

Infraguard: Despliegue Y protección

De Infraestructuras con Docker



Centro: IES La Vereda

Autor: Emanuel Rojas Collazos

2024

Tutor: José Manuel Romero Martínez

Índice

1	Introducción	4
1.1	¿Qué es un SIEM?	4
1.2	¿Qué es un contenedor de Docker?.....	4
1.3	Objetivos del proyecto	4
2	Introducció	5
2.1	Què és un SIEM?	5
2.2	¿Què és un contenidor de Docker?.....	5
2.3	Objectius del projecte	5
3	Introduction	6
3.1	What is a SIEM?.....	6
3.2	What is a Docker container?.....	6
3.3	Project objectives	6
4	Módulos que implica	7
4.1	Implantación de Sistemas Operativos (ISO).....	7
4.2	Planificación y Administración de Redes (PAR)	7
4.3	Administración de Sistemas Gestores de Bases de Datos (ASGBD)	7
4.4	Seguridad y Alta Disponibilidad (SAD)	7
4.5	Servicios de Red e Internet (SRI).....	8
5	Escenario Inicial	8
5.1	Estudio Previo	8
5.2	Plan de Trabajo	9
5.3	Topología De Red	9
6	Fundamentos teóricos y conceptos.....	11
6.1	¿Qué es Wazuh?.....	11
6.2	Funciones	11
6.3	¿Qué es una red Perimetral?	12
6.4	Tecnologías y funcionalidades	12
7	Preparación del sistema.....	13
7.1	Creación de Particiones	13
7.2	Instalación y configuración del Sistema Operativo	14
8	Despliegue y Configuración de la red Perimetral	16
8.1	Instalación y configuración de Docker	16

8.2	Configuración de los Host de Docker	18
8.3	Reglas de Iptables.....	18
8.4	Filtrar servicios	21
8.5	Despliegue y configuración de Wazuh	22
8.6	Creación del Agente de Wazuh.....	25
8.7	Instalación del Agente de Wazuh	28
9	Monitoreo de Alertas en Wazuh.....	31
9.1	Instalación de Un Paquete.....	31
9.2	Hacer un Restart al agente de Wazuh.....	33
10	Documentación Github.....	34
11	Recursos Utilizados	34
11.1	Hardware.....	34
11.2	Software	34
11.3	Sistemas Operativos	35
12	Implementación en una empresa.....	35
13	Conclusión	36
13.1	Problemas encontrados	36
13.2	Mejoras	37
14	Bibliografía/Webgrafía.....	38
14.1	Documentación	38
14.2	Videos	38

1 Introducción

En un mundo donde la tecnología está en auge, se nos hace necesario implementar medidas de seguridad. Tanto para pymes como grandes empresas, la información o los datos de cada una de ellas se vuelven cada vez más relevantes. Y la protección y detección ante amenazas se vuelve una máxima prioridad. Por eso cada una de ellas se decanta por implementar en su sistema de seguridad un SIEM.

1.1 ¿Qué es un SIEM?

Es un Sistema de gestión de Eventos e Información de Seguridad (*Security Information and Event Management*) es un tipo de software creado para poder gestionar, analizar y administrar los diversos hosts que se encuentran en una red determinada, con la finalidad de detectar y responder a posibles riesgos y amenazas de seguridad. Así monitorizar cada uno de ellos desde los accesos no autorizados hasta si un equipo se desconecta de la red. Por lo cual la herramienta SIEM que utilizaremos será Wazuh.

1.2 ¿Qué es un contenedor de Docker?

Un contenedor de Docker es una unidad de software ligera y portátil que comprime una aplicación y todos sus repositorios y dependencias, incluidas las bibliotecas y otros archivos necesarios para ejecutarla, en un entorno virtualizado y optimizado ya que estas ocupan muy poco espacio en disco.

1.3 Objetivos del proyecto

Este proyecto tiene como objetivo principal implementar un entorno de SIEM utilizando el despliegue de la herramienta Wazuh en una red perimetral, aprovechando las ventajas de la virtualización con Docker y la gestión simplificada de contenedores con Docker Compose. Esta arquitectura nos permite una fácil escalabilidad, gestión y despliegue del SIEM, garantizando una respuesta rápida y efectiva ante incidentes de seguridad. Utilizando como sistema operativo para monitorizar Ubuntu y como base de despliegue de la herramienta SIEM el sistema operativo Linux.

2 Introducció

En un món on la tecnologia està en auge, se'ns fa necessari implementar mesures de seguretat. Tant per a pymes com grans empreses, la informació o les dades de cadascuna d'elles es tornen cada vegada més rellevants. I la protecció i detecció davant amenaces es torna una màxima prioritat. Per això cadascuna d'elles es decanta per implementar en el seu sistema de seguretat un SIEM.

2.1 Què és un SIEM?

És un Sistema de gestió d'Esdeveniments i Informació de Seguretat (Security Information and Event Management) és un tipus de programari creat per a poder gestionar, analitzar i administrar els diversos hosts que es troben en una xarxa determinada, amb la finalitat de detectar i respondre a possibles riscos i amenaces de seguretat. Així monitorar cadascun d'ells des dels accessos no autoritzats fins a si un equip es disconnecta de la xarxa. Per la qual cosa l'eina SIEM que utilitzarem serà Wazuh.

2.2 ¿Què és un contenidor de Docker?

Un contenidor de Docker és una unitat de programari lleugera i portàtil que comprimeix una aplicació i tots els seus repositoris i dependències, incloses les biblioteques i altres arxius necessaris per a executar-la, en un entorn virtualitzat i optimitzat ja que aquestes ocupen molt poc espai en disc.

2.3 Objectius del projecte

Aquest projecte té com a objectiu principal implementar un entorn de SIEM utilitzant el desplegament de l'eina Wazuh en una xarxa perimetral, aprofitant els avantatges de la virtualització amb Docker i la gestió simplificada de contenidors amb Docker Compose. Aquesta arquitectura ens permet una fàcil escalabilitat, gestió i desplegament del SIEM, garantint una resposta ràpida i efectiva davant incidents de seguretat. Utilitzant com a sistema operatiu per a monitorar Ubuntu i com a base de desplegament de l'eina SIEM el sistema operatiu Linux.

3 Introduction

In a world where technology is booming, it is necessary for us to implement security measures. For both SMEs and large companies, the information or data of each of them becomes increasingly relevant. And threat protection and detection become a top priority. That is why each of them chooses to implement a SIEM in their security system.

3.1 What is a SIEM?

It is a Security Information and Event Management System is a type of software created to be able to manage, analyze and administer the various hosts that are in a given network, in order to detect and respond to possible security risks and threats. In this way, you can monitor each of them, from unauthorized access to if a computer is disconnected from the network. Therefore, the SIEM tool we will use will be Wazuh.

3.2 What is a Docker container?

A Docker container is a lightweight, portable unit of software that compresses an application and all its repositories and dependencies, including the libraries and other files needed to run it, into a virtualized and optimized environment because they take up very little disk space.

3.3 Project objectives

The main objective of this project is to implement a SIEM environment using the deployment of the Wazuh tool in a perimeter network, taking advantage of virtualization with Docker and simplified container management with Docker Compose. This architecture allows us to easily scalable, manage and deploy the SIEM, ensuring a quick and effective response to security incidents. Using the Linux operating system as an operating system to monitor Ubuntu and as a basis for deploying the SIEM tool.

4 Módulos que implica

4.1 Implementación de Sistemas Operativos (ISO)

Este módulo abarca la instalación, configuración y administración de sistemas operativos.

En el proyecto, se requiere:

- ✚ Crear particiones en un disco duro.
- ✚ Instalar y configurar un sistema operativo Linux (Cinnamon Edition) en dual boot con Windows, utilizando el gestor de arranque GRUB.
- ✚ Instalar y configurar herramientas básicas del sistema operativo (boot-repair para reparar GRUB).

4.2 Planificación y Administración de Redes (PAR)

Este módulo se centra en la configuración y administración de redes, fundamental para:

- ✚ Configurar la red perimetral y las distintas subredes (interna, DMZ, y WAN).
- ✚ Configurar reglas de iptables para enrutar y filtrar el tráfico de red entre diferentes hosts.
- ✚ Crear y gestionar una topología de red, asegurando que los contenedores y hosts pueden comunicarse adecuadamente.

4.3 Administración de Sistemas Gestores de Bases de Datos (ASGBD)

- ✚ Aunque este proyecto no se enfoca en bases de datos, la administración de sistemas que incluyen bases de datos es una competencia relevante para gestionar logs y registros de eventos que genera Wazuh.

4.4 Seguridad y Alta Disponibilidad (SAD)

Este módulo es crucial para:

- ✚ Implementar y configurar Wazuh como una herramienta de SIEM para la detección y gestión de incidentes de seguridad.
- ✚ Monitorizar eventos de seguridad, identificar accesos no autorizados, y gestionar alertas.

4.5 Servicios de Red e Internet (SRI)

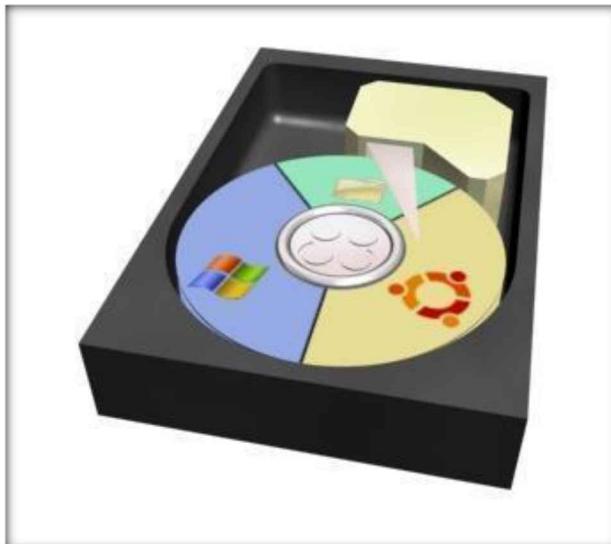
Este módulo incluye la instalación y configuración de servicios de red, vital para:

- ✚ Configurar Docker y Docker Compose, que permiten desplegar Wazuh y otros servicios de monitorización.
- ✚ Configurar servicios de red como enruteadores y firewalls dentro de un entorno virtualizado.

5 Escenario Inicial

Implica en desplegar una herramienta SIEM la cual será Wazuh en local, a través de Docker Compose en una red perimetral. Este despliegue se realiza en un entorno donde el sistema operativo principal es Linux, pero para ello crearemos una partición de un disco duro SSD M2 256 GB la cual será de 50GB y esta partición de disco su sistema de archivos por defecto será Ext4.

Una vez creada la Partición procederemos a instalar el sistema operativo correspondiente el cual será Linux. Y configuraremos un arranque dual con el Grub el cual es un cargador de arranque que nos permitirá elegir entre los sistemas operativos que haya instalados en el sistema.



5.1 Estudio Previo

En un principio iba a utilizar herramientas de monitorización que cada una hacia una cosa diferente como por ejemplo Prometheus y Grafana para la monitorización del sistema, recopilando métricas y generando dashboards para la visualización.

Decidí trabajar con Docker y no máquinas virtuales ya que podía trabajar en un entorno virtualizado y optimizado ya que su software es ligero y no gasta muchos recursos.

Pero nos dimos cuenta con mi tutor que ya existen herramientas que traen todas estas funciones y muchas más como un SIEM. Me puse a investigar qué tipo de herramientas había y que fueran open-source de software libre y nos decantamos por Wazuh.

Cuando ya había elegido la herramienta que iba a utilizar me tocaba investigar de cómo hacer para poder desplegarla. Y me di cuenta de que hay muy poca documentación e información de cómo hacerlo. Así que me lo tome como un reto.

5.2 Plan de Trabajo

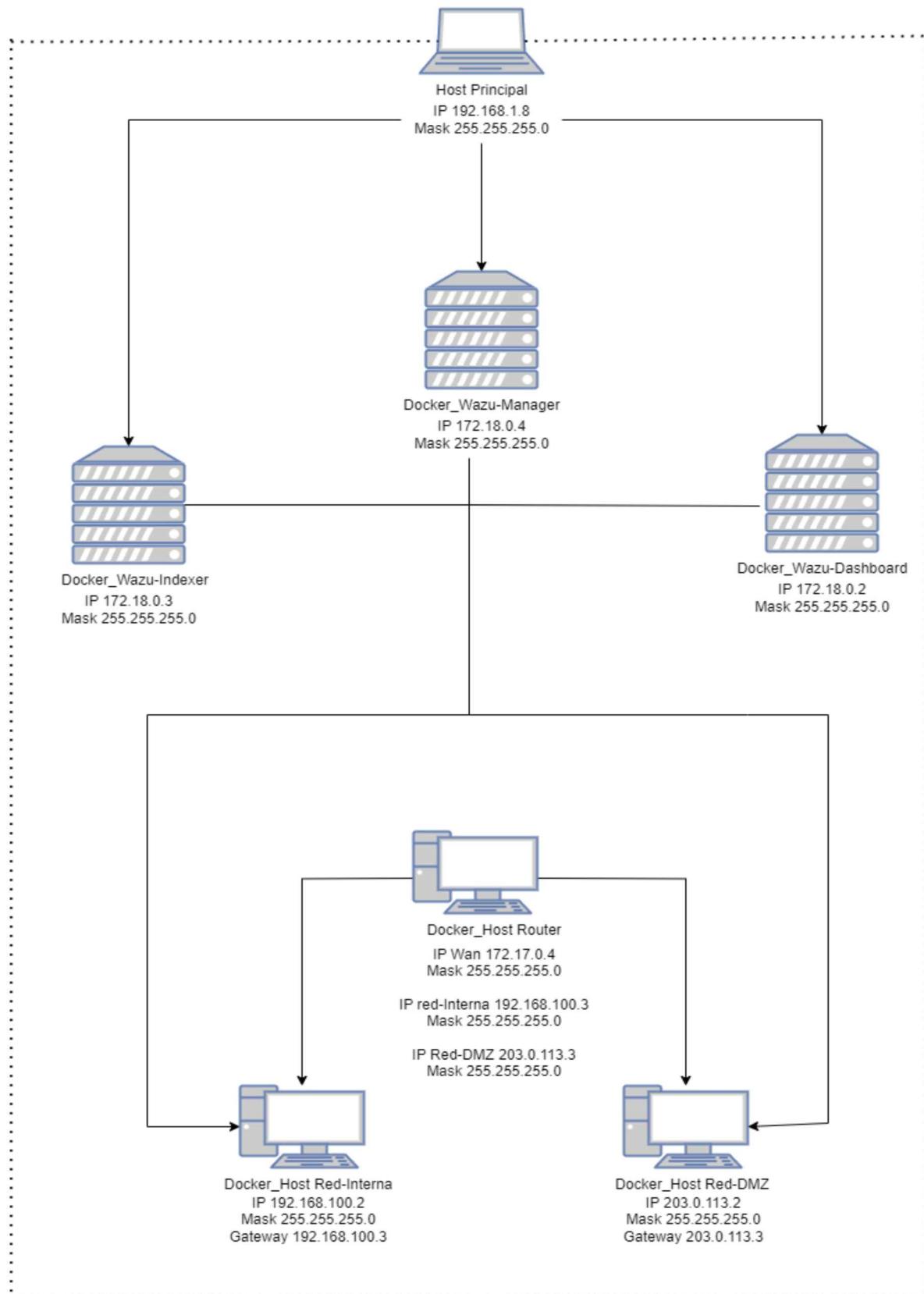


5.3 Topología De Red

En la siguiente gráfica, podemos observar una configuración compleja que involucra seis contenedores de Docker. Tres de estos contenedores son instancias de Wazuh, desplegadas localmente utilizando Docker-Compose. Los otros tres contenedores ejecutan imágenes de Ubuntu y se inician con el comando Docker RUN.

Los tres contenedores de Wazuh están configurados para proporcionar una solución integral de monitorización y seguridad. Desplegados con Docker-Compose, estos contenedores trabajan en conjunto para asegurar un entorno robusto y eficiente:

- ✚ Wazuh Manager: Responsable de gestionar la recopilación y el análisis de datos de seguridad.
- ✚ Wazuh Indexer: Encargado de indexar los datos de seguridad, facilitando búsquedas y análisis rápidos.
- ✚ Wazuh Dashboard: Proporciona una interfaz gráfica para la visualización y gestión de los datos de seguridad.



6 Fundamentos teóricos y conceptos.

6.1 ¿Qué es Wazuh?

Wazuh es un sistema para detectar intrusos. La mayoría de los sistemas operativos, incluidos Linux, AIX, HP-UX, macOS, Solaris y Windows, ofrecen detección de intrusiones. Wazuh, con su arquitectura multiplataforma y centralizada, facilita la gestión y monitoreo de múltiples sistemas.

Es una solución de monitorización de seguridad gratuita y de código abierto que ayuda a las empresas a detectar amenazas, vigilar la integridad, responder a incidentes y cumplir.



y errores del sistema.

Se encarga de proporcionar un conjunto de gráficas y vistas predefinidas y listas para usar, los cuales están creados para casos de uso estándar como el cumplimiento y monitorización de la seguridad. Estas graficas brindan datos sobre eventos de seguridad comunes, intentos de acceso fallidos, detección de malware

6.2 Funciones

- ✚ Detección anticipada: Wazuh se adelanta al identificar y advertir sobre actividades sospechosas o maliciosas en tiempo real, brindando a los equipos de seguridad la oportunidad de responder de inmediato ante las amenazas, previniendo posibles consecuencias negativas.
- ✚ Relación de Eventos: Mediante la combinación y examen de información de diferentes fuentes, Wazuh puede identificar vínculos entre eventos que parecen no estar conectados, generando alertas más exactas y disminuyendo los falsos positivos de manera significativa.
- ✚ Regulación Cumplimiento: Wazuh facilita el cumplimiento de normativas de seguridad al revisar y analizar actividades, brindando un valioso apoyo para demostrar que la infraestructura se ajusta a las leyes en vigor.

- ✚ Visualización y Reportes: Kibana ofrece una visualización clara y adaptable de los eventos de seguridad a través de su interfaz, lo que permite a los analistas crear

informes detallados y comprender mejor el panorama de amenazas correspondiente.

Wazuh se distingue por su gran capacidad de crecer y ajustarse a las necesidades particulares de cada empresa a través de reglas y configuraciones personalizadas. Esto garantiza una administración de seguridad efectiva y adaptada a las características específicas de cada ambiente empresarial.

6.3 ¿Qué es una red Perimetral?

Una red perimetral es una estructura de defensa esencial para proteger la integridad, confidencialidad y disponibilidad de los recursos internos de una organización frente a amenazas externas.

Consiste en hacer un filtrado de quien puede y quien no puede acceder a la red interna.

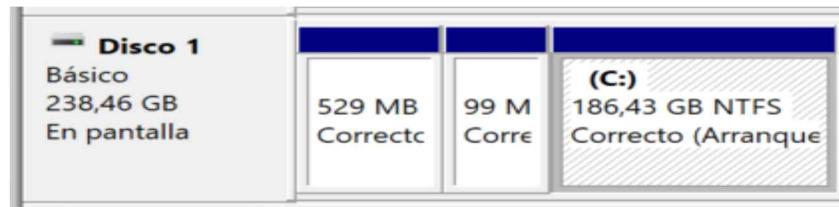
6.4 Tecnologías y funcionalidades

- Los Firewalls: Son dispositivos que permiten o deniegan el paso en función de una serie de reglas que especifican qué tipo de tráfico está o no permitido.
- Enrutadores fronterizos: Se trata de equipos de seguridad perimetral que actúan como si se tratase de señales de tráfico que dirigen y controlan la circulación de las redes, filtrando todo aquello que no sea confiable.
- Protección de subredes: Pequeñas redes que contienen servicios públicos de control perimetral conectados directamente y que ofrecen protección por el firewall u otros dispositivos de filtrado.
- Sistemas de detección de intrusos (IDS): Es un sistema de alarma que sirve para detectar y avisar sobre actividades sospechosas.
- Sistemas de prevención de intrusiones (IPS): Estos sistemas mejoran la seguridad perimetral en redes, ya que no solo detectan intrusos, sino que ponen en marcha, de inmediato, acciones para defender el sistema, como por ejemplo técnicas de bloqueo automático de tráfico malicioso.

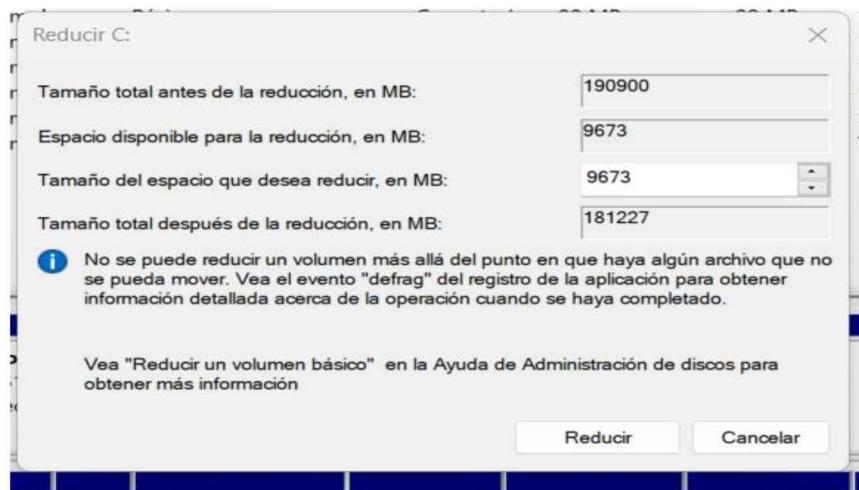
7 Preparación del sistema

7.1 Creación de Particiones

Como tenemos un disco duro SSD M2 256 GB en el cual ya tenemos instalado Windows vamos a crear una partición de 50 GB. Con el software que trae Windows “Crear y Formatear Particiones del Disco Duro”.



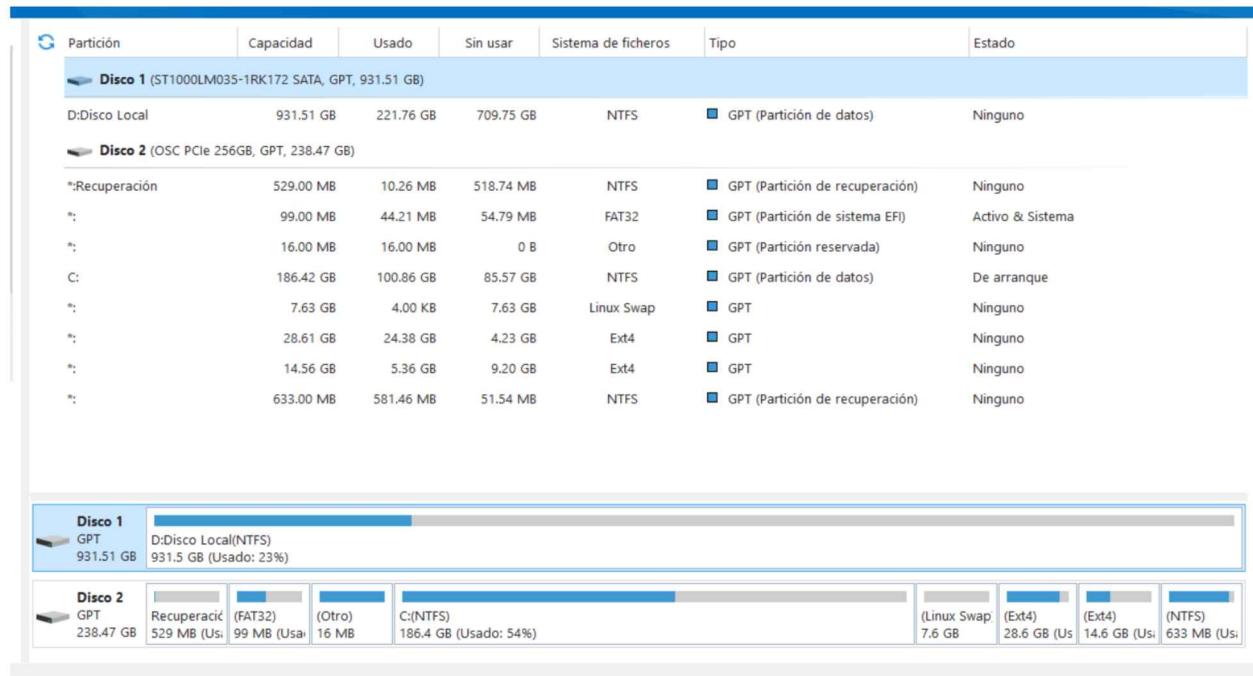
Pero al momento de crear la partición solo nos deja de 9 GB porque Windows por



defecto no nos deja ya que está configurado para proteger los datos que ya hay en el disco duro y así no se corrompan o se pierdan. Como lo veremos en la siguiente imagen.

Para ello utilizamos el siguiente programa para hacer la partición “MiniTool Partition Wizard” y aquí seleccionamos el disco duro lo particionamos y nos deja ver cómo queda y nos permite cambiarle el sistema de archivos a la partición el cual paso de NTFS a Ext4.

Y una vez hecho clicamos en guardar cambios y nos queda de la siguiente manera.



7.2 Instalación y configuración del Sistema Operativo

Una vez creada la partición preparamos un USB con la imagen de Linux “Cinamon Edition” seguimos todos los pasos necesarios para instalarlo, le seleccionamos la partición en la cual lo instalaremos.

Una vez vaya terminando de instalar el sistema detectará que en el disco duro hay otro sistema operativo en el cual debemos decir que no formatee esa parte del disco, procederá a descargar el GRUB. Una vez ya instalado nos pide que reiniciemos que ya todo está listo.

Pero por temas de versiones y repositorios el GRUB se ha instalado mal con errores. Por lo cual al momento de reiniciar te va a parecer el GRUB para poder seleccionar el sistema operativo con que quieres iniciar.

Pero Una vez apagues y vuelvas a iniciar no va a volver aparecer.

Para ello antes de reiniciar debemos de descargar a través de comandos la siguiente aplicación “boot-reapair” para poder arreglar el GRUB.

- Primero vamos a descargar los repositorios de la aplicación.

```
sudo add-apt-repository ppa:yannubuntu/boot-repair
```

- Una vez hecho esto esperamos a que se instale y actualizamos los repositorios con el comando.

```
sudo apt update
```

- ⊕ Una vez actualizado el sistema procedemos a instalar la aplicación.

```
sudo apt-get install boot-repair -y
```

- ⊕ Una vez ya instalada Procedemos a ejecutarla y hacemos clic en la primera opción.

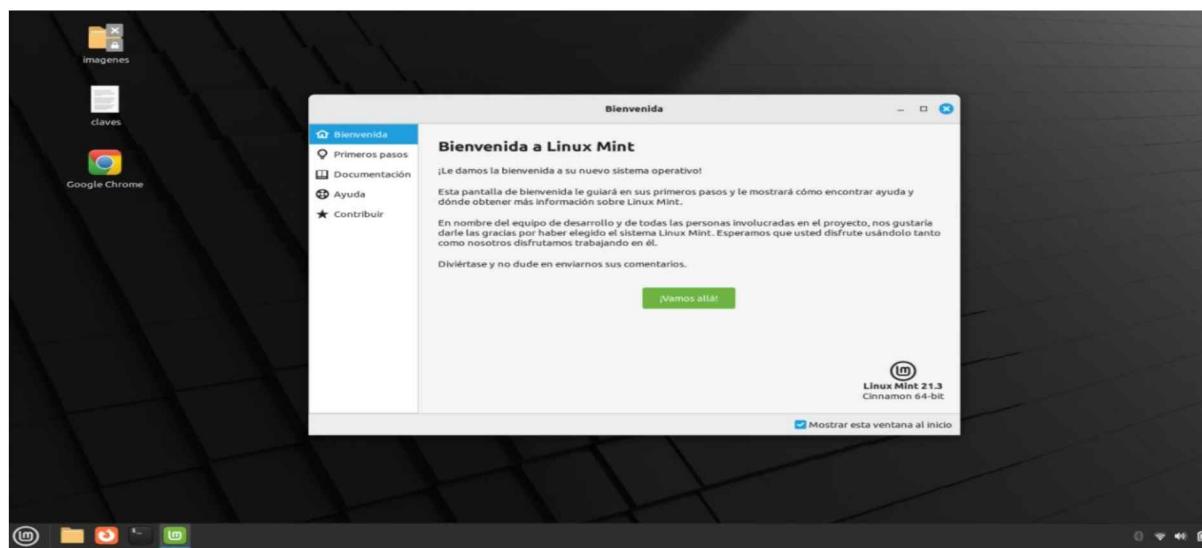
```
boot-repair
```

- ⊕ Nos preguntara si queremos crear un reporte seleccionaremos no.

- ⊕ Una vez hecho esto procedemos a esperar a que la aplicación termine de arreglar el GRUB cuando haya terminado nos saldrá este letrero.
- ⊕ Una vez hecho esto procedemos a reiniciar el equipo y nos saldrá el GRUB así para poder seleccionar el sistema operativo que queramos el cual será en este caso Linux.



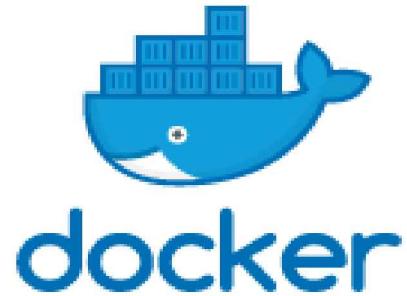
- ⊕ Una vez seleccionado esperamos que arranque y vemos que arranca perfectamente.



8 Despliegue y Configuración de la red Perimetral

8.1 Instalación y configuración de Docker

Para poder desplegar nuestra herramienta Wazuh primero debemos instalar en nuestro sistema Docker y Docker Compose. Por consiguiente, mostrare todos los pasos a seguir para que nuestro Docker quede bien configurado y el servicio este activo sin errores.



- ✚ Lo primero que haremos es ejecutar el siguiente código para remover o desinstalar cualquier dependencia de Docker.

```
for pkg in docker.io docker-doc docker-compose podman-docker containerd runc; do sudo apt-get remove $pkg; done
```

- ✚ Una Vez hecho esto procedemos a actualizar el sistema, instalar los siguientes paquetes y repositorios oficiales de Docker.

```
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl https://download.docker.com/linux/debian/gpg | gpg --keyring /etc/apt/keyrings/docker.asc --import
sudo chmod a+r /etc/apt/keyrings/docker.asc
```

- ✚ Una vez hecho esto añadimos los repositorios a APT

```
echo \ "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/debian \
$(. /etc/os-release && echo "$VERSION_CODENAME")
stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

- ✚ Una vez hecho esto volvemos a Actualizar el sistema, y comenzamos a descargar los paquetes, pero vemos que al momento que el comienza a descargar los paquetes que le hemos pasado da error como que no los encuentra y esto es por qué estamos descargando una versión que no es compatible con nuestro sistema operativo.

```
Package 'docker-ce' has no installation candidate
Unable to locate package docker-ce-cli
Unable to locate package containerd.io
Couldn't find any package by glob 'containerd.io'
Couldn't find any package by regex 'containerd.io'
Unable to locate package docker-buildx-plugin
Unable to locate package docker-compose-plugin
```

- ✚ Para corregir esto nos debemos de ir a la siguiente ruta /etc/apt/sources.list.d y editar el archivo docker.list

```
sudo nano /etc/apt/sources.list.d/docker.list
```

- ✚ Una vez aquí debemos remplazar el victoria por el jammy

```
docker.list *
signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu jammy stable
```

- ✚ Una vez hecho esto guardamos e instalamos los paquetes de Docker y vemos que ahora si se nos instalan.

```
sudo apt-get install docker-ce docker-ce-cli containerd.io
docker-buildx-plugin docker-compose-plugin
```

- ✚ Vemos que ahora si se ha comenzado a instalar perfectamente.

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  docker-ce-rootless-extras libslirp0 pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli
  docker-ce-rootless-extras docker-compose-plugin libslirp0 pigz slirp4netns
0 upgraded, 9 newly installed, 0 to remove and 11 not upgraded.
Need to get 114 MB of archives.
After this operation, 414 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 https://download.docker.com/linux/ubuntu jammy/stable amd64 containerd.io
amd64 1.6.22-1 [28,3 MB]
0% [Connecting to archive.ubuntu.com (91.189.91.83)] [1 containerd.io 15,9 kB/2]
```

- ⊕ Una vez ya instalado arrancamos el servicio de Docker y que se inicie automáticamente en el arranque del sistema.

```
sudo service docker start
sudo systemctl enable docker
```

8.2 Configuración de los Host de Docker

Host	Tarjeta Red	IP	Puerta de enlace
Red-interna	192.168.100.0/24	192.168.100.2	192.168.100.3
Red-DMZ	203.0.113.0/24	203.0.113.2	203.0.113.3
Router	172.17.0.2/24 192.168.100.0/24 203.0.113.0/24	172.17.0.2 203.0.113.3 192.168.100.3	172.17.0.1

8.3 Reglas de Iptables

El host router tiene unas reglas de iptables para poder enrutar y capar el tráfico de red entre los hosts y que estos al igual puedan salir a internet. Para ello hemos hecho las siguientes configuraciones.

- ⊕ Lo primero es que debemos de instalar las iptables en nuestro host si no están instaladas.

```
apt install iptables
```

- ⊕ Una vez hecho esto debemos habilitar el ip_forwarding para que el sistema actúe como un enrutador, permitiendo que los paquetes de red que recibe en una interfaz sean reenviados a otra interfaz.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

- ⊕ Una vez hecho esto procedemos a configurar las reglas de iptables.

- Configurar reglas de NAT (Network Address Translation)

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

- ⊕ Hay que tener en cuenta que esta regla es la que nos deja salir a internet con la red de Docker predeterminada.
- Configurar reglas de enruteamiento para la WAN (eth2).

```
iptables -A FORWARD -i eth2 -o eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

- Configurar reglas de enrutamiento para la DMZ (eth1) y red interna (eth2)

```
iptables -A FORWARD -i eth2 -o eth0 -j ACCEPT
iptables -A FORWARD -i eth0 -o eth1 -j ACCEPT
```

- ✚ Una vez hecho esto nos vamos a cada uno de los hosts y cambiamos su puerta predeterminada de enlace por las ip de cada una de las tarjetas de red del router tanto de la red interna como la de DMZ.
- ✚ Para ello utilizamos el siguiente comando para ver la puerta de enlace por defecto del host.

```
ip route show default
```

- ✚ Si vemos que en la puerta de enlace no tiene la IP del router correspondiente a su red, lo que hacemos es eliminarle su puerta de enlace por defecto.

```
ip route del default
```

- ✚ Una vez hecho esto agregamos la puerta de enlace correspondiente.

```
ip route add default via 203.0.113.3
```

- ✚ Una vez hecho esto ya tendríamos configurado nuestro router y comprobamos que podemos hacer ping de un host a otro.

Red Interna:

```
root@hots-redinterna:/# ping 203.0.113.2
PING 203.0.113.2 (203.0.113.2) 56(84) bytes of data.
64 bytes from 203.0.113.2: icmp_seq=1 ttl=63 time=0.308 ms
64 bytes from 203.0.113.2: icmp_seq=2 ttl=63 time=0.139 ms
64 bytes from 203.0.113.2: icmp_seq=3 ttl=63 time=0.086 ms
64 bytes from 203.0.113.2: icmp_seq=4 ttl=63 time=0.093 ms
^C
--- 203.0.113.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3074ms
rtt min/avg/max/mdev = 0.086/0.156/0.308/0.089 ms
root@hots-redinterna:/# █
```

Red DMZ:

```
root@host-redDMZ:/# ping 192.168.100.2
PING 192.168.100.2 (192.168.100.2) 56(84) bytes of data.
64 bytes from 192.168.100.2: icmp_seq=1 ttl=63 time=0.210 ms
64 bytes from 192.168.100.2: icmp_seq=2 ttl=63 time=0.120 ms
64 bytes from 192.168.100.2: icmp_seq=3 ttl=63 time=0.132 ms
^C
--- 192.168.100.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2042ms
rtt min/avg/max/mdev = 0.120/0.154/0.210/0.039 ms
root@host-redDMZ:/# █
```

Ya tendríamos bien configuradas nuestras reglas de Iptables, pero tenemos un inconveniente y es que cada vez que nos salgamos de la máquina, nuestras reglas de IP se borrarán por ello debemos hacerlas persistentes para ello crearemos un script.

El cual lo crearemos en la siguiente ruta “/etc/rc.local” el cual quedara de la siguiente manera.

```
root@host-redDMZ:/          root@hots-redinterna:/  
GNU nano 6.2  
#!/bin/sh -e  
#  
# rc.local  
#  
# This script is executed at the end of each multiuser runlevel.  
# Make sure that the script will "exit 0" on success or any other  
# value on error.  
#  
# In order to enable or disable this script just change the execution  
# bits.  
#  
# By default this script does nothing.  
/sbin/iptables -P FORWARD ACCEPT  
/sbin/iptables -P INPUT DROP  
/sbin/iptables -P OUTPUT DROP  
/sbin/iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE  
/sbin/iptables -A FORWARD -i eth2 -o eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT  
/sbin/iptables -A FORWARD -i eth2 -o eth0 -j ACCEPT  
/sbin/iptables -A FORWARD -i eth0 -o eth1 -j ACCEPT  
exit 0
```

- Una vez hecho esto le daremos permisos de ejecución.

```
chmod +x /etc/rc.local
```

- Una vez hecho esto lo recargamos la configuración de los servicios sin detenerlos o reiniciarlos.

```
systemctl daemon-reload
```

- Una vez hecho esto ponemos en marcha nuestro servicio. Con “/etc/rc.local” y presionando enter. Y de esta forma se cargarán todas nuestras reglas de iptables cada vez que ingresemos al sistema.
- Una vez hecho esto si ejecutamos el siguiente comando podremos ver las reglas de iptables configuradas en el sistema y así vemos que nuestro script ha funcionado.

```
Iptables -L
```

```
root@prueba_router:/# /etc/rc.local
root@prueba_router:/# iptables -L
Chain INPUT (policy DROP)
target    prot opt source          destination
ACCEPT    all  --  anywhere        anywhere
ACCEPT    all  --  anywhere        anywhere
ACCEPT    all  --  anywhere        anywhere

Chain FORWARD (policy ACCEPT)
target    prot opt source          destination
ACCEPT    all  --  anywhere        anywhere          state RELATED,ESTABLISHED
ACCEPT    all  --  anywhere        anywhere
ACCEPT    all  --  anywhere        anywhere

Chain OUTPUT (policy DROP)
target    prot opt source          destination
root@prueba_router:/# service rc-local status
rc-local: unrecognized service
root@prueba_router:/# service status rc-local
status: unrecognized service
root@prueba_router:/#
```

8.4 Filtrar servicios

Una vez hecho esto vamos a permitir que la red interna pueda conectarse por medio de SSH a la DMZ, pero de la DMZ a la red interna no, para ello necesitamos configurar una regla de iptables directamente en el host de la red interna.

```
iptables -A INPUT -i eth0 -p tcp --dport 22 -j REJECT
```

```
root@hots-redinterna:/# iptables -L
Chain INPUT (policy ACCEPT)
target    prot opt source          destination
REJECT   tcp  --  anywhere        anywhere          tcp dpt:ssh reject-with icmp-port-unreachable

Chain FORWARD (policy ACCEPT)
target    prot opt source          destination

Chain OUTPUT (policy ACCEPT)
target    prot opt source          destination
root@hots-redinterna:/#
```

Lo que hace esta regla es bloquear todos los intentos de conexión SSH que lleguen a la interfaz eth0, enviando una respuesta de rechazo al origen del intento de conexión.

Esta regla originalmente la deberíamos colocar en el router, pero he decidido configurarla en el host directamente de la red interna ya que la he colocado en el router y no ha funcionado. También hay que tener en cuenta que la hemos hecho persistente como en las reglas del router.

Hay que tener en cuenta que se ha instalado previamente el servicio de SSH en cada uno de los hosts tanto en el de la red interna como el de la DMZ.

SSH de Red DMZ a Interna:

- ✚ Vemos que nos rechaza la conexión directamente.

```
root@host-redDMZ:/# ssh emanuel@192.168.100.2
ssh: connect to host 192.168.100.2 port 22: Connection refused
root@host-redDMZ:/#
```

SSH de Red Interna a DMZ:

- ✚ Vemos que nos acepta la conexión e ingresamos correctamente.

```
root@hots-redinterna:/# ssh emanuel2@203.0.113.2
emanuel2@203.0.113.2's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-101-generic x86_64)

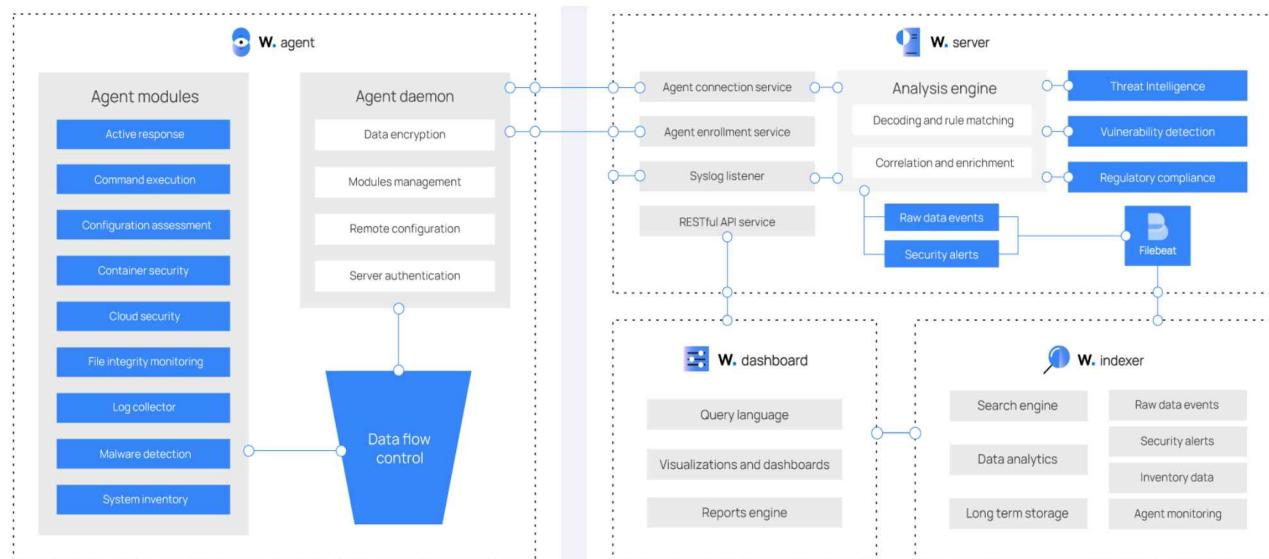
 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Sat May  4 10:50:15 2024 from 192.168.100.2
emanuel2@host-redDMZ:~$ exit
logout
Connection to 203.0.113.2 closed.
```

8.5 Despliegue y configuración de Wazuh

La arquitectura modular y escalable de Wazuh, desplegada mediante Docker Compose para cada uno de sus componentes principales como lo es el indexer, la dashboard y el management representa un enfoque avanzado y flexible en la protección de sistemas y la gestión de incidentes de seguridad.



- Una vez que ya hemos instalado Docker procederemos a instalar en nuestro sistema el GIT y una vez hecho esto clonaremos un repositorio donde estarán los deploys para poder desplegar nuestra herramienta.

```
sudo apt install git -y
git clone https://github.com/wazuh/wazuh-docker.git -b v4.7.4
```

- Una vez ya tengamos los repositorios clonados procedemos a entrar en la dirección /wazuh-docker/single-node y dentro de esta ejecutamos el siguiente código. Para que nos cree los certificados correspondientes para cada uno de los contenedores el indexer, el manager y el de dashboard.

```
docker-compose -f generate-indexer-certs.yml run --rm generator
```

```
emmanuel@proyecto:~/wazuh-docker/single-node$ ls
config docker-compose.yml generate-indexer-certs.yml README.md
emmanuel@proyecto:~/wazuh-docker/single-node$
```

- Una vez hecho esto dentro de la carpeta de single-node estará nuestro deploy que será docker-compose.yml. El cual dentro tendrá cada uno de los servicios que se van a desplegar. Procederemos a desplegar los servicios en modo local pero dentro de un contenedor con el comando y vemos que se despliegan correctamente sin ningún error.

```
docker-compose up -d
```

```
emmanuel@proyecto:~/wazuh-docker/single-node$ docker-compose up -d
Creating network "single-node_default" with the default driver
Creating single-node_wazuh.indexer_1 ... done
Creating single-node_wazuh.manager_1 ... done
Creating single-node_wazuh.dashboard_1 ... done
```

- Una vez hecho esto sí que remos comprobar que los servicios se han desplegado correctamente ejecutamos el siguiente comando nos mostrara una lista de los Dockers activos.

```
docker ps -a
```

```
5bd35329152d    wazuh/wazuh-dashboard:4.4.1    "/entrypoint.sh"      3 weeks ago   Up 7 minutes
        443/tcp, 0.0.0.0:443->5601/tcp, :::443->5601/tcp
                                                               single-node_wazuh.dashboard_1
9180e0982331    wazuh/wazuh-manager:4.4.1     "/init"            3 weeks ago   Up 7 minutes
        0.0.0.0:1514-1515->1514-1515/tcp, :::1514-1515->1514-1515/tcp, 0.0.0.0:514->514/udp, :::514->514/ud
        p, 0.0.0.0:55000->55000/tcp, :::55000->55000/tcp, 1516/tcp
                                                               single-node_wazuh.manager_1
23f5c09029c9    wazuh/wazuh-indexer:4.4.1    "/entrypoint.sh open..."  3 weeks ago   Up 7 minutes
        0.0.0.0:9200->9200/tcp, :::9200->9200/tcp
                                                               single-node_wazuh.indexer_1
emmanuel@proyecto:~$
```

- Una vez vemos que los servicios ya están arrancados correctamente, nos vamos a la dashboard de Wazuh para poder gestionar y configurar nuestra herramienta de monitorización. Nos vamos al navegador y colocamos nuestra IP local en este caso la 192.168.1.8 o también la del local host 127.0.0.1.



- Una vez aquí para ingresar a la configuración debemos de colocar el siguiente usuario y contraseña.

*Usuario: admin *Contraseña: SecretPassword

- Una vez hecho esto si le damos a "Log in", nos mostrara la siguiente pantalla que se empezaran a cargar todos los servicios y debemos esperar a que todos estén ok. Si hay alguno no ok es porque hay un error de configuración en el deploy al momento de desplegar el servicio o de pronto es que el servicio se ha quedado colgado para ello debemos parar los servicios y volver a desplegarlos.

A screenshot of a Mozilla Firefox browser window showing the Wazuh health check results. The title bar says "Wazuh — Mozilla Firefox". The address bar shows the URL "https://127.0.0.1/app/wazuh#/health-check". The main content area displays a list of checks with their status: "Check Wazuh API connection" (green checkmark), "Check Wazuh API version" (yellow circle), "Check alerts index pattern" (green checkmark), "Check monitoring index pattern" (green checkmark), "Check statistics index pattern" (green checkmark), "Check timeline:max_buckets setting" (green checkmark), "Check metaFields setting" (green checkmark), and "Check timepicker:timeDefaults setting" (green checkmark).

- Una vez hecho eso cargarán los servicios y nos redirigirá a la página principal de Wazuh donde podremos ver todos los servicios y los agentes que administramos, pero por el momento no tenemos ninguno ya que no hemos agregado ningún agente y nos pide agregarlo donde dice "Add Agent".

The screenshot shows the Wazuh Overview page in Mozilla Firefox. At the top, it displays agent statistics: Total agents (0), Active agents (0), Disconnected agents (0), Pending agents (0), and Never connected agents (0). Below this, a message says "No agents were added to this manager. Add agent". The page is divided into several sections:

- SECURITY INFORMATION MANAGEMENT:** Includes "Security events" (Browse through your security alerts, identifying issues and threats in your environment) and "Integrity monitoring" (Alerts related to file changes, including permissions, content, ownership and attributes).
- AUDITING AND POLICY MONITORING:** Includes "Policy monitoring" (Verify that your systems are configured according to your security policies baseline) and "System auditing" (Audit users behavior, monitoring command execution and alerting on access to critical files).
- THREAT DETECTION AND RESPONSE:** (This section is partially visible)
- REGULATORY COMPLIANCE:** (This section is partially visible)

8.6 Creación del Agente de Wazuh

- Una vez clicamos donde dice "Add Agent" nos redirigirá a una página donde crearemos un deploy del agente correspondiente para el sistema operativo que queramos auditar, la versión y si es de 32 o 64 bits.

The screenshot shows the "Deploy a new agent" configuration page in Mozilla Firefox. It consists of four numbered steps:

- Choose the operating system:** Options include Red Hat Enterprise, CentOS, Ubuntu (selected), Windows, and macOS. A "Show more" link is available.
- Choose the version:** Options include Ubuntu 14 and Ubuntu 15+ (Ubuntu 15+ is selected).
- Choose the architecture:** Options include i386, x86_64 (selected), armhf, and aarch64.
- Wazuh server address:** A text input field with placeholder text: "This is the address the agent uses to communicate with the Wazuh server. It can be an IP address or a fully qualified domain name (FQDN)".

- ✚ Luego nos pedirá la dirección IP del servidor de Wazuh para que puedan intercomunicarse, la cual en nuestro caso es la 192.168.1.8.

```
emanuel@proyecto:~/wazuh-docker/single-node$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default 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: enp2s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether d4:5d:64:64:e2:f0 brd ff:ff:ff:ff:ff:ff
3: wlp4s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether d8:c0:a6:0e:e8:89 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.8/24 brd 192.168.1.255 scope global dynamic noprefixroute wlp4s0
        valid_lft 85797sec preferred_lft 85797sec
    inet6 fe80::a435:2420:c4df:b8e3/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

4 Wazuh server address

This is the address the agent uses to communicate with the Wazuh server. It can be an IP address or a fully qualified domain name (FQDN).

192.168.1.28

- ✚ Una vez hecho esto podemos definirle un nombre y un grupo al agente. En nuestro caso solo agregaremos el nombre y el grupo lo dejamos por defecto. Ya que si queremos que este dentro de un grupo específico para poder organizarlo mejor debemos primero antes de crear el deploy del agente crear el grupo.

5 Assign a name and a group to the agent

host-redinterna

Select one or more existing groups

Select group

- Una vez hecho esto la misma herramienta nos da un que en realidad es un deploy el cual debemos ejecutar en el equipo que queremos monitorizar que está en la red.

6 Install and enroll the agent

You can use this command to install and enroll the Wazuh agent.

ⓘ If the installer finds another Wazuh agent in the system, it will upgrade it preserving the configuration.

```
curl -so wazuh-agent.deb https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.4.1-1_amd64.deb
&& sudo WAZUH_MANAGER='192.168.1.8' WAZUH_AGENT_NAME='host-redinterna' dpkg -i ./wazuh-agent.deb
```

```
curl -so wazuh-agent.deb
https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-
agent/wazuh-agent_4.4.1-1_amd64.deb && sudo
WAZUH_MANAGER='192.168.1.8' WAZUH_AGENT_NAME='host-
redinterna' dpkg -i ./wazuh-agent.deb
```

Explicación del Deploy:

→ Primera parte:

curl lo que hace es descargar el archivo del paquete de Wazuh Agent desde la url especificada

curl -s: Nos indica que debe ejecutarse en modo silencioso en el cual no mostrara ninguna barra de progreso.

curl -o: Nos indica el nombre de salida del archivo en este caso “Wazuh-agent.deb”.

→ Segunda Parte:

Establece dos variables de entorno las cuales son la IP del servidor de Wazuh “192.168.1.8” y el nombre del agente a instalar “host-redinterna”.

→ Tercera Parte:

Utiliza el comando dpkg -i para instalar el paquete .deb “Wazuh-agent.deb”.

8.7 Instalación del Agente de Wazuh

- Para comenzar a ejecutar el deploy de instalación del agente debemos de recordar instalar antes la herramienta “curl” en el contenedor de Docker que queremos monitorizar. Y debemos editar el deploy ya que en el entorno de Docker no es necesario utilizar el “sudo” que son los permisos de super usuario ya que el usuario predeterminado es root y ya tiene estos privilegios o permisos.

```
curl -so wazuh-agent.deb
https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-
agent/wazuh-agent_4.4.1-1_amd64.deb &&
WAZUH_MANAGER='192.168.1.8' WAZUH_AGENT_NAME='host-
redinterna' dpkg -i ./wazuh-agent.deb
```

- Una vez hecho esto procedemos a ejecutar el despliegue. Es mejor utilizar Ubuntu debido a problemas con los repositorios y dependencias en Debian que nos impiden la instalación del agente. Sin embargo, antes de ejecutar el despliegue en Ubuntu, debemos ejecutar los siguientes comandos para solucionar problemas similares que pueden presentarse en esta distribución.
- Debemos de ejecutar el siguiente comando para instalar el paquete “lsb-release” ya que es una herramienta utilizada para obtener información sobre la distribución del sistema operativo.

```
apt install lsb-release
```

- Una vez hecho ejecutamos el siguiente comando para para corregir dependencias de paquetes rotas o paquetes parcialmente instalados.

```
apt --fix-broken install
```

- Una vez hecho esto ejecutamos el deploy y vemos que no nos da ningún error.

```
root@host-redinternal_1:/# curl -so wazuh-agent.deb https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.4.1-1_amd64.deb && WAZUH_MANAGER='192.168.1.8' WAZUH_AGENT_NAME='host-redinternal' dpkg -i ./wazuh-agent.deb
Selecting previously unselected package wazuh-agent.
(Reading database ... 5753 files and directories currently installed.)
Preparing to unpack ./wazuh-agent.deb ...
Unpacking wazuh-agent (4.4.1-1) ...
Setting up wazuh-agent (4.4.1-1) ...
root@host-redinternal_1:/#
```

- Una vez hecho esto procedemos a ejecutar los siguientes códigos que nos dice la herramienta para poder activar el servicio del agente en nuestro host.

Systemd

```
sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
```

- ✚ Pero vemos que al momento de ejecutar el “systemctl start Wazuh-agent” nos da un error el cual es que la variable de entorno la cual es la IP del servidor no funciona o es incorrecta.

```
root@host-redinterna:/# service wazuh-agent start
2024/04/27 08:27:33 wazuh-agentd: ERROR: (4112): Invalid server address found: 'MANAGER_IP'
2024/04/27 08:27:33 wazuh-agentd: CRITICAL: (1215): No client configured. Exiting.
wazuh-agentd: Configuration error. Exiting
```

- ✚ Por lo cual nos iremos a editar el siguiente archivo “var/ossec/etc/ossec.conf” la cual es la ruta donde está la configuración del agente instalado.

```
nano var/ossec/etc/ossec.conf
```

- ✚ Una vez dentro cambiamos solo la sección que dice address por una ip y guardamos ya que estaba “MANAGER_IP”.

```
GNU nano 6.2                               var/ossec/etc/ossec.conf
<!--
Wazuh - Agent - Default configuration for ubuntu 22.04
More info at: https://documentation.wazuh.com
Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>
  <client>
    <server>
      <address>192.168.1.8</address>
      <port>1514</port>
      <protocol>tcp</protocol>
```

- ✚ Una vez hecho esto volvemos arrancar el servicio del agente y vemos que arranca perfectamente sin ningún error.

```
service wazu-agent start
```

```
root@host-redinterna:/# service wazuh-agent start
Starting Wazuh v4.4.1...
Started wazuh-execd...
Started wazuh-agentd...
Started wazuh-syscheckd...
Started wazuh-logcollector...
Started wazuh-modulesd...
Completed.
```

Proyecto 2º ASIR 23/24 Infraguard: Despliegue y protección de infraestructuras con Docker

- Una vez hecho esto nos vamos al dashboard y vemos que ya hay configurado un agente y está activo.

The screenshot shows the Wazuh dashboard with the following sections:

- SECURITY INFORMATION MANAGEMENT:**
 - Security events:** Browse through your security alerts, identifying issues and threats in your environment.
 - Integrity monitoring:** Alerts related to file changes, including permissions, content, ownership and attributes.
- AUDITING AND POLICY MONITORING:**
 - Policy monitoring:** Verify that your systems are configured according to your security policies baseline.
 - System auditing:** Audit users behavior, monitoring command execution and alerting on access to critical files.
- THREAT DETECTION AND RESPONSE:**
 - Vulnerabilities:** Discover what applications in your environment are affected by well-known vulnerabilities.
 - MITRE ATT&CK:** Security events from the knowledge base of adversary tactics and techniques based on real-world observations.
- REGULATORY COMPLIANCE:**
 - PCI DSS:** Global security standard for entities that process, store or transmit payment cardholder data.
 - NIST 800-53:** National Institute of Standards and Technology Special Publication 800-53 (NIST 800-53) sets guidelines for federal information systems.

- Si clicamos en ese agente vemos que ahora podemos monitorizarlo y nos muestra toda la información del equipo desde su IP, el sistema operativo que tiene y su versión, los eventos de este, cuando se registró en el sistema.

The screenshot shows the Wazuh agent details page for the host `host-redinterna`. The top navigation bar includes tabs for Agents, host-redinterna, and host-redinterna. The main content area displays the following information:

- Host Details:**

ID	Status	IP address	Version	Groups	Operating system	Cluster node	Registration date	Last keep alive
001	active	172.17.0.2	Wazuh v4.4.1	default	Ubuntu 22.04 LTS	node01	Apr 27, 2024 @ 10:30:55.000	Apr 27, 2024 @ 10:53:01.000
- Compliance:** A donut chart showing PCI DSS compliance status: 2.2 (184) in green, 10.6.1 (7) in blue, and 10.2.6 (2) in red.
- FIM: Recent events:** A table showing recent events with columns: Time, Path, Action, Rule description, Rule Le..., and Rule Id. It notes "No recent events".
- Events count evolution:** A line chart showing the evolution of event counts over time, ranging from 150 to 200.
- SCA: Lastest scans:** A table showing SCA policy results for Ubuntu Linux 22.04 LTS, including columns: Policy, End scan, Passed, Failed, Not app..., and Score. One entry is shown: `cis_ubuntu22-04`.

9 Monitoreo de Alertas en Wazuh

9.1 Instalación de Un Paquete

- 💡 Vamos a instalar un paquete en el host de la red interna y veremos que nos mostrara una alerta o evento que se ha instalado un paquete.

```
root@hots-redinterna:/# apt install net-tools
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  net-tools
0 upgraded, 1 newly installed, 0 to remove and 18 not upgraded.
Need to get 204 kB of archives.
After this operation, 819 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 net-tools amd64 1.60+git20181103.0eebece-lubuntu5 [204 kB]
Fetched 204 kB in 1s (263 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package net-tools.
(Reading database ... 12436 files and directories currently installed.)
Preparing to unpack .../net-tools 1.60+git20181103.0eebece-lubuntu5_amd64.deb ...
Unpacking net-tools (1.60+git20181103.0eebece-lubuntu5) ...
Setting up net-tools (1.60+git20181103.0eebece-lubuntu5) ...
root@hots-redinterna:/#
```

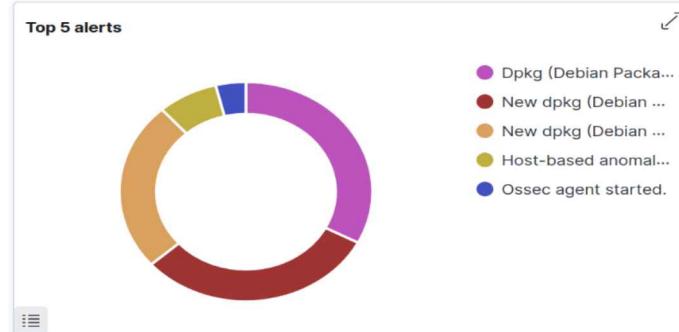
- 💡 Una vez instalado el paquete nos vamos al dashboard de Wazuh y en el agente de “redinterna” apartado “Security Events” y vemos como nos saltan varias alertas de descarga de paquetes.

Agents (5)								
ID	Name	IP address	Group(s)	Operating system	Cluster node	Version	Status	Actions
005	red_DMZ	203.0.113.2	default	Ubuntu 22.04.4 LTS	node01	v4.4.1	●	
004	redinterna	192.168.100.2	default	Ubuntu 22.04.4 LTS	node01	v4.4.1	●	

[Deploy new agent](#) [Export formatted](#) [⚙️](#)

The dashboard displays two stacked area charts showing event counts over time (per 30 minutes). The left chart shows a sharp peak of multiple colored areas (sca, ossec, rootcheck, dpkg, syslog, config_changed) around 12:00. The right chart shows a similar peak for dpkg and syslog. Below the charts are three donut charts: 'Top 5 alerts' (Dpkg, New dpkg, New dpkg, Host-based anomaly, Ossec agent started), 'Top 5 rule groups' (dpkg, syslog, config_changed, ossec, sca), and 'Top 5 PCI DSS Requirements' (10.6.1, 10.2.7, 2.2, 10.2.6).

- Si nos centramos donde dice “Top 5 alerts” vemos que son porque existen nuevos paquetes y ha cambiado la configuración del sistema.



- Si nos vamos al registro de eventos que hay en la parte de abajo nos enseñara más detalladamente la hora, el paquete instalado, versión, los log de instalación.

Security Alerts					
Time ↓	Technique(s)	Tactic(s)	Description	Level	Rule ID
> May 19, 2024 @ 11:57:04.828			Dpkg (Debian Package) half configured.	7	2904
> May 19, 2024 @ 11:57:04.828			New dpkg (Debian Package) installed.	7	2902
> May 19, 2024 @ 11:57:04.828			New dpkg (Debian Package) requested to install.	3	2901
> May 19, 2024 @ 11:48:26.307			Dpkg (Debian Package) half configured.	7	2904
> May 19, 2024 @ 11:48:26.307			New dpkg (Debian Package) installed.	7	2902
> May 19, 2024 @ 11:48:26.307			Dpkg (Debian Package) half configured.	7	2904

- Si lo desplegamos veremos toda la información al respecto.

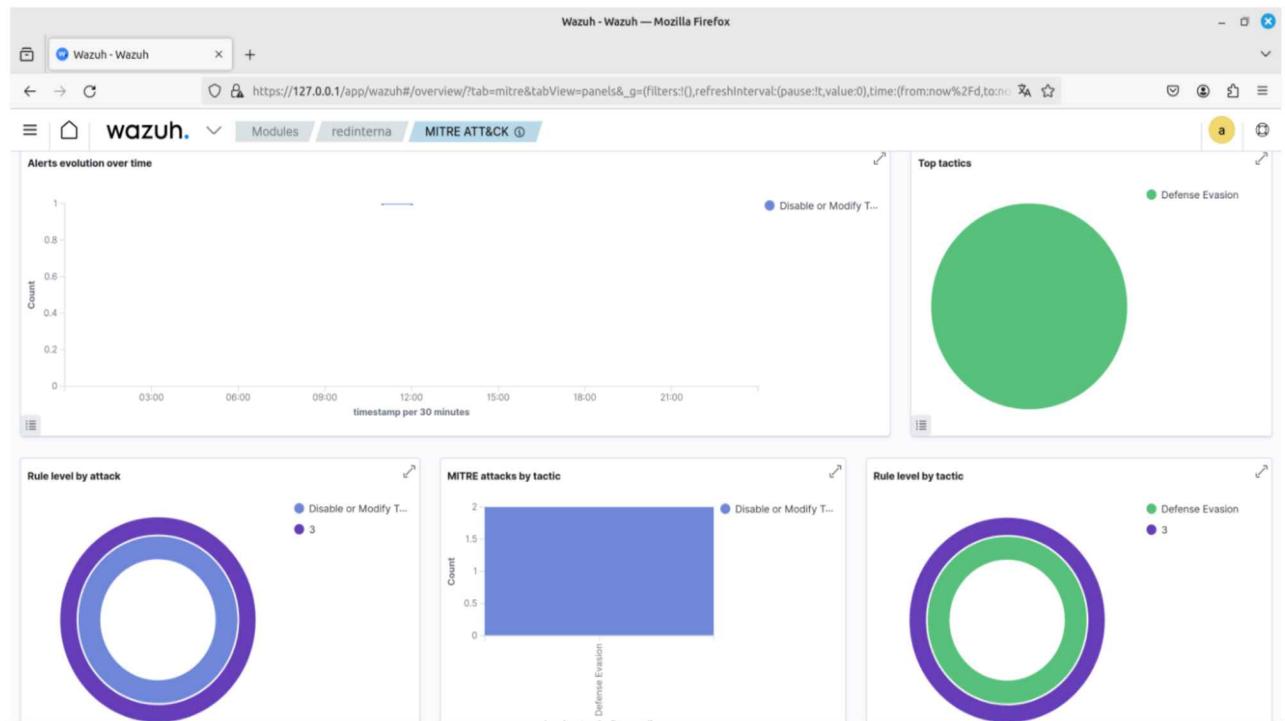
Security Alerts					
Time ↓	Technique(s)	Tactic(s)	Description	Level	Rule ID
May 19, 2024 @ 11:57:04.828			Dpkg (Debian Package) half configured.	7	2904
Table					
Table	JSON	Rule			
			@timestamp	2024-05-19T09:57:04.828Z	
			_id	6KBIKI8Bsxch_9Ejkr1j	
			agent.id	004	
			agent.ip	192.168.100.2	
			agent.name	redinterna	
			data.arch	amd64	
			data.dpkg_status	status half-configured	
			data.package	net-tools	
			data.version	1.60+git20181103.0eebece-1ubuntu5	
			decoder.name	dpkg-decoder	
			full_log	2024-05-19 09:57:03 status half-configured net-tools:amd64 1.60+git20181103.0eebece-1ubuntu5	

9.2 Hacer un Restart al agente de Wazuh

- ✚ Vamos a parar el servicio del agente de Wazuh solo lo que tarde en hacer el restart del mismo y veremos que nos saltaran unas alertas críticas.

```
root@hots-redinterna:/# service wazuh-agent restart
Killing wazuh-modulesd...
Killing wazuh-logcollector...
Killing wazuh-syscheckd...
Killing wazuh-agentd...
Killing wazuh-execd...
Wazuh v4.4.1 Stopped
Starting Wazuh v4.4.1...
Started wazuh-execd...
Started wazuh-agentd...
Started wazuh-syscheckd...
Started wazuh-logcollector...
Started wazuh-modulesd...
Completed.
```

- ✚ Una vez reiniciado el servicio del agente de Wazuh nos vamos al dashboard de Wazuh y en el agente de “redinterna” apartado “MITRE ATT&CK” y vemos como nos saltan varias alertas de ataque del sistema.



- Una vez hecho esto si nos vamos a los eventos y miramos los log vemos que es porque el servicio del agente se ha detenido y lo toma como Evasión de la Defensa.



Entre estas alertas de Wazuh tiene más funcionalidades como lo son:

- Detección de accesos fallidos.
- Accesos satisfactorios.
- Alertas de archivos corrompidos.
- Alertas de ejecución de comandos por usuarios.

10 Documentación Github

- Enlace:https://github.com/ManuEduGVA/2324_ASIR_EMANUEL_ROJAS_COLLAZOS_INFRAGUARD
- Contenido del proyecto de Github:
 - Memoria.
 - Scripts de iptables.
 - Despliegue contenedores Docker.
 - Despliegue de Wazuh en Docker.

11 Recursos Utilizados

11.1 Hardware

- Procesador Ryzen 5 3550H
- Memoria RAM 24 GB
- Disco duro SSD M2 256 GB

11.2 Software

- MiniTool Partition Wizard
- Wazuh
- Docker
- Docker-compose

- Kibana
- Iptables

11.3 Sistemas Operativos

- Windows 11 pro
- Linux Mint 21.3 Cinnamon

12 Implementación en una empresa

Como emprendedor, mi objetivo es crear una herramienta para pequeñas y medianas empresas (pymes) que se integrará en su infraestructura interna. Inicialmente, mi equipo y yo la instalaremos y configuraremos, y luego formaremos al personal de IT para su uso continuo.

Esta herramienta supervisa y soluciona errores de los equipos de la empresa, además de gestionar actualizaciones e instalaciones de software. Su ventaja principal es que no requiere instalación en los equipos de usuario, ya que opera mediante Docker, ahorrando espacio en los discos duros.



Nuestro servicio es económicamente ventajoso al ser externalizado, lo que ahorra gastos en contratación de personal. Mi equipo y yo estamos registrados como autónomos y recibimos formación en prevención de riesgos laborales.

Para promocionar el servicio y diferenciarlo de la competencia, he contratado una empresa de marketing que se encargará de estudiar el mercado y aplicar estrategias para aumentar la demanda.

En cuanto a la gestión financiera y fiscal, he contratado un asesor fiscal para asegurar el cumplimiento normativo y optimizar los recursos. Mi equipo está compuesto por dos técnicos y yo, todos cotizando como autónomos, con responsabilidades claramente definidas: yo desarrollo y actualizo la herramienta, mientras que los técnicos brindan soporte y formación al personal de IT de las empresas clientes.

13 Conclusión

Este proyecto fue un reto a nivel personal ya que he usado herramientas que desconocía totalmente y para implementar la herramienta de monitorización Wazuh use Docker porque me parecía original e inusual y así no empleaba máquinas virtuales, que es un poco más sencillo.

Inicialmente, consideré otras herramientas de monitorización, pero me decidí por Wazuh debido a su amplia gama de funcionalidades. A lo largo del proceso, hubo complicaciones como la falta de documentación y problemas de instalación, pero mediante la experimentación y la búsqueda de soluciones alternativas se encontró solución.

La implementación de un sistema SIEM como Wazuh en un entorno perimetral desplegado con Dokcer-Compose presenta una serie de desafíos y etapas clave en su configuración. A través de un proceso que abarca desde la creación de particiones en el disco duro hasta la configuración detallada de reglas de iptables, se logra establecer un entorno robusto para la monitorización de la seguridad del sistema.

En conclusión, el proyecto implicó una planificación detallada, investigación exhaustiva y resolución creativa de problemas técnicos. A través de este proceso, adquirí habilidades valiosas en el despliegue y configuración de sistemas de seguridad.

13.1 Problemas encontrados

Los problemas más contundentes que he tenido ha sido el poder ejecutar la herramienta de Wazuh dentro de un contenedor de Docker. Y al ver que me generaba muchas complicaciones decidí leer documentaciones oficiales y buscar información sobre cómo hacerlo funcionar y la solución que encontré fue desplegar Wazuh con Docker-compose.

Otro de los problemas fue que al momento de instalar el agente de Wazuh para poder monitorizar el equipo de Debian me daba un error de repositorios y dependencias así que decidí cambiar el sistema operativo a Ubuntu y al ejecutar el deploy me instalo sin problemas.

Pero al momento de iniciar el servicio del agente de Wazuh me daba un error de IP, en la cual no reconocía la IP del servidor de Wazuh y era porque había tomado el nombre de la variable de entorno y no su valor que era la IP.

13.2 Mejoras

Mas que mejora sería una ampliación la integración de Wazuh y Elastic Stack enriquece el enfoque de monitoreo de seguridad al combinar las capacidades avanzadas de recolección y análisis de datos de Wazuh con las potentes herramientas de visualización y análisis en tiempo real de Elastic Stack, proporcionando una solución robusta y escalable para la gestión de la seguridad.

Cabe recalcar que intente implementarlo haciendo el despliegue con Docker-Compose de la herramienta Elastic Stack.

- ✚ Aquí vemos como me hace bien el despliegue con Docker-compose

```
emanuel@proyecto:~/elk-stack$ docker compose up -d
WARN[0000] /home/emanuel/elk-stack/docker-compose.yaml: `version` is obsolete
[+] Running 14/14
✓ logstash 13 layers [██████████]
  ✓ fa1c85952657 Already exists
  ✓ d3cc84fa6bf0 Pull complete
  ✓ 6ee6da1d45b6 Pull complete
  ✓ 4d48ff848293 Pull complete
  ✓ 4ca545ee6d5d Pull complete
  ✓ dac7aa3905e1 Pull complete
  ✓ c0b792304bf3 Pull complete
  ✓ 908a10f7fb59 Pull complete
  ✓ 9d1abc927e3e Pull complete
  ✓ db1d17e224a0 Pull complete
  ✓ 46a418bc521f Pull complete
  ✓ f3e3ed749918 Pull complete
  ✓ 40bcbbc14d38f Pull complete
[+] Running 4/5
{: Network elk-stack_default Created
✓ Container kibana Started
✓ Container logstash Started
✓ Container setup Started
✓ Container elasticsearch Started
emanuel@proyecto:~/elk-stack$
```

- ✚ Pero al momento de acceder a la página para colocar las credenciales me daba el siguiente error.



- ✚ Y si me iba a los logs me aparecía que no había sido posible autenticar el usuario "Kibana_system".

```
Root causes:
  security_exception: unable to authenticate user [kibana_system] for REST request
```

14 Bibliografía/Webgrafía

14.1 Documentación

- Arranque Dual: <https://www.xataka.com/basics/como-instalar-gnu-linux-a-windows-11-ordenador>
- Instalar Docker: <https://docs.docker.com/engine/install/debian/>
- Información de Wazuh: <https://es.linkedin.com/pulse/wazuh-como-herramienta-siem-esencial-para-la-de-tu-maurice>
- Despliegue de Wazuh agent: <https://documentation.wazuh.com/4.4/installation-guide/wazuh-agent/wazuh-agent-package-linux.html>
- Despliegue de Wazuh: <https://es.linux-console.net/?p=20658>
- Implementar WAZUH + ELK: <https://wazuh.com/blog/detection-with-elastic-stack-integration/>

14.2 Videos

- Arreglar el grub: <https://www.youtube.com/watch?v=JU1kw1k-DGE>
- Instalación de Docker: https://www.youtube.com/watch?v=tb1wAOssVds&ab_channel=TechconAgust
- Despliegue de Wazuh: https://www.youtube.com/watch?v=mOFiKvmZ2F4&ab_channel=Jos%C3%A9MariaLabarta