphere.
- c. Ninguno de los anteriores.
- d. Microsoft .net.

7- ¿Qué se puede comprobar con la aplicación Nmap?:

TEST I 9/10: 1 a), 2 b), 3 c), 4 c), 5 b), 6 a), 7 d), 8 a), 9 a), 10 c)
TEST II 10/10: 1 c), 2 c), 3 a), 4 a), 5 b), 6 a), 7 b), 8 c), 9 c), 10 a)

- a. Ninguna de las anteriores.
- b. Sólo si una aplicación está utilizando un puerto determinado de una dirección IP.
- c. Si una aplicación usa un puerto, se encuentra a la escucha y con qué servicio o protocolo.
- d. Si una aplicación usa un puerto y además está escuchando por él en un momento dado.

8- ¿Qué software se debe instalar en un PC para trabajar con Kibana?:

- a. Ninguno, basta con disponer de un navegador de Internet.
- b. El framework .net.
- c. El cliente de Kibana para PC.

9- ¿En qué formato se presenta la salida del Grok Debugger?:

- a. En JSON.
- b. En UML.
- c. En XML.

10- ¿Para qué sirve el patrón Greedydata?:

- a. Para capturar información en formato hexadecimal.
- b. Para capturar sólo letras, mayúsculas y minúsculas.
- c. Para capturar una cadena de caracteres completa y grabarla en una variable.
- d. Para discriminar entre números y letras.

TEST II

1- ¿En qué consiste la etapa final del proceso en un SIEM?:

- a. En la presentación de información en forma de tableros.
- b. En la presentación de información en forma de métricas e histogramas.
- c. En el análisis de información para detectar patrones y sacar conclusiones de cara a la Prevención de Incidentes.

2- ¿Dónde se ajusta la RAM consumida por la Pila ELK?:

- a. En los ajustes de los Pipelines.
- b. En los ajustes de los ficheros de log.
- c. En las opciones de la Máquina Virtual Java.

3- ¿Cuál es la misión de Elasticsearch en el SOC?:

- a. Almacenamiento de la información.
- b. Detección y Prevención de Intrusiones.
- c. Monitorización de la información.
- d. Filtrado de la información de los logs.

4- ¿Cuál es la opción del menú de Kibana que sirve para crear métricas con los Datos de un índice de Elasticsearch?:

- a. Visualize.
- b. Dashboard.
- c. Discover.

5- ¿Qué módulos de la Pila ELK precisan un ajuste de los límites de memoria RAM?:

- a. Todos los módulos de la pila.
- b. Logstash y Elasticsearch.
- c. Sólo Logstash.
- d. Sólo Kibana.

6- ¿Cómo se presenta la información de Elasticsearch a través del navegador?:

- a. En JSON.
- b. Como un EJB.
- c. En UML.
- d. En XML.
- e. Como un POJO.

7- ¿Cuál es el Comodín en el lenguaje Grok?:

- a. El carácter "asterisco".
- b. El carácter "punto".
- c. El carácter "ampersand".

8- ¿Cuál debe ser el orden de arranque de las aplicaciones en el SIEM ELK?:

- a. Kibana, Logstash y Elasticsearch.
- b. Logstash, Kibana y Elasticsearch.
- c. Se debe arrancar Elasticsearch en primer lugar, el orden del resto de aplicaciones es indiferente.

9- ¿En qué módulo del SOC se utiliza el lenguaje Grok?:

- a. En Kibana.
- b. En Elasticsearch.
- c. En Logstash.

10- ¿En qué módulo del SOC se preparan los Cuadros de Mando?:

- a. En Kibana.
- b. En Elasticsearch.
- c. En Logstash.

En la Unidad 7 hemos estudiado cómo instalar y configurar un SIEM ELK completo, situándolo en la misma red que los diferentes agentes IDS que detectarán las intrusiones y enviarán la información de registros de SNORT a través de Filebeat.

La estructura creada en la Tarea 6, es la presente en cualquier entorno productivo en la que suele haber una sonda Snort en cada una de las máquinas perimetrales, comprometidas, vulnerables, etc., cuya información de logging se ha de redirigir hacia una única máquina en la que estará instalado el SIEM (Elastic Stack).

Pues bien, continuando con el trabajo iniciado en la Tarea 6 asociada a la Unidad 6, en esta tarea abordaremos la lectura y tratamiento del log que centraliza la información de todas las sondas Snort, filtrando su contenido, almacenándolo en la base de datos y preparando un conjunto de visualizaciones que recogeremos en un Tablero de Monitorización Multipunto.

Apartado 1: Instalación y configuración de Elasticsearch.

- a) Detallar la configuración a efectuar en Elasticsearch.
 - b) Prueba de funcionamiento de Elasticsearch.

Descargamos e instalamos Elasticsearch con los siguientes comandos:

Descargamos e instalamos Elasticsearch con las siguientes comandadas:

```
sudo dpkg -i elasticsearch-7.10.2-amd64.deb
```

Iniciamos el servicio y comprobamos su estado.

```
SOC-IC06 [Corriendo] - Oracle VirtualBox
Enlaces ▾
ALBA MOREJON GARCIA
SOC-IC06 [Corriendo] - Oracle VirtualBox
Archivo Maquina Ver Entrada Dispositivos Ayuda
Mar 26 19:05
alba.morejon@SOCIC06:~ alba.morejon@SOCIC06:~
alba.morejon@SOCIC06:~ $ sudo systemctl start elasticsearch
alba.morejon@SOCIC06:~ $ sudo systemctl enable elasticsearch
Synchronizing state of elasticsearch.service with SysV service script with /usr/lib/
systemd/system-sysv-install.
Executing: /usr/lib/systemd/system-sysv-install enable elasticsearch
Created symlink /etc/systemd/system/multi-user.target.wants/elasticsearch.service →
/usr/lib/systemd/system/elasticsearch.service.
alba.morejon@SOCIC06:~ $ sudo systemctl status elasticsearch
● elasticsearch.service - Elasticsearch
  Loaded: loaded (/usr/lib/systemd/system/elasticsearch.service; enabled; preset:
  Active: active (running) since Wed 2025-03-26 19:04:55 UTC; 48s ago
    Docs: https://www.elastic.co
  Main PID: 4700 (java)
     Tasks: 61 (limit: 2873)
    Memory: 1.2G (peak: 1.2G)
       CPU: 26.727s
      CGroup: /system.slice/elasticsearch.service
             └─[4700] /usr/share/elasticsearch/jdk/bin/java -Xshare:auto -Des.networkb
[4876] /usr/share/elasticsearch/modules/x-pack-ml/platform/linux-x86_64

Mar 26 19:04:14 SOCIC06 systemd[1]: Starting elasticsearch.service - Elasticsearch.
Mar 26 19:04:55 SOCIC06 systemd[1]: Started elasticsearch.service - Elasticsearch.
```

Añadimos la siguiente configuración en el fichero `elasticsearch.yml`
para que pueda iniciar y funcionar correctamente: asignamos un nombre al cluster y al nodo, le indicamos las interfaces de red disponibles, le indicamos la dirección y el nodo principal desde el que se actuará.

Comprobamos que esté bien configurado:

Se realiza una solicitud al servidor para obtener información básica sobre el nodo (nombre del nodo y el cluster, versión elasticsearch y otras configuraciones).

```
alba.morejon@SOCIC06: ~ curl -X GET "localhost:9200/"  
{  
    "name" : "node-1",  
    "cluster_name" : "siem-cluster",  
    "cluster_uuid" : "noHTe0JLQzGGuittEcR6dw",  
    "version" : {  
        "number" : "7.10.2",  
        "build_flavor" : "default",  
        "build_type" : "deb",  
        "build_hash" : "747e1cc71def077253878a59143c1f785afa92b9",  
        "build_date" : "2021-01-13T00:42:12.435326Z",  
        "build_snapshot" : false,  
        "lucene_version" : "8.7.0",  
        "minimum_wire_compatibility_version" : "6.8.0",  
        "minimum_index_compatibility_version" : "6.0.0-beta1"  
    },  
    "tagline" : "You Know, for Search"
```

Se realiza una solicitud para tener información sobre el estado del cluster, el número de nodos, número de shards.

```
alba.morejon@SOCIC06:~$ curl -X GET "localhost:9200/_cluster/health?pretty"
{
  "cluster_name": "siem-cluster",
  "status": "green",
  "timed_out": false,
  "number_of_nodes": 1,
  "number_of_data_nodes": 1,
  "active_primary_shards": 0,
  "active_shards": 0,
  "relocating_shards": 0,
  "initializing_shards": 0,
  "unassigned_shards": 0,
  "delayed_unassigned_shards": 0,
  "number_of_pending_tasks": 0,
  "number_of_in_flight_fetch": 0,
  "task_max_waiting_in_queue_millis": 0,
  "active_shards_percent_as_number": 100.0
}
```

Creamos un índice (test.index) y vemos que se haya configurado correctamente. Listamos todos los índices existentes dentro del clúster, mostrando el estado, el nombre el UUID...

```
alba.morejon@SOCIC06:~$ curl -X PUT "localhost:9200/test-index?pretty"
{
  "acknowledged": true,
  "shards_acknowledged": true,
  "index": "test-index"
}
alba.morejon@SOCIC06:~$ curl -X GET "localhost:9200/_cat/indices?v"
health status index      uuid                                     pri  rep docs.count docs.deleted store.size pri.store.size
yellow open   test-index hAf_LQuARjRTpnlxr0Dew  1    1      0          0
208b        208b
alba.morejon@SOCIC06:~$
```

Pruebas:

Creamos un documento en el índice creado anteriormente, test-index (con los datos del inicio del código)

```
alba.morejon@SOCIC06:~$ curl -X POST "localhost:9200/test-index/_doc/1?pretty" -H 'Content-Type: application/json' -d'
{
  "user": "alba",
  "message": "Elasticsearch está funcionando correctamente",
  "post_date": "2025-03-26"
}
{
  "_index": "test-index",
  "_type": "_doc",
  "_id": "1",
  "_version": 1,
  "result": "created",
  "shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "seq_no": 0,
  "_primary_term": 1
}
```

Realizamos una búsqueda en el índice buscando el documento anterior

```
alba.morejon@SOCIC06:~$ curl -X GET "localhost:9200/test-index/_search?q=user:alba&pretty"
{
  "took": 76,
  "timed_out": false,
  "shards": [
    {
      "total": 1,
      "successful": 1,
      "skipped": 0,
      "failed": 0
    }
  ],
  "hits": [
    {
      "total": {
        "value": 1,
        "relation": "eq"
      },
      "max_score": 0.2876821,
      "hits": [
        {
          "_index": "test-index",
          "_type": "_doc",
          "_id": "1",
          "_score": 0.2876821,
          "source": {
            "user": "alba",
            "message": "Elasticsearch está funcionando correctamente",
            "post_date": "2025-03-26"
          }
        }
      ]
    }
}
```

Agregamos otro documento en el mismo índice:

```
alba.morejon@SOCIC06:~$ curl -X POST "localhost:9200/test-index/_doc/2?pretty" -H 'Content-Type: application/json' -d
{
  "user": "juan",
  "message": "Probando Elasticsearch",
  "post_date": "2025-03-26"
}

{
  "_index": "test-index",
  "_type": "doc",
  "_id": "2",
  "_version": 1,
  "_result": "created",
  "_shards": {
    "total": 1,
    "successful": 1,
    "failed": 0
  },
  "_seq_no": 1,
  "_primary_term": 2
}
```

Utilizamos el siguiente comando para contar cuantos documentos existen de cada usuario

```
alba.morejon@SOCIC06:~$ curl -X GET "localhost:9200/test-index/_search?pretty" -H 'Content-Type: application/json' -d
{
  "size": 0,
  "aggs": {
    "users": {
      "terms": {
        "field": "user.keyword"
      }
    }
  }
}

{
  "took": 706,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": {
      "value": 2,
      "relation": "eq"
    },
    "max_score": null,
    "hits": []
  },
  "aggregations": {
    "users": {
      "doc_count_error_upper_bound": 0,
      "sum_other_doc_count": 0,
      "buckets": [
        {
          "key": "alba",
          "doc_count": 1
        },
        {
          "key": "juan",
          "doc_count": 1
        }
      ]
    }
  }
}
```

Obtenemos información y verificamos de nuevo que el clúster esté funcionando correctamente

```
alba.morejon@SOCIC06:~$ curl -X GET "localhost:9200/_cluster/stats?pretty"
{
  "_nodes": {
    "total": 1,
    "successful": 1,
    "failed": 0
  },
  "cluster_name": "siem-cluster",
  "cluster_uuid": "noHTe0JLQzGGuittEcR6dw",
  "timestamp": 1743019500331,
  "status": "yellow",
  "indices": {
    "count": 1,
    "shards": {
      "total": 1,
      "primaries": 1,
      "replication": 0.0,
      "index": {
        "shards": {
          "min": 1,
          "max": 1,
          "avg": 1.0
        },
        "primaries": {
          "min": 1,
          "max": 1,
          "avg": 1.0
        }
      },
      "replication": 0
    }
  }
}
```

Apartado 2: Instalación y configuración de Kibana.

a) Detallar la configuración a efectuar en Kibana.

b) Prueba de acceso al menú principal de Kibana.

Descargamos e instalamos Kibana

Iniciamos el servicio y comprobamos el estado

```
alba.morejon@SOCIC06:~$ wget https://artifacts.elastic.co/downloads/kibana/kibana-7.10.2-and64.deb
--2023-03-26 20:12:20-- https://artifacts.elastic.co/downloads/kibana/kibana-7.10.2-and64.deb
Resolving artifacts.elastic.co (artifacts.elastic.co)... 34.120.127.130
Connecting to artifacts.elastic.co (artifacts.elastic.co)|34.120.127.130|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 253198756 (244M) [application/octet-stream]
Saving to: "kibana-7.10.2-and64.deb"

kibana-7.10.2-and64.d 100%[=====] 241.47M 7.29MB/s in 35s
2023-03-26 20:12:20 [kibana-7.10.2-and64.deb] saved [253198756/253198756]

alba.morejon@SOCIC06:~$ sudo dpkg -i kibana-7.10.2-and64.deb
[sudo] password for alba.morejon:
Selecting previously unselected package kibana.
(Reading database ... 150248 files and directories currently installed.)
Preparing to unpack kibana-7.10.2-and64.deb ...
Unpacking kibana (7.10.2) ...
Setting up kibana (7.10.2) ...
```

```
alba.morejon@SOCIC06:~$ sudo systemctl start kibana
[sudo] password for alba.morejon:
alba.morejon@SOCIC06:~$ sudo systemctl enable kibana
Synchronizing state of kibana.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable kibana
Created symlink /etc/systemd/system/multi-user.target.wants/kibana.service → /etc/systemd/system/kibana.service.
alba.morejon@SOCIC06:~$ sudo systemctl status kibana
● kibana.service - Kibana
   Loaded: loaded (/etc/systemd/system/kibana.service; enabled; preset: enabled)
     Active: active (running) since Wed 2023-03-26 20:34:04 UTC; 15s ago
       Main PID: 6248 (node)
          Tasks: 11 (limit: 4669)
        Memory: 334.6M (peak: 334.8M)
         CPU: 7.735s
      CGroup: /system.slice/kibana.service
           └─ 6248 /usr/share/kibana/bin/../node/bin/node /usr/share/kibana/bin/ ./src/main.js

Mar 26 20:34:14 SOCIC06 kibana[6248]: {"type": "log", "@timestamp": "2023-03-26T20:34:14Z", "source": "elasticsearch", "offset": 1000000000000000000, "index": "test-index", "type": "log", "host": "localhost:5601/app/home#"}
Mar 26 20:34:14 SOCIC06 kibana[6248]: {"type": "log", "@timestamp": "2023-03-26T20:34:14Z", "source": "elasticsearch", "offset": 1000000000000000000, "index": "test-index", "type": "log", "host": "localhost:5601/app/home#"}
Mar 26 20:34:14 SOCIC06 kibana[6248]: {"type": "log", "@timestamp": "2023-03-26T20:34:14Z", "source": "elasticsearch", "offset": 1000000000000000000, "index": "test-index", "type": "log", "host": "localhost:5601/app/home#"}
Mar 26 20:34:14 SOCIC06 kibana[6248]: {"type": "log", "@timestamp": "2023-03-26T20:34:14Z", "source": "elasticsearch", "offset": 1000000000000000000, "index": "test-index", "type": "log", "host": "localhost:5601/app/home#"}
```

Editamos el archivo de configuración kibana.yml, añadimos el puerto en el que la herramienta está escuchando, configuramos todas las interfaces de red que escuchará, especificamos la dirección Elasticsearch al que se conectará y define el nombre del índice en Elasticsearch donde Kibana almacenará sus datos.

```
alba.morejon@SOCIC06:~$ nano 7.2 /etc/kibana/kibana.yml
# Kibana is served by a back end server. This setting specifies the port to use.
server.port: 5601
server.host: "0.0.0.0"
elasticsearch.hosts: ["http://localhost:9200"]
kibana.index: "kibana"
# Set this to the address to which the Kibana server will bind to addressees and hosts.
```

Comprobamos que funcione correctamente navegando en la web:
http://localhost:5601

- Cargamos la página
- Creamos el patrón de índice para poder gestionarlo (elegimos el índice creado con Elasticsearch)

- En la sección visualize, creamos una nueva visualización con los datos del índice text-index
- Creamos un Dashboard

The left screenshot shows the Elasticsearch interface with the 'Visualize' section open. A 'Count' visualization is displayed for the 'test-index' index. The right screenshot shows a dashboard being created, with a single visualization panel containing a 'Count' visualization for 'All docs'.

Apartado 3: Instalación y configuración de Filebeat.

a) Detallar la configuración a efectuar en el agente IDS para enviar los registros de Snort a Logstash.

b) Mostrar una prueba de funcionamiento de Filebeat mostrando los registros por consola.

Descargamos e instalamos Filebeat

Iniciamos el servicio

```
alba.morejon@SOCIC06:~$ wget https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.10.2-amd64.deb
--2025-03-26 21:20:00-- https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.10.2-amd64.deb
Resolving artifacts.elastic.co (artifacts.elastic.co)... 34.120.127.130
Connecting to artifacts.elastic.co (artifacts.elastic.co)|34.120.127.130|:443...
HTTP request sent, awaiting response... 200 OK
Length: 34274384 (33M) [application/octet-stream]
Saving to: 'filebeat-7.10.2-amd64.deb'

HTTP/2.0 200 [=====] 32.69M 8.09MB/s in 4.7s
2025-03-26 21:20:06 (6.99 MB/s) - 'filebeat-7.10.2-amd64.deb' saved [34274384/34274384]

alba.morejon@SOCIC06:~$ sudo dpkg -i filebeat-7.10.2-amd64.deb
[sudo] password for alba.morejon:
Selecting previously unselected package filebeat.
(Reading database ... 207348 files and directories currently installed.)
Preparing to unpack filebeat-7.10.2-amd64.deb ...
Unpacking filebeat (7.10.2) ...
Setting up filebeat (7.10.2) ...
alba.morejon@SOCIC06:~$
```

Configuraremos Filebeat para que pueda leer los registros de Snort (habilitamos la entrada de logs, con la ruta de archivos que debe leer) y enviarlos Logstash (especificando la dirección del servidor al que el servicio mandará los registros, utilizando esta misma máquina en el puerto 5044).

Habilitamos el módulo de Filebeat para Snort

```
alba.morejon@SOCIC06:~$ nano /etc/filebeat/filebeat.yml
# configuration file.

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

filebeat.inputs:
- type: log
  enabled: true
  paths:
  - /var/log/snort/*.log

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

```
alba.morejon@SOCIC06:~$ sudo systemctl start filebeat
alba.morejon@SOCIC06:~$ sudo systemctl enable filebeat
Synchronizing state of filebeat.service with SysV service script with /usr/lib/systemd/system-sysv-install.
Executing: /usr/lib/systemd/system-sysv-install enable filebeat
Created symlink /etc/systemd/system/multi-user.target.wants/filebeat.service → /usr/lib/systemd/system/filebeat.service.
alba.morejon@SOCIC06:~$ 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 Wed 2025-03-26 21:22:08 UTC; 19s ago
     Docs: https://www.elastic.co/products/beats/filebeat
          >Main PID: 12055 (filebeat)
             Tasks: 8 (limit: 4609)
            Memory: 24.1M (peak: 24.8M)
              CPU: 139ms
            CGroup: /system.slice/filebeat.service
                    └─12055 /usr/share/filebeat/bin/filebeat --environment systemd -c /etc/fil...
```

```
alba.morejon@SOCIC06:~$ sudo nano /etc/filebeat/filebeat.yml
alba.morejon@SOCIC06:~$ sudo filebeat modules enable snort
Module snort is already enabled
alba.morejon@SOCIC06:~$
```

Configuraremos el módulo de Snort en el archivo snort.yml. Verificamos que los puertos estén abiertos

The image shows two side-by-side Linux desktop environments. The left window is titled "SOC-IC06 [Corriendo] - Oracle VirtualBox" and displays a terminal session for "alba.morejon@SOCIC06 ~". The terminal shows the configuration of the "snort" module in filebeat, specifying log files under "/var/log/snort/*.log". The right window is titled "SOC-IC06 [Corriendo] - Oracle VirtualBox" and shows a terminal session for "alba.morejon@SOCIC06 ~". It displays netstat and ufw commands to check for port 5044 listening on both TCP and UDP protocols, and to verify the status of the ufw firewall.

Hacemos una prueba de verificar la configuración y probamos la salida de Logstash. Y Ejecutamos Filebeat en modo depuración para ver los registros

Apartado 4: Instalación y configuración de Logstash.

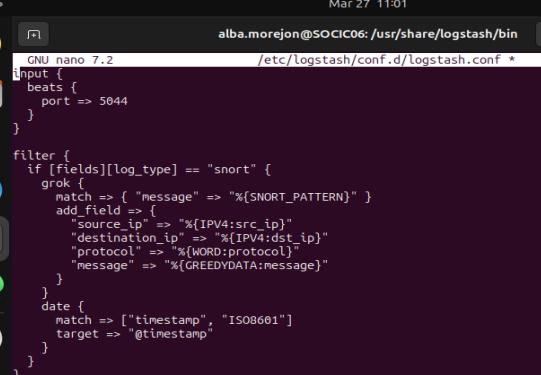
- a) Detallar la configuración a efectuar en Logstash para recibir los logs de Filebeat y reenviarlos a Elasticsearch.
 - b) Mostrar una prueba de funcionamiento en la que se puede visualizar la información del índice de logstash en Kibana.

Agregamos la clave de Elasticshared y el repositorio de Logstash. Instalamos Logstash, iniciamos y habilitamos el servicio Logstash comprobando que su estado correcto

```
alba.morejon@SOC-IC06: ~
```

```
alba.morejon@SOC-IC06: $ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo gpg --dearmail -r -o /usr/share/keyrings/elastic.gpg
[sudo] password for alba.morejon:
File '/usr/share/keyrings/elastic.gpg' exists. Overwrite? (y/N) y
alba.morejon@SOC-IC06: $ echo "deb [signed-by=/usr/share/keyrings/elastic.gpg] https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-7.x.list
[downloaded 7.0M]
alba.morejon@SOC-IC06: $ sudo apt update
alba.morejon@SOC-IC06: $ sudo apt install logstash
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
logstash is already the newest version (1:7.17.28-1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
alba.morejon@SOC-IC06: $ sudo systemctl start logstash
alba.morejon@SOC-IC06: $ sudo systemctl enable logstash
alba.morejon@SOC-IC06: $ sudo systemctl status logstash
● logstash.service - logstash
   Loaded: loaded (/etc/systemd/system/logstash.service; enabled; preset: enabled)
     Active: active (running) since Wed 2025-03-26 22:46:31 UTC; 7min ago
       Main PID: 916 (java)
          Tasks: 40 (limit: 4669)
            Memory: 728.9M (peak: 756.5M)
           CPU: 1min 7.072s
```

Añadimos la configuración para Filebeat



```
GNU nano 7.2                               /etc/logstash/conf.d/logstash.conf *  
input {  
    beats {  
        port => 5044  
    }  
}  
  
filter {  
    if [fields][log_type] == "snort" {  
        grok {  
            match => { "message" => "%{SNORT_PATTERN}" }  
            add_field => {  
                "source_ip" => "%{IPV4:src_ip}"  
                "destination_ip" => "%{IPV4:dst_ip}"  
                "protocol" => "%{WORD:protocol}"  
                "message" => "%{GREEDYDATA:message}"  
            }  
            date {  
                match => [ "timestamp", "ISO8601" ]  
                target => "@timestamp"  
            }  
        }  
    }  
}  
  
output {  
    elasticsearch {  
        hosts => [ "http://localhost:9200" ]  
        index => "snort-logs-%{+YYYY_MM_DD}"  
    }  
}
```

Verificamos que la configuración no tenga errores

```
alba.morejon@SOCIC06:~$ sudo systemctl restart logstash
alba.morejon@SOCIC06:~$ cd /usr/share/logstash/bin/
alba.morejon@SOCIC06:/usr/share/logstash/bin$ sudo ./logstash --config.test_and_exit -f /etc/logstash/conf.d/logstash.conf
Using bundled JDK: /usr/share/logstash/jdk
WARNING: Could not find logstash.yml which is typically located in $LS_HOME/config or /etc/logstash. You can specify the path using --path.settings. Continuing using the defaults
Could not find log4j2 configuration at path /usr/share/logstash/config/log4j2.properties. Using default config which logs errors to the console
[WARN] 2025-03-27 10:51:36.706 [main] runner - Starting from version 9.0, running with superuser privileges is not permitted unless you explicitly set 'allow_superuser' to true, thereby acknowledging the possible security risks
[INFO] 2025-03-27T11:11:52,120 [main] WARN runner - Starting from version 9.0, running with superuser privileges is not permitted unless you explicitly set 'allow_superuser' to true, thereby acknowledging the possible security risks
[INFO] 2025-03-27T11:11:52,144 [main] INFO runner - Log4j configuration path used is: /usr/share/logstash/config/log4j2.properties
[WARN] 2025-03-27T11:11:52,145 [main] WARN runner - 'pipeline.buffer.type' setting is not explicitly defined. Before moving to 9.x set it to 'heap' and tune heap size upward, or set it to 'direct' to maintain existing behavior.
[INFO] 2025-03-27T11:11:52,148 [main] INFO runner - Starting Logstash {"logstash.version"=>"8.17.4", "jruby.version"=>"jruby 9.4.9.0 (3.1.4) 2024-11-04 547c6b150e OpenJDK 64-Bit Server VM 21.0.6+7", "LTS on 21.0.6+7-LTS +indy +jit [x86_64-linux]"}
[INFO] 2025-03-27T11:11:52,153 [main] INFO runner - JVM bootstrap flags: [-Xms1g, -Xmx1g, -Djava.awt.headless=true, -Dfile.encoding=UTF-8, -Druby.compile.invokedynamic=true, -XX+HeapDumpOnOutOfMemoryError, -Djava.security.egd=file:/dev/urandom, -Dlog4j2.isThreadContextMapInheritable=true, -Dlogstash.jackson.stream.read.constraints.max_string_length=2000000000, -Dlogstash.jackson.stream.r
```

Le añadimos la siguiente información a los ficheros logstash.yml

```
alba.morejon@SOCIC06:~$ nano /etc/logstash/logstash.yml
# -----
# Node identity -----
# Use a descriptive name for the node:
# node.name: test
node.name: "logstash-node"
path.data: "/var/lib/logstash"
path.logs: "/var/log/logstash"
pipeline.workers: 2
pipeline.batch.size: 125
pipeline.batch.delay: 50
```

```
alba.morejon@SOCIC06:~$ nano /etc/logstash/logstash.yml
node.name: "logstash-node"
path.data: "/var/lib/logstash"
path.logs: "/var/log/logstash"
pipeline.workers: 2
pipeline.batch.size: 125
pipeline.batch.delay: 50
```

Creamos el fichero log4j2.properties y añadimos lo siguiente

```
alba.morejon@SOCIC06:~$ nano /usr/share/logstash/config/log4j2.properties
status = error
name = LogstashPropertiesConfig
appender.console.type = Console
appender.console.name = console
appender.console.layout.type = PatternLayout
appender.console.layout.pattern = "%d{ISO8601} [%t] %-5p %c{1} %marker - %m%n"
rootLogger.level = error
rootLogger.appenderRefs = console
rootLogger.appenderRef.console.ref = console
```

Ejecutamos logs en modo depuración para ver los registros de consola

```
alba.morejon@SOCIC06:~$ sudo /usr/share/logstash/bin/logstash --path.settings /usr/share/logstash/config -f /etc/logstash/conf.d/logstash.conf -log.level debug
Using bundled JDK: /usr/share/logstash/jdk
Sending Logstash logs to /var/log/logstash which is now configured via log4j2.properties
"2025-03-27T11:11:52,120 [main] WARN runner - Starting from version 9.0, running with superuser privileges is not permitted unless you explicitly set 'allow_superuser' to true, thereby acknowledging the possible security risks
"2025-03-27T11:11:52,144 [main] INFO runner - Log4j configuration path used is: /usr/share/logstash/config/log4j2.properties
"2025-03-27T11:11:52,145 [main] WARN runner - 'pipeline.buffer.type' setting is not explicitly defined. Before moving to 9.x set it to 'heap' and tune heap size upward, or set it to 'direct' to maintain existing behavior.
"2025-03-27T11:11:52,148 [main] INFO runner - Starting Logstash {"logstash.version"=>"8.17.4", "jruby.version"=>"jruby 9.4.9.0 (3.1.4) 2024-11-04 547c6b150e OpenJDK 64-Bit Server VM 21.0.6+7", "LTS on 21.0.6+7-LTS +indy +jit [x86_64-linux]"}
"2025-03-27T11:11:52,153 [main] INFO runner - JVM bootstrap flags: [-Xms1g, -Xmx1g, -Djava.awt.headless=true, -Dfile.encoding=UTF-8, -Druby.compile.invokedynamic=true, -XX+HeapDumpOnOutOfMemoryError, -Djava.security.egd=file:/dev/urandom, -Dlog4j2.isThreadContextMapInheritable=true, -Dlogstash.jackson.stream.read.constraints.max_string_length=2000000000, -Dlogstash.jackson.stream.r
```

Editamos el archivo filebeat.yml y vemos los logs

```
GNU nano 7.2          /etc/filebeat/filebeat.yml *
# For more available modules and options, please see the filebeat.reference.yml sample
# configuration file.

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

filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/log/snort/*
  fields:
    log_type: snort

output.logstash:
  hosts: ["localhost:5044"]

logging.level: debug
logging.to_files: true
logging.files:
  path: /var/log/filebeat
  name: filebeat
  keepfiles: 7
  permissions: 0644
```

```
root@SOCIC06:/var/log/filebeat# sudo tail -f /var/log/filebeat/filebeat
2025-03-27T11:42:34.462Z      DEBUG  [input] log/input.go:226      input
states cleaned up. Before: 0, After: 0, Pending: 0
2025-03-27T11:42:37.037Z      DEBUG  [input] input/input.go:139     Run in
put
2025-03-27T11:42:44.462Z      DEBUG  [input] input/input.go:139     Run in
put
2025-03-27T11:42:44.462Z      DEBUG  [input] log/input.go:205      Start
next scan
2025-03-27T11:42:44.462Z      DEBUG  [input] log/input.go:226      input
states cleaned up. Before: 0, After: 0, Pending: 0
2025-03-27T11:42:47.038Z      DEBUG  [input] input/input.go:139     Run in
```

Apartado 5: Creación de un filtro en Logstash.

a) Detallar la configuración a efectuar en Logstash para crear un Pipeline que filtre la información creando campos para los diferentes valores del mensaje de log creado en Snort.

b) Mostrar el índice que se crea con la información de sus diferentes campos.

Configuramos el fichero logstash.conf

```
GNU nano 7.2          /etc/logstash/conf.d/logstash.conf *
input {
  beats {
    port => 5044
  }
}

filter {
  if [fields][log_type] == "snort" {
    grok {
      match => { "message" => "%{SNORT_PATTERN}" }
      add_field => [
        "source_ip" => "%{IPV4:src_ip}"
        "destination_ip" => "%{IPV4:dst_ip}"
        "protocol" => "%{WORD:protocol}"
        "message" => "%{GREEDYDATA:message}"
      ]
    }
    date {
      match => ["timestamp", "ISO8601"]
      target => "@timestamp"
    }
  }
}

output {
  elasticsearch {
    hosts => ["http://localhost:9200"]
    index => "snort-logs-%{+YYYY.MM.dd}"
  }
}
```

Creamos el índice snort-logs de forma manual

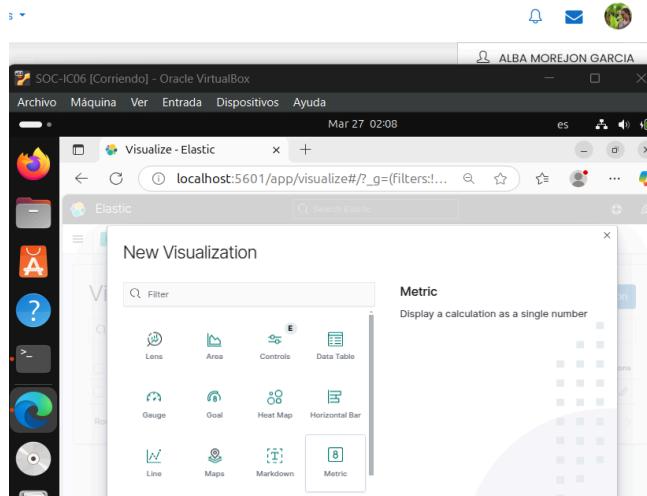
```
PUT snort-logs_000001
2: { "settings":{ ...
3:   "number_of_shards": 1,
4:   "number_of_replicas":1
5: }, ...
6:   "mappings":{ ...
7:     "properties":{ ...
8:       "timestamp":{ ...
9:         "type": "date"
10:      },
11:      "source_ip":{ ...
12:        "type": "ip"
13:      },
14:      "destination_ip":{ ...
15:        "type": "ip"
16:      },
17:      "protocol":{ ...
18:        "type": "keyword"
19:      },
20:      "message":{ ...
21:        "type": "text"
22:      }
23:    }
24:  }
25: }
26: }
```

Name	Type	Format	Search...	Aggregations	Excluded
_id	string				
_index	string				
_score	number				
_source	_source	_source			
_type	string				
destination_ip	ip	IP address			
message	string				
protocol	string				
source_ip	ip	IP address			

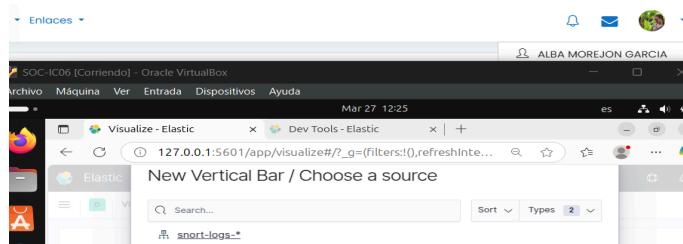
Apartado 6: Tablero de Monitorización Multipunto.

- a) Detallar la creación de dos contadores, uno para todos los PINGs desde la red interna y otro para los de la red externa.
- b) Detallar la creación de dos histogramas, uno para los intentos de inicio de sesión por SSH y otro para los accesos a PHPMyadmin.
- c) Detallar la creación de un nuevo tablero (dashboard) en Kibana que contenga los contadores y los histogramas creados anteriormente.

Seleccionando en el menú lateral el apartado Visualize (dentro de Kibana). y pulsamos en “Create visualization”. Nos da la opción de nos da la opción de seleccionar Metric y procedemos a crear dos visualizaciones para establecer red donde van a actuar.



Elegimos el índice que creamos



Establecemos los valores de Count y le damos el rango de IPs donde actuará

Haremos lo mismo pero cambiando el rango para la red externa

A continuación crearemos otras visualizaciones de tipo Histograma para los inicios de sesión con SSH y PHPMyadmin dando los siguientes valores

SOC-IC06 [Corriendo] - Oracle VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

Mar 27 12:31 es

Visualize - Elastic Dev Tools - Elastic

Elastic

Search Elastic

message: "SSH login attempt" KQL Off Refresh

No results found

snort-logs-*

SOC-IC06 [Corriendo] - Oracle VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

Mar 27 12:32 es

Visualize - Elastic Dev Tools - Elastic

Elastic

Search Elastic

Metrics > Y-axis Count

Buckets > X-axis

Aggregation Date Histogram

Field timestamp

Minimum Interval Day

Custom label Histograma Intentos SSH

SOC-IC06 [Corriendo] - Oracle VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

Mar 27 12:36 es

Visualize - Elastic Histograma Intentos SSH

Elastic

message: "PHPMyAdmin access" KQL Off Refresh

Count

All docs

snort-logs-*

Data Metrics & axes Panel settings

Metrics > Y-axis Count

Creamos un nuevo Dashboard, seleccionando en el menú lateral “Dashboard” y haciendo click sobre “Create Dashboard” y añadimos las vistas creadas anteriormente

SOC-IC06 [Corriendo] - Oracle VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

Mar 27 12:40 es

Visualize - Elastic Dashboards - Elastic

Elastic

Dashboard Editing New Dashboard Options Share Add Cancel Save Create new

Add panels

Search Sort Types Create new

Contador PINGs Red Externa

Contador PINGs Red Interna

Histograma Intentos SSH

PHPMyAdmin

Add an existing or new object to this dashboard

Create new

SOC-IC06 [Corriendo] - Oracle VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

Mar 27 12:38 es

Visualize - Elastic IC07-Elastic

Elastic

Dashboard IC07

Contador PINGs Red Externa

0 192.168.11 to 192.168.1.100 - Count

Contador PINGs Red Interna

0 10.0.2.1 to 10.0.2.10 - Count

Histograma Intentos SSH

No results found

PHPMyAdmin

No results found

Dashboard YC07 was saved