



CICLO FORMATIVO DE GRADO SUPERIOR
ADMINISTRACIÓN DE SISTEMAS INFORMÁTICOS
EN RED

Proyecto de Fin de Ciclo

**Implantación de Centro de Operaciones de Seguridad
Automatizado**

David Julián Lagunar

Tutor: Ruth Esther Gómez

Curso: CURSO 2025/2026

RESUMEN:

Este proyecto presenta el diseño y despliegue de un Centro de Operaciones de Seguridad (SOC) automatizado utilizando la plataforma Wazuh.

El objetivo principal ha sido monitorizar la integridad de endpoints multiplataforma (Windows y Linux), integrar la inteligencia de amenazas de VirusTotal y crear políticas de contención autónomas. Como resultado, las pruebas de intrusión demostraron que el sistema es capaz de detectar, catalogar y eliminar el malware mediante Respuesta Activa en un muy pocos segundos.

En conclusión, la automatización reduce drásticamente el Tiempo Medio de Respuesta (MTTR), sustituyendo un modelo más reactivo a los acontecimientos por uno proactivo y automatizado. Reduciendo la carga de trabajo de los analistas de ciberseguridad y aumentando la eficacia en su trabajo.

ABSTRACT:

This project presents the design and deployment of an automated Security Operations Center (SOC) using the Wazuh platform. The primary objective was to monitor the integrity of cross-platform endpoints (Windows and Linux), integrate threat intelligence from VirusTotal, and create autonomous containment policies. As a result, penetration tests demonstrated that the system is capable of detecting, cataloging, and removing malware through Active Response in just a few seconds.

In conclusion, automation drastically reduces the Mean Time to Response (MTTR), replacing a more reactive model with a proactive and automated one. This reduces the workload of cybersecurity analysts and increases the efficiency of their work.

AGRADECIMIENTOS:

Quería dedicar este apartado a agradecer a las personas que me han acompañado durante esos dos cursos, que han creído en mi capacidad y posibilidades de reformularme profesionalmente. Que han sido sostén, apoyo y motivación en los momentos de flaqueza y sin los que no podría haber llegado hasta el final de esta carrera.

En especial a mi familia, a mi pareja, mis amigos y compañeros que he conocido en esta travesía, mención especial a mi compañero y amigo Abel Ramírez Navacerrada y al profesor Javier Fernández de Cara.

Índice

RESUMEN:	1
ABSTRACT:	1
AGRADECIMIENTOS:	1
1. Introducción	2
1.1 Objetivos	2
1.2 Contextualización	2
1.3 Justificación	3
2. Gestión del proyecto	1
2.1 Metodología de trabajo	1
2.2 Planificación	3
3. Herramientas y tecnologías utilizadas	4
4. Desarrollo del proyecto	5
Fase 1: Despliegue de Infraestructura y Conectividad	5
4.1.1 Arquitectura de Red y Acceso al SIEM	5
4.1.2 Despliegue del Agente en Windows 10	9
4.1.3 Despliegue del Agente en Linux (Ubuntu)	11
4.1.4 Verificación de Conectividad	13
Fase 2: Detección de Malware (Integración con VirusTotal)	14
4.2.1 Configuración del Servidor (Wazuh Manager en Ubuntu)	14
4.2.2 Preparación del Endpoint (Windows 10)	15
4.2.3 Simulación de Ataque y Detección Crítica	17
6. Pruebas y resultado	21
5.1 Configuración del Servidor (Wazuh Manager)	21
5.2 Despliegue del Script en el Endpoint (Windows 10)	22
5.3 Prueba de Concepto y Verificación	23
7. Conclusiones y trabajos futuro	25
6.1 Conclusiones	25
6.2 Trabajos futuros	25
BIBLIOGRAFÍA	26

1. Introducción

En este documento presentamos el diseño e implementación de un Centro de Operaciones de Seguridad (SOC) automatizado, cuyo objetivo es detectar y neutralizar amenazas cibernéticas en tiempo real. El proyecto aborda la necesidad de aportar nuevas herramientas a los analistas de ciberseguridad y de reducir los tiempos de respuesta ante incidentes en entornos corporativos.

Para lograrlo, se ha desplegado la plataforma SIEM/EDR Wazuh en un entorno virtualizado, monitorizando la integridad de archivos (FIM) tanto en servidores Linux como en estaciones de trabajo Windows. Mediante la integración de la API de VirusTotal, el sistema adquiere la capacidad de evaluar de forma autónoma la reputación de cualquier archivo sospechoso. La principal aportación de este proyecto es la configuración de políticas de Respuesta Activa (*Active Response*), las cuales permiten aislar y eliminar el malware en menos de dos segundos de forma automatizada.

A lo largo de esta memoria, describiremos detalladamente la topología de red empleada, las fases de configuración de las herramientas tecnológicas y los resultados obtenidos en las pruebas de intrusión controladas.

1.1 Objetivos

Los objetivos principales que se pretenden lograr con la implementación de este Centro de Operaciones de Seguridad (SOC) son los siguientes:

- **Desplegar** una infraestructura de red virtualizada que centralice la gestión de eventos de seguridad, comunicando un servidor maestro Wazuh Manager, con agentes instalados en sistemas operativos Windows y Linux.
- **Monitorizar** en tiempo real la integridad del sistema de archivos (FIM) en los *endpoints*, permitiendo detectar de forma instantánea la creación o alteración de posibles vectores de ataque.
- **Integrar** fuentes externas de Inteligencia de Amenazas (CTI) mediante la API de VirusTotal, para evaluar y confirmar la severidad de los eventos sospechosos con múltiples motores antivirus simultáneamente.
- **Automatizar** la mitigación de incidentes de seguridad mediante la configuración de políticas de Respuesta Activa (*Active Response*), aislando o eliminando la amenaza sin necesidad de intervención manual.
- **Validar** la eficacia del sistema y la reducción del Tiempo Medio de Respuesta (MTTR) mediante la simulación controlada de inyección de malware (archivos de prueba EICAR).

1.2 Contextualización

Nuestro proyecto se enmarca en el sector de la ciberseguridad corporativa, específicamente en las operaciones de defensa, monitorización y respuesta ante incidentes el llamado *Blue Team*. En la actualidad, las infraestructuras tecnológicas empresariales se enfrentan a un aumento exponencial de ciberataques automatizados. Amenazas como el *ransomware*, los troyanos o los ataques de día cero (*zero-day*) operan a velocidades que superan con creces la capacidad de reacción manual de los equipos de seguridad.

Sin embargo, muchas de las soluciones de protección tradicionales, como los antivirus locales, operan de forma aislada. Estas herramientas suelen basarse en la detección estática mediante firmas y carecen de una visión centralizada del estado de la red. Esta carencia provoca que los analistas de seguridad sufran lo que en la industria se conoce como "fatiga de alertas" (*alert fatigue*), retrasando significativamente el triaje, la identificación y la contención del software malicioso.

Por lo tanto, nuestra infraestructura SOC tiene como objetivo proporcionar una solución centralizada y dinámica que supere estas limitaciones, integrando capacidades de detección y respuesta en los *endpoints* (EDR). Además, el sistema se ha diseñado con un enfoque estricto en la automatización y la inteligencia externa. Esto permite la ejecución proactiva de medidas de remediación de forma autónoma, protegiendo los activos críticos de manera inmediata y eficiente ante cualquier brecha de seguridad.

1.3 Justificación

La justificación principal de este proyecto reside en la eficiencia operativa y en la optimización de los recursos humanos dentro de un Centro de Operaciones de Seguridad (SOC). En la actualidad, los analistas de ciberseguridad (especialmente los de nivel 1) invierten una gran parte de su jornada revisando alertas repetitivas, falsos positivos o amenazas conocidas y documentadas.

Este volumen de trabajo rutinario consume un tiempo y un talento muy valioso que deberían estar enfocados en tareas mucho más críticas. Por tanto, el desarrollo de esta infraestructura busca actuar como un "filtro automatizado". Al delegar en Wazuh y VirusTotal la responsabilidad de detectar, confirmar y eliminar automáticamente el *malware* básico en cuestión de segundos, conseguimos que el sistema asuma la carga operativa más pesada.

El objetivo final es liberar de tiempo al analista de ciberseguridad. Al reducir el ruido y las tareas repetitivas, el equipo humano puede redirigir sus esfuerzos hacia áreas donde su criterio e intuición son irremplazables, como la caza proactiva de amenazas (*Threat Hunting*), el análisis forense avanzado, el estudio de nuevas vulnerabilidades o el diseño de mejores políticas de seguridad. En definitiva, este proyecto demuestra cómo la automatización no sustituye al analista, sino que lo potencia.

2. Gestión del proyecto

La correcta ejecución de este proyecto de ciberseguridad ha requerido mucha organización, una gran cantidad de ensayos y una búsqueda de información para la resolución de errores.

Para mejorar la eficiencia, tanto en la asimilación de los conceptos como en la realización técnica del proyecto, hemos dividido el despliegue del SOC en fases estructuradas.

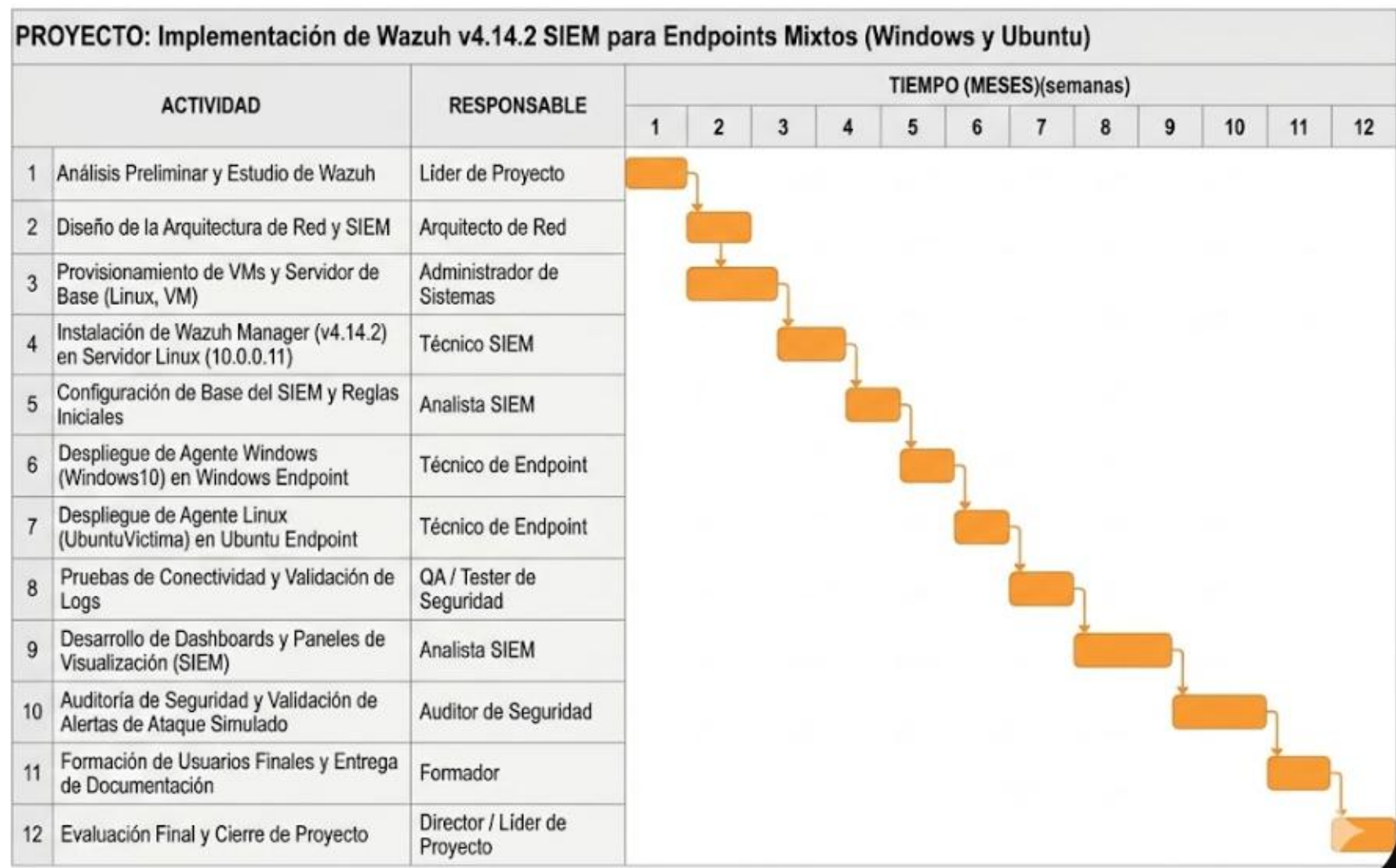
2.1 Metodología de trabajo

Para la ejecución de este proyecto, hemos adoptado una metodología dividida en fases. Este enfoque nos ha permitido afrontar la dificultad del proyecto de forma progresiva, garantizando la estabilidad de cada parte antes de avanzar al siguiente nivel. Las fases desarrolladas han sido las siguientes:

1. **Fase de Investigación y Diseño Teórico:** En esta etapa inicial, realizamos un estudio de las plataformas SIEM/EDR como Splunk o la utilizada Wazuh, analizando la documentación y las especificaciones de la API de VirusTotal. Se diseñó la arquitectura de red con un modelo de Red NAT en entorno aislado para asegurar que las pruebas se realizarían en un espacio seguro.
2. **Fase de Despliegue de Infraestructura:** Consistió en la creación y configuración de las máquinas virtuales. Se instaló el servidor central, Wazuh Manager, sobre Ubuntu y, después, se procedió al despliegue y registro de los agentes en los endpoints objetivo: Windows 10 y Ubuntu.
3. **Fase de Configuración Analítica (Detección):** En esta fase, ajustamos las políticas de seguridad. Habilitamos y configuramos el módulo *File Integrity Monitoring* (FIM) en el agente Windows para supervisar directorios críticos en tiempo real. A su vez se integró la API de VirusTotal en el Manager para dotar al sistema de inteligencia externa.
4. **Fase de Pruebas y Simulación (Ensayo y Error):** Se procedió a la inyección controlada de *malware* simulado (archivos con firma EICAR) en el *endpoint*. Esta fase requirió la resolución de diversas incidencias técnicas, como ajustes de permisos y evasión temporal de antivirus locales hasta lograr la correcta generación de alertas en el panel de control.
5. **Fase de Automatización y Remediación:** es la parte innovadora del proyecto. Se desarrollaron y vincularon *scripts* de respuesta automatizada para que el Manager pudiera ejecutar comandos de neutralización directamente en el *endpoint* comprometido, eliminando la amenaza sin intervención humana.

6. **Fase de Documentación:** Recopilación de evidencias forenses, trazas de logs y capturas de pantalla del Dashboard para redactar la presente memoria técnica.

2.2 Planificación



3. Herramientas y tecnologías utilizadas

Para desarrollar y montar este laboratorio de seguridad, hemos utilizado una serie de herramientas y tecnologías enfocadas en la virtualización y la automatización:

- **VirtualBox:** Lo hemos utilizado como hipervisor para montar toda la infraestructura. Nos ha permitido crear una "Red NAT" aislada para que las máquinas virtuales (servidor y víctimas) se comuniquen entre sí de forma segura y sin afectar a nuestro equipo físico.
- **Wazuh (Manager y Agent):** Es el motor principal del proyecto. Hemos utilizado Wazuh como sistema SIEM/EDR. El servidor central (Manager) se ha instalado sobre Linux para analizar los datos, y los sensores (Agents) los hemos desplegado en los equipos para recoger los *logs* y vigilar la integridad de los archivos.
- **Sistemas Operativos (Windows 10 y Ubuntu):** Se han utilizado como los *endpoints* o máquinas víctima de la red. Windows 10 ha sido el entorno principal donde hemos realizado las inyecciones de *malware* para comprobar si el sistema se defendía correctamente.
- **API de VirusTotal:** Para dotar de inteligencia al sistema, hemos conectado Wazuh con la API de VirusTotal. De esta forma, cuando el agente detecta un archivo nuevo, el sistema consulta automáticamente su base de datos para confirmar con decenas de motores si se trata de una amenaza real.
- **PowerShell y Bash:** Hemos utilizado la línea de comandos para la configuración y la lógica del proyecto. Por ejemplo, hemos usado PowerShell en el entorno Windows para realizar la instalación desatendida del agente, desactivar temporalmente la protección de Windows Defender, inyectar el *malware* de prueba (EICAR) y ejecutar el *script* encargado de eliminar la amenaza de forma automática.

4. Desarrollo del proyecto

El desarrollo técnico de este proyecto lo hemos estructurado de forma modular. Hemos comenzado por establecer la infraestructura base y asegurar la comunicación, para después ir escalando hacia la configuración de los motores de detección avanzada. A continuación, detallamos los procedimientos técnicos llevados a cabo en cada fase.

Fase 1: Despliegue de Infraestructura y Conectividad

El objetivo de esta fase inicial es establecer el entorno de red y asegurar la comunicación bidireccional entre el servidor central (Wazuh Manager) y los *endpoints* que vamos a monitorizar.

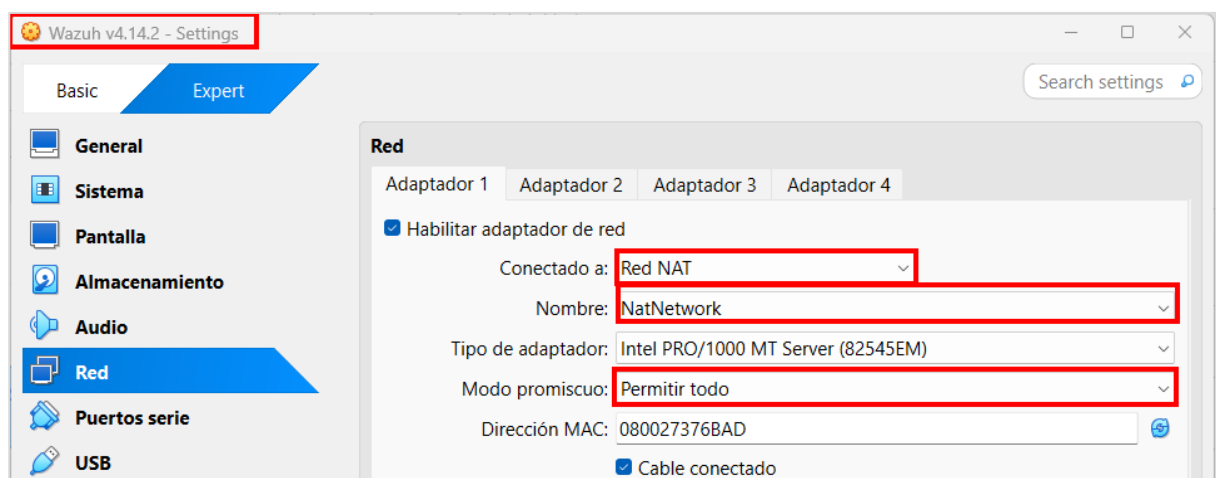
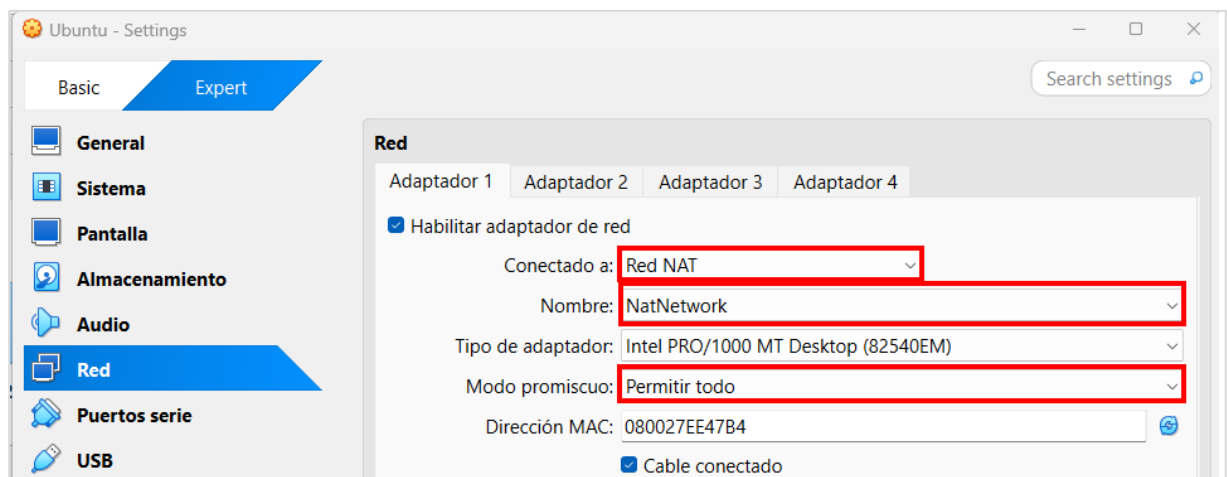
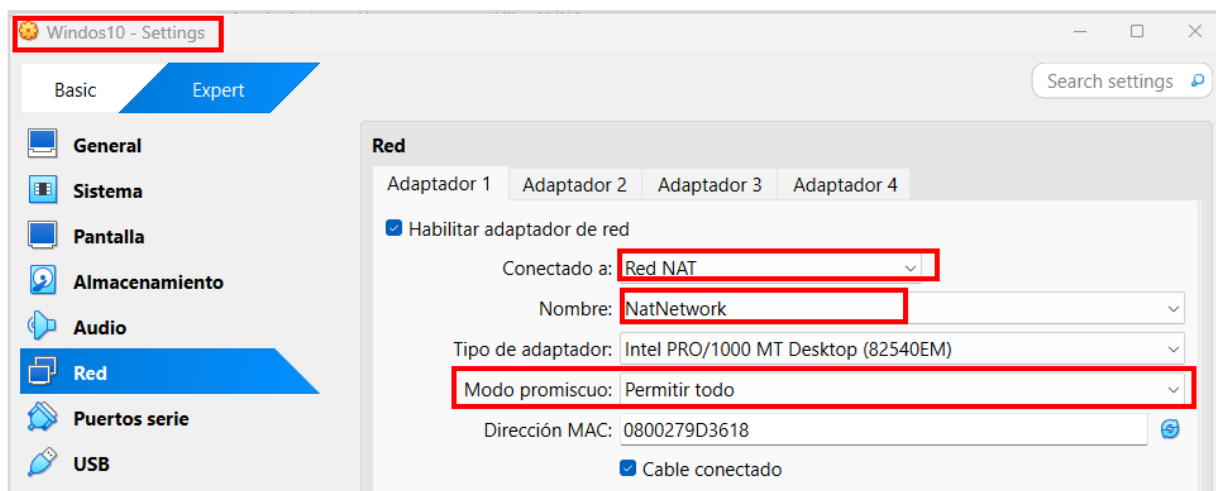
4.1.1 ARQUITECTURA DE RED Y ACCESO AL SIEM

Para garantizar el aislamiento de las pruebas y la correcta comunicación interna entre las máquinas virtuales, hemos configurado una red dedicada de tipo "Red NAT" en VirtualBox con el direccionamiento 10.0.2.0/24 y el servicio DHCP habilitado.

Una vez configurada la red, procedimos a verificar la conectividad en las tres máquinas de la infraestructura (Windows 10, Ubuntu y Wazuh Server).

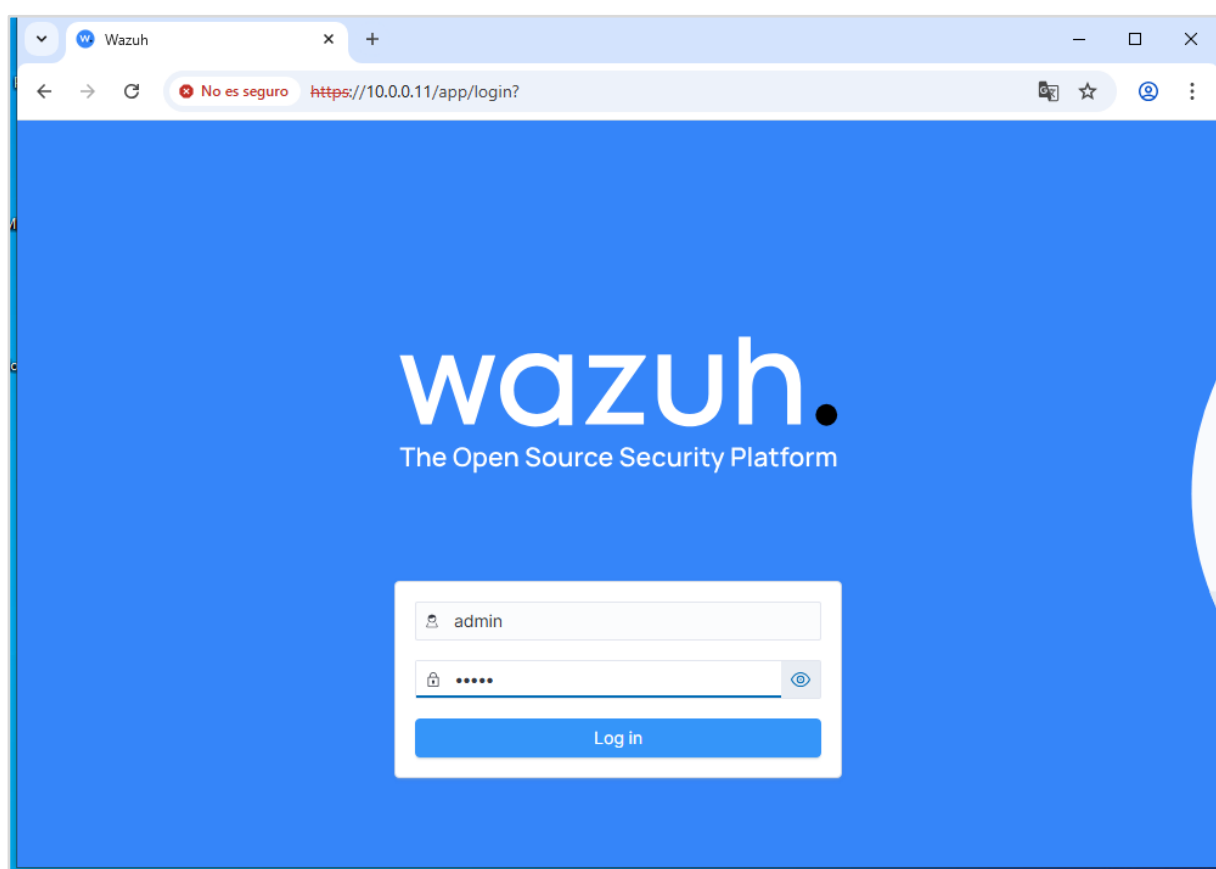
Redes solo-anfitrión Redes NAT Redes en la nube			
Nombre	Prefijo IPv4	Prefijo IPv6	Servidor DHCP
NatNetwork	10.0.0.0/24	fd17:625c:f037::/64	Habilitado
NatNetwork1	20.0.0.0/24	fd17:625c:f037::/64	Habilitado
NatNetwork2	30.0.0.0/24	fd17:625c:f037::/64	Habilitado

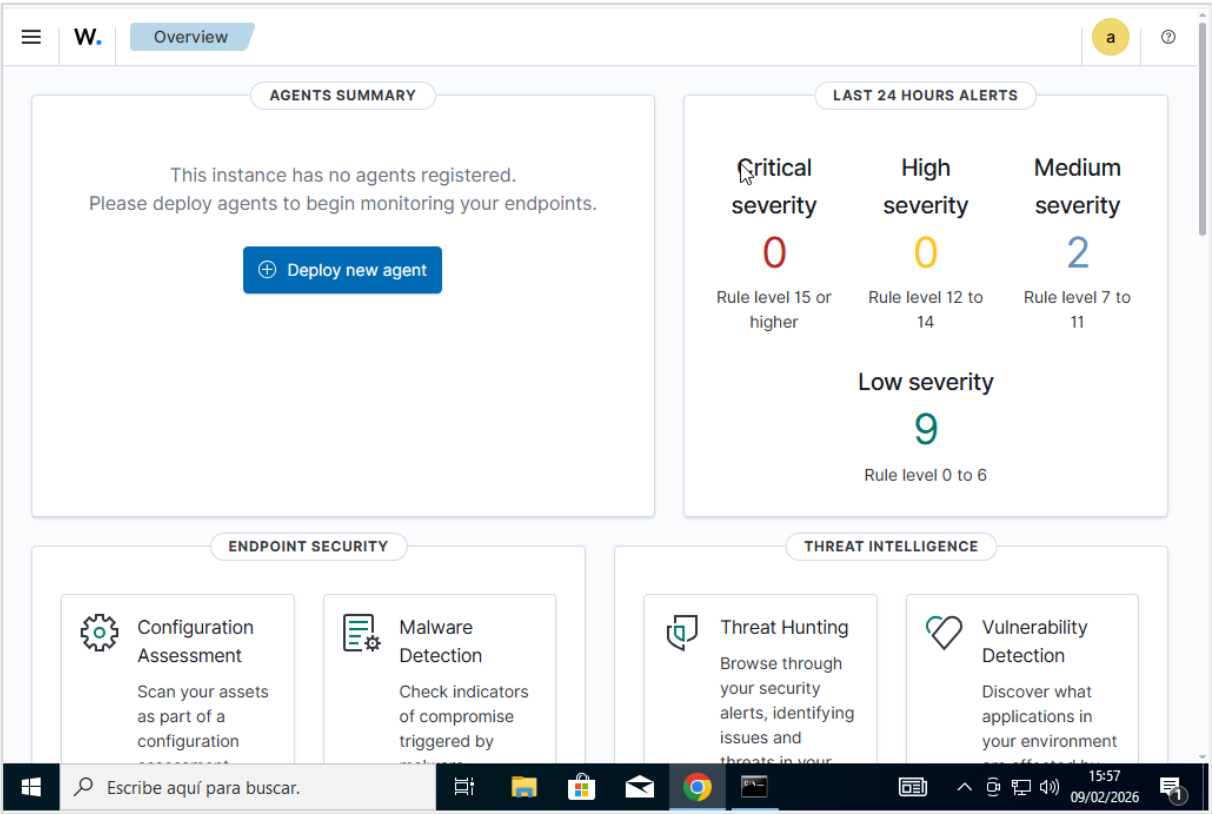
Opciones generales	Reenvío de puertos
Nombre: NatNetwork	
Prefijo IPv4: 10.0.0.0/24	
☒ Habilitar DHCP	



Para la identificación del servidor, tras iniciar la máquina virtual de Wazuh e ingresar con las credenciales del sistema operativo (wazuh-user / wazuh), utilizamos el comando `ip a`. Esto nos permitió confirmar que la interfaz de red del servidor había recibido la IP 10.0.0.11 por parte del DHCP.

Con esta confirmación, nos dirigimos a la máquina Windows 10. Desde el navegador Chrome, accedimos a la interfaz web del SIEM mediante la URL <https://10.0.0.11>. Alcanzamos la pantalla de *login* de Wazuh y nos autenticamos con las credenciales administrativas por defecto (admin / admin). Tras este paso, obtuvimos acceso completo al *Dashboard* de gestión.

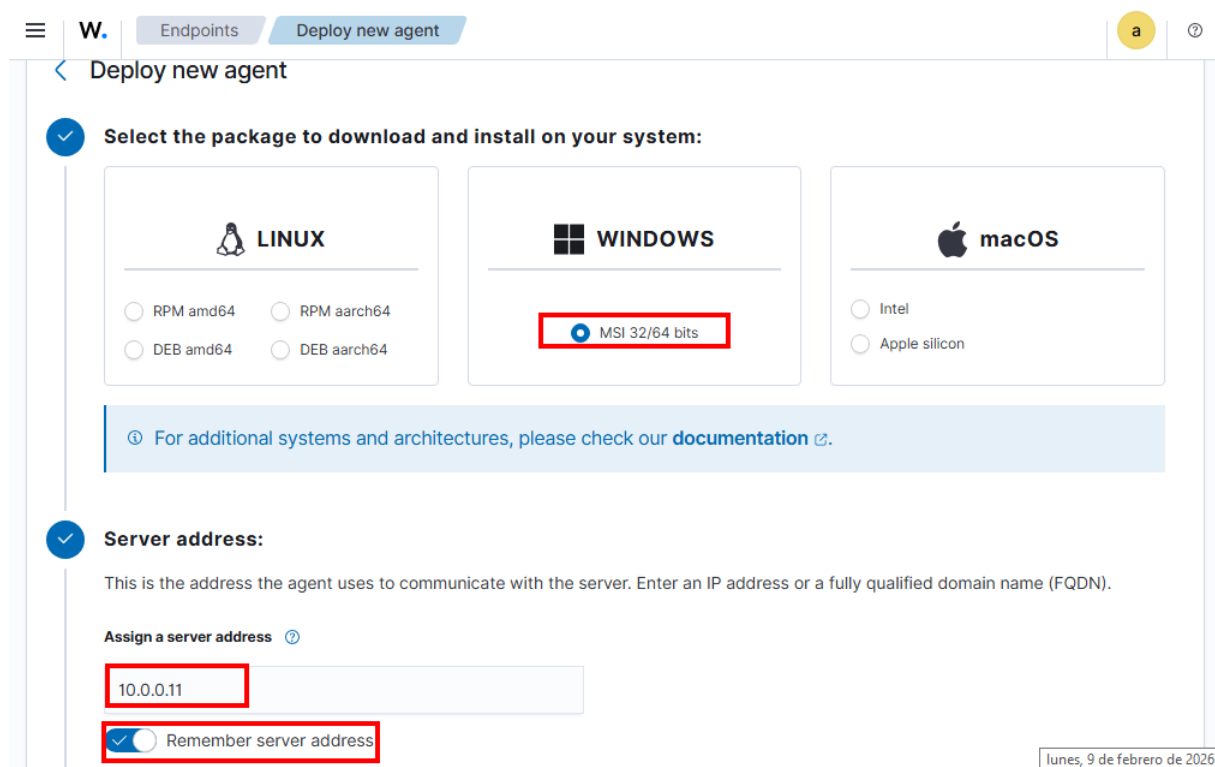




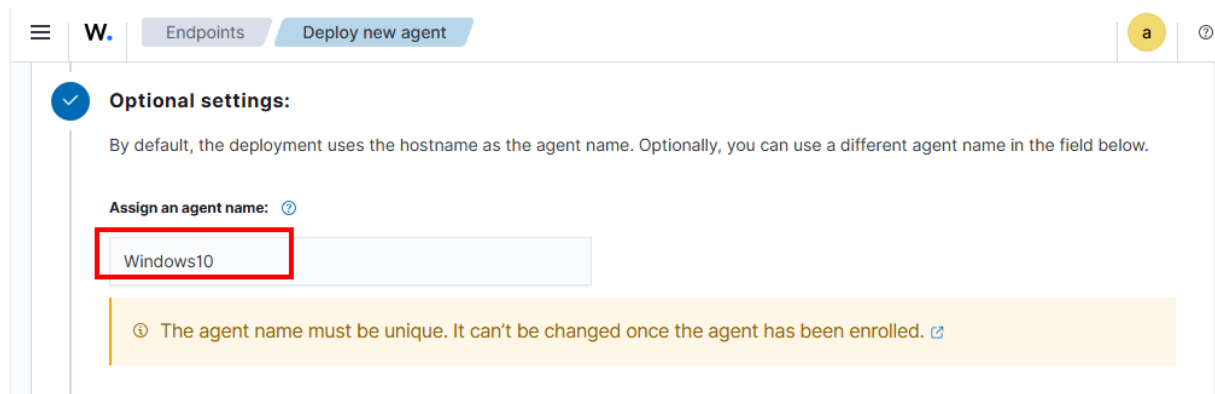
4.1.2 DESPLIEGUE DEL AGENTE EN WINDOWS 10

Para conectar nuestro *endpoint* de Windows a la monitorización, utilizamos el asistente del panel *Deploy new agent* con el fin de generar un instalador preconfigurado. Los parámetros definidos fueron los siguientes:

- **Operating System:** Windows.
- **Server address:** 10.0.0.11 (IP de nuestro Manager).
- **Agent name:** Windows10 (Identificador único de la máquina).



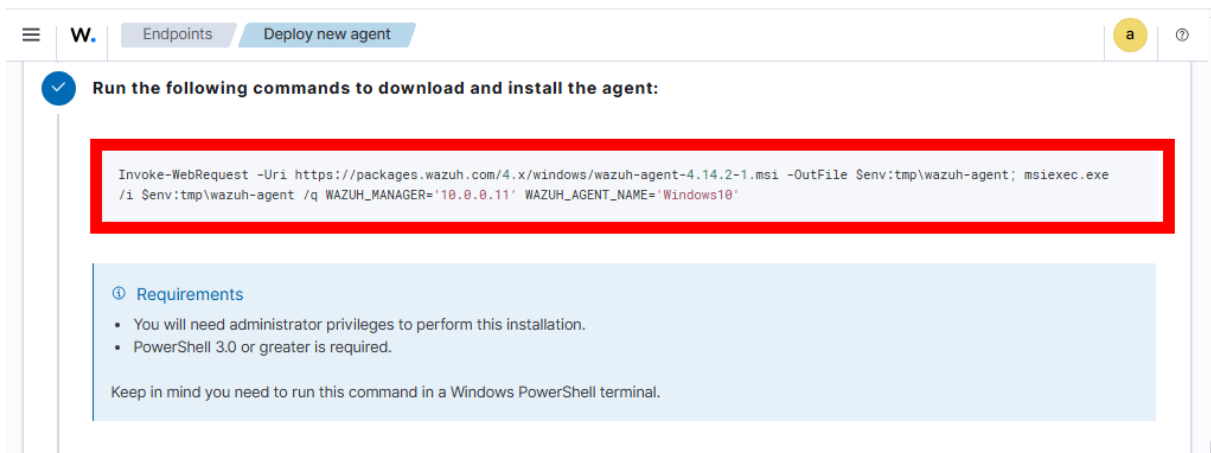
The screenshot shows the 'Deploy new agent' interface. The 'Operating System' section has three tabs: LINUX, WINDOWS, and macOS. The 'WINDOWS' tab is selected, and the 'MSI 32/64 bits' option is chosen. The 'Server address' section shows the IP address '10.0.0.11' entered in the text field, and the 'Remember server address' checkbox is checked. A date stamp 'lunes, 9 de febrero de 2026' is visible in the bottom right corner.



The screenshot shows the 'Optional settings' section of the 'Deploy new agent' interface. The 'Assign an agent name' text field contains the value 'Windows10'. A warning message at the bottom states: 'The agent name must be unique. It can't be changed once the agent has been enrolled.'

Esta configuración generó automáticamente un bloque de código para la instalación. Para ejecutarlo, abrimos una consola de PowerShell como administrador en la máquina Windows 10 y lanzamos el comando para descargar e instalar el agente de forma desatendida:

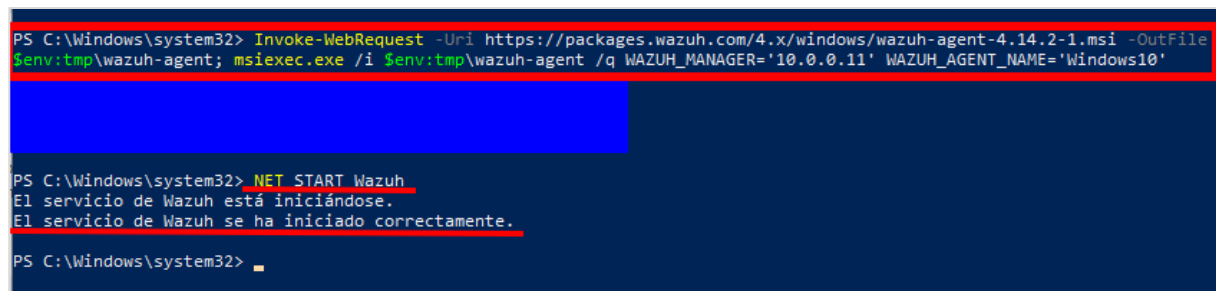
```
Invoke-WebRequest -Uri  
https://packages.wazuh.com/4.x/windows/wazuh-agent-  
4.14.2.1.msi -Outfile $env:tmp\wazuh-agent; msixec.exe /i  
$env:tmp\wazuh-agent /q WAZUH_MANAGER='10.0.0.11'  
WAZUH_AGENT_NAME='Windows10'
```



The screenshot shows the Wazuh dashboard interface. At the top, there's a navigation bar with a menu icon, the Wazuh logo, and tabs for 'Endpoints' and 'Deploy new agent'. Below the navigation bar, a blue checkmark icon is followed by the text 'Run the following commands to download and install the agent:'. A red rectangular box highlights the PowerShell command: `Invoke-WebRequest -Uri https://packages.wazuh.com/4.x/windows/wazuh-agent-4.14.2-1.msi -OutFile $env:tmp\wazuh-agent; msixec.exe /i $env:tmp\wazuh-agent /q WAZUH_MANAGER='10.0.0.11' WAZUH_AGENT_NAME='Windows10'`. Below this, there's a section titled 'Requirements' with a list of two items: 'You will need administrator privileges to perform this installation.' and 'PowerShell 3.0 or greater is required.' At the bottom of this section, it says 'Keep in mind you need to run this command in a Windows PowerShell terminal.'

Tras finalizar la instalación, procedimos a la activación del servicio mediante el comando `NET START Wazuh`, logrando que el agente comenzara a reportar eventos al servidor.

```
NET START Wazuh
```



The screenshot shows a Windows PowerShell terminal window. The first command entered is `Invoke-WebRequest -Uri https://packages.wazuh.com/4.x/windows/wazuh-agent-4.14.2-1.msi -OutFile $env:tmp\wazuh-agent; msixec.exe /i $env:tmp\wazuh-agent /q WAZUH_MANAGER='10.0.0.11' WAZUH_AGENT_NAME='Windows10'`. The second command is `NET START Wazuh`. The output shows 'El servicio de Wazuh está iniciándose.' followed by 'El servicio de Wazuh se ha iniciado correctamente.'

4.1.3 DESPLIEGUE DEL AGENTE EN LINUX (UBUNTU)

Repetimos el proceso de despliegue para integrar nuestra máquina Linux en el ecosistema. Volvimos al asistente *Deploy new agent* y cumplimentamos los datos correspondientes:

- **Operating System:** Linux.
- **OS:** Debian/Ubuntu.
- **Architecture:** amd64.
- **Server address:** 10.0.0.11.
- **Agent name:** UbuntuVictima.

The screenshot shows the 'Deploy new agent' wizard in a web interface. The top navigation bar includes a menu icon, a 'W.' logo, and tabs for 'Endpoints' and 'Deploy new agent'. The main heading is 'Deploy new agent'. A blue checkmark icon indicates the current step: 'Select the package to download and install on your system:'. Below this, there are three main categories: LINUX, WINDOWS, and macOS. Under LINUX, there are four radio button options: 'RPM amd64', 'RPM aarch64', 'DEB amd64' (which is selected and highlighted with a red box), and 'DEB aarch64'. Under WINDOWS, there is one option: 'MSI 32/64 bits'. Under macOS, there are two options: 'Intel' and 'Apple silicon'. A light blue banner below the LINUX section contains a link to 'documentation'. The next step, 'Server address:', is partially visible. It includes a text input field with '10.0.0.11' (highlighted with a red box) and a 'Remember server address' checkbox that is checked.

The screenshot shows the 'Optional settings' step of the 'Deploy new agent' wizard. A blue checkmark icon indicates the current step. The text says: 'By default, the deployment uses the hostname as the agent name. Optionally, you can use a different agent name in the field below.' Below this, there is a section 'Assign an agent name:' with a text input field containing 'UbuntuVictima' (highlighted with a red box). A yellow banner below the input field contains a warning: 'The agent name must be unique. It can't be changed once the agent has been enrolled.' At the bottom, there is a section 'Select one or more existing groups:' with a dropdown menu showing 'Default'.

Copiamos el *script* generado y lo ejecutamos en la terminal de nuestra máquina Ubuntu. Esta instrucción en una sola línea se encargó de descargar el paquete .deb mediante wget e instalarlo configurando el Manager automáticamente:

```
wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent_4.14.2-1_amd64.deb && sudo WAZUH_MANAGER='10.0.0.11' WAZUH_AGENT_NAME='UbuntuVictima' dpkg -i ./wazuh-agent_4.14.2-1_amd64.deb
```

4 Run the following commands to download and install the agent:

```
wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.14.2-1_amd64.deb && sudo WAZUH_MANAGER='10.0.0.11' WAZUH_AGENT_NAME='UbuntuVictima' dpkg -i ./wazuh-agent_4.14.2-1_amd64.deb
```

④ Requirements

- You will need administrator privileges to perform this installation.
- Shell Bash is required.

Keep in mind you need to run this command in a Shell Bash terminal.

5 Start the agent:

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

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

```
cliente@cliente-VirtualBox:~$ wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.14.2-1_amd64.deb && sudo WAZUH_MANAGER='10.0.0.11' WAZUH_AGENT_NAME='UbuntuVictima' dpkg -i ./wazuh-agent_4.14.2-1_amd64.deb
--2026-02-09 19:46:32-- https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.14.2-1_amd64.deb
Resolviendo packages.wazuh.com (packages.wazuh.com)... 99.84.9.92, 99.84.9.126, 99.84.9.57, ...
Conectando con packages.wazuh.com (packages.wazuh.com)[99.84.9.92]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 13121764 (13M) [application/vnd.debian.binary-package]
Guardando como: 'wazuh-agent_4.14.2-1_amd64.deb'

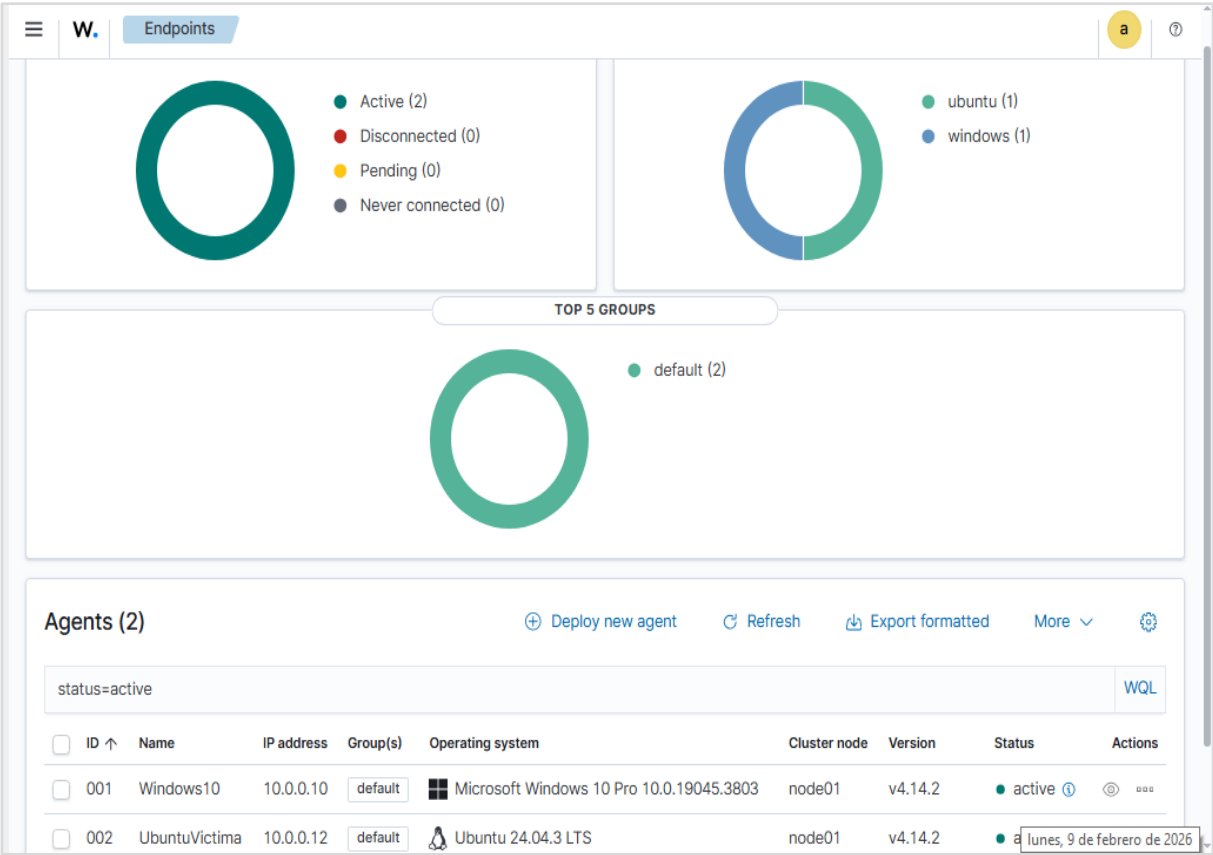
wazuh-agent_4.14.2-1_and 100%[=====] 12,51M 18,1MB/s en 0,7s
2026-02-09 19:46:33 (18,1 MB/s) - 'wazuh-agent_4.14.2-1_amd64.deb' guardado [13121764/13121764]

[sudo] contraseña para cliente:
Seleccionando el paquete wazuh-agent previamente no seleccionado.
(Leyendo la base de datos ... 150286 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../wazuh-agent_4.14.2-1_amd64.deb ...
Desempaquetando wazuh-agent (4.14.2-1) ...
Configurando wazuh-agent (4.14.2-1) ...
cliente@cliente-VirtualBox:~$ sudo systemctl daemon-reload
cliente@cliente-VirtualBox:~$ sudo systemctl enable wazuh-agent
Created symlink /etc/systemd/system/multi-user.target.wants/wazuh-agent.service → /usr/lib/systemd/system/wazuh-agent.service.
cliente@cliente-VirtualBox:~$ sudo systemctl start wazuh-agent
cliente@cliente-VirtualBox:~$
```

4.1.4 VERIFICACIÓN DE CONECTIVIDAD

Para concluir esta primera fase, fue necesario validar que toda la arquitectura de red se estaba comunicando de manera efectiva.

Tras el despliegue de ambos agentes, nos dirigimos al apartado *Agents Summary* dentro del *Dashboard* de Wazuh. Confirmamos visualmente que nuestros dos agentes (Windows10 y UbuntuVictima) figuraban con el estado: Active. Esto certificó que la comunicación RED <-> Servidor era correcta y que el sistema estaba preparado para la recolección de eventos, dando por finalizada la fase de infraestructura.



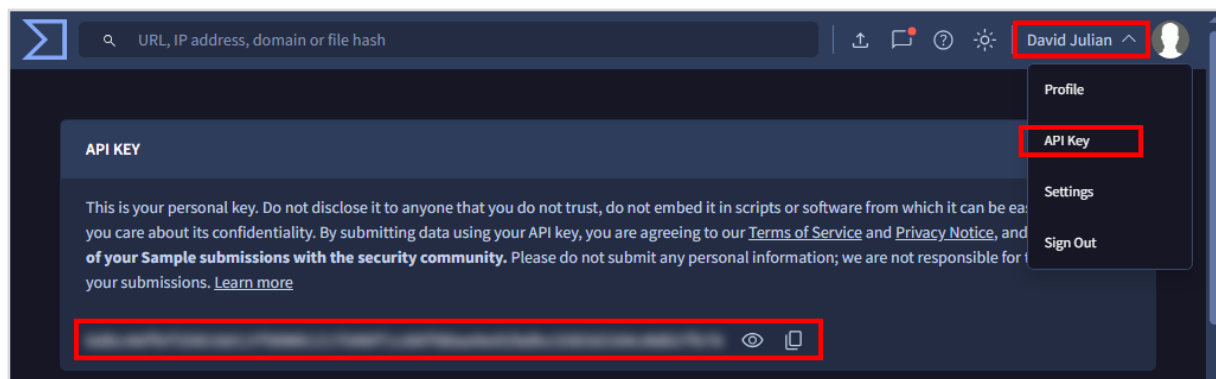
Fase 2: Detección de Malware (Integración con VirusTotal)

El objetivo de esta fase es doble: por un lado, configurar el "cerebro" del sistema (el servidor Wazuh) para que sea capaz de analizar amenazas de forma inteligente y, por otro, preparar la máquina víctima (Windows 10) para que vigile sus carpetas en tiempo real.

4.2.1 CONFIGURACIÓN DEL SERVIDOR (WAZUH MANAGER EN UBUNTU)

Necesitábamos que el Manager se conectara con la base de datos de Inteligencia de Amenazas (CTI) de VirusTotal. De este modo, en cuanto un agente reportase un archivo sospechoso, el sistema lo analizaría automáticamente sin intervención humana.

Para lograrlo, tras registrarnos en la plataforma de VirusTotal, obtuvimos nuestra API Key. Esta clave actúa como la llave maestra que autoriza a Wazuh a realizar consultas automatizadas en la nube.



A continuación, accedimos a la consola de Ubuntu y editamos el archivo de configuración central de Wazuh `/var/ossec/etc/ossec.conf`. Bajamos hasta el final del documento e inyectamos el siguiente bloque de código para vincular nuestra clave y forzar el formato JSON en las alertas:

```
<localfile>
<log_format>syslog</log_format>
<location>/var/ossec/logs/active-responses.log</location>
</localfile>

<integration>
<name>virustotal</name>
<api_key>TU_CLAVE_REAL_AQUI</api_key>
<group>syscheck</group>
<alert_format>json</alert_format>
</integration>
```

Para aplicar estos cambios, guardamos el archivo y reiniciamos el servicio del *manager* mediante el comando:

```
sudo systemctl restart wazuh-manager.
```

4.2.2 PREPARACIÓN DEL ENDPOINT (WINDOWS 10)

En este punto, preparamos el terreno para que el *malware* "sobreviviera" lo justo en el sistema operativo para ser cazado por nuestro SIEM.

Evasión del Antivirus: Desactivamos temporalmente la protección en tiempo real de Windows Defender mediante PowerShell con privilegios de Administrador. Si no lo hacíamos, Defender fulminaría el archivo malicioso antes de que nuestro agente tuviera tiempo de procesarlo.

```
Set-MpPreference -DisableRealtimeMonitoring $true
```

Configuración del Agente (FIM): Creamos una carpeta en la raíz del sistema llamada C:\HACKING que actuaría como nuestro *Honeypot* o trampa. Acto seguido, editamos el archivo de configuración local del agente

```
C:\Program Files (x86)\ossec-agent\ossec.conf.
```

Para activar el *File Integrity Monitoring* (FIM), buscamos la etiqueta <syscheck> y añadimos la siguiente línea:

```
<directories check_all="yes"  
realtime="yes">C:\HACKING</directories>
```

```

PS C:\Windows\system32> notepad "C:\Program Files (x86)\ossec-agent\ossec.conf"
PS C:\Windows\system32>

ossec.conf: Bloc de notas
Archivo Edición Formato Ver Ayuda

<!-- Security Configuration Assessment -->
<sca>
  <enabled>yes</enabled>
  <scan_on_start>yes</scan_on_start>
  <interval>12h</interval>
  <skip_nfs>yes</skip_nfs>
</sca>

<!-- File integrity monitoring -->
<syscheck>

  <disabled>no</disabled>

  <!-- Frequency that syscheck is executed default every 12 hours -->
  <frequency>43200</frequency>
  <directories check_all="yes" realtime="yes">C:\HACKING</directories>

  <!-- Default files to be monitored. -->
  <directories recursion_level="0" restrict="regedit.exe$|system.ini$|win.ini$" %WINDIR%<

  <directories recursion_level="0" restrict="at.exe$|attrib.exe$|cacls.exe$|cmd.exe$|even<
  <directories recursion_level="0">%WINDIR%\SysNative\drivers\etc</directories>
  <directories recursion_level="0" restrict="WMTC.exe$" %WINDIR%\SysNative\wham</director
  
```

Explicación técnica de la regla: No le indicamos a Wazuh que vigile todo el disco duro por cuestiones de rendimiento, sino exclusivamente nuestra carpeta objetivo. El parámetro `check_all="yes"` instruye al módulo para registrar cualquier vector de cambio (tamaño, permisos o *hash* criptográfico). Por su parte, `realtime="yes"` es la directiva crítica de esta demostración: enlaza al agente directamente con el *kernel* de Windows, detectando la alteración en el milisegundo exacto en que ocurre.

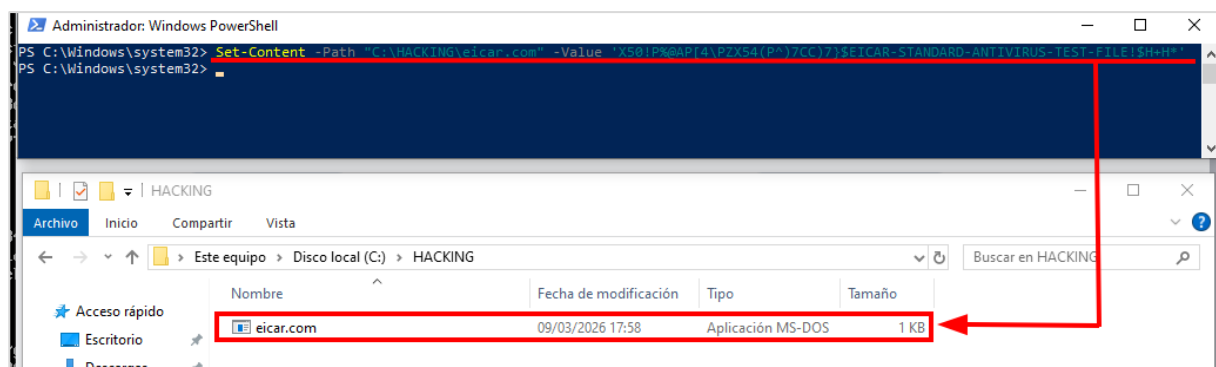
Finalmente, reiniciamos el servicio del agente desde PowerShell y dando comienzo a la vigilancia.

```
Restart-Service -Name WazuhSvc
```

4.2.3 SIMULACIÓN DE ATAQUE Y DETECCIÓN CRÍTICA

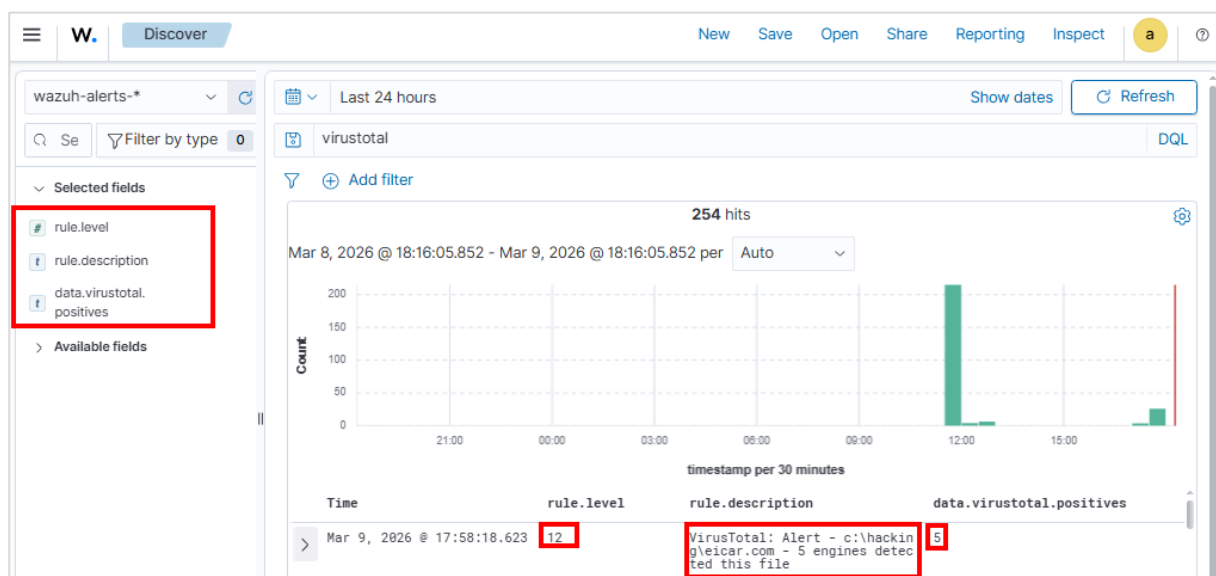
Con la infraestructura lista, procedimos a la inyección del *malware*. Simulamos la entrada de una amenaza directamente en el directorio monitorizado introduciendo la firma estándar de prueba antivirus (EICAR) a través de PowerShell:

```
Set-Content -Path "C:\HACKING\eicar.com" -Value 'X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*'
```



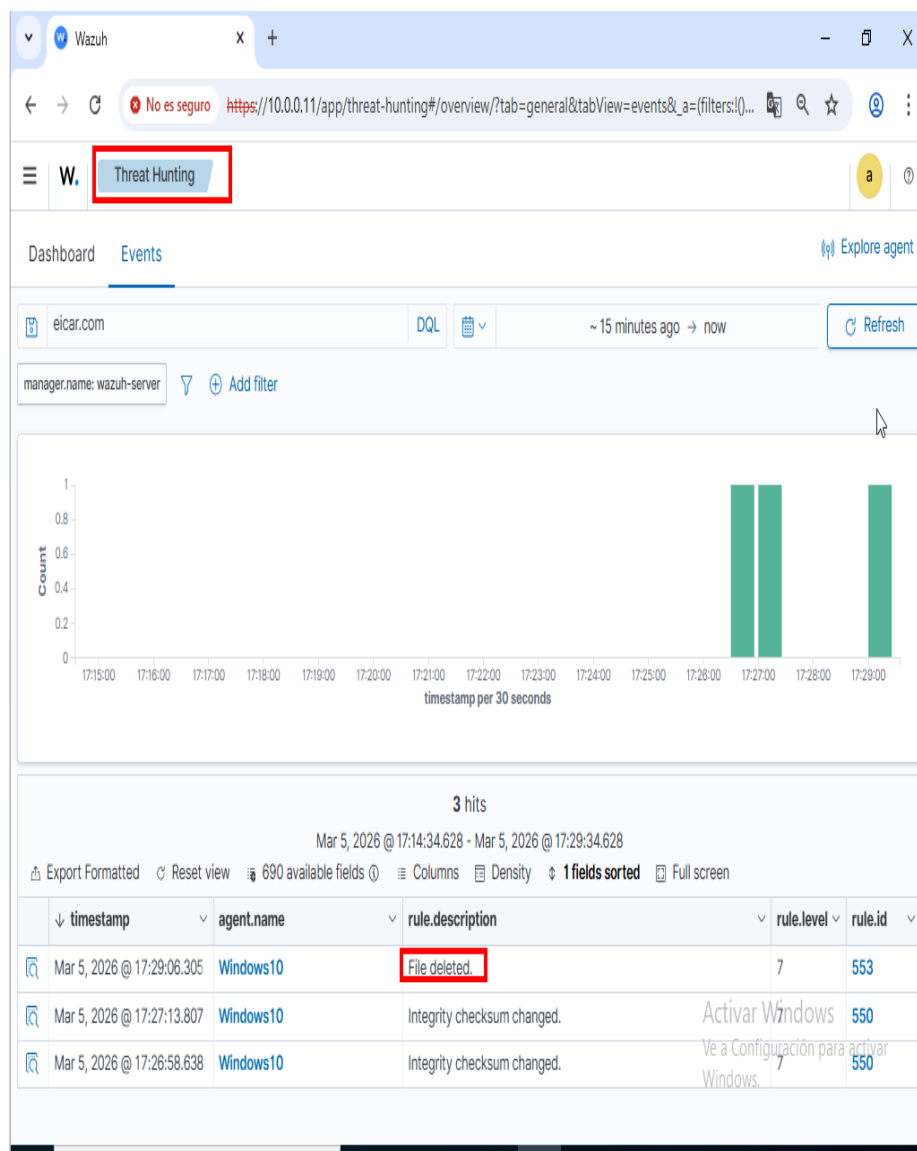
Verificación en el Dashboard:

El éxito de la configuración se hizo evidente de forma instantánea. Al acceder al panel de alertas, observamos que el SIEM había generado automáticamente una alerta de Nivel 12, de severidad crítica.



Al analizar las evidencias extraídas del panel Discover y el módulo de Threat Hunting, pudimos validar la eficiencia de nuestra arquitectura gracias a los siguientes hitos:

- **Latencia Nula:** El agente logró leer el archivo `c:\hacking\eicar.com` en modo realtime y calculó sus hashes (MD5, SHA1, SHA256). La marca de tiempo (17:58:18.623) coincidió de forma exacta con el instante de su creación.
- **Inteligencia Confirmada:** El servidor central recibió el hash y lo consultó mediante la API. El campo `data.virustotal.positives: 5` confirmó que cinco motores antivirus externos detectaron la amenaza simultáneamente.
- **Trazabilidad Forense:** Observamos en el registro un tercer evento etiquetado como `File deleted`. Esto indica que, a pesar de nuestras exclusiones, algún mecanismo de seguridad residual de Windows en la nube acabó borrando el archivo. Sin embargo, nuestro módulo FIM demostró ser más rápido, logrando registrar el incidente crítico antes de que la amenaza desapareciera.



Windows10 [Corriendo] - Oracle VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

Wazuh

No es seguro https://10.0.0.11/app/threat-hunting#/overview/?tab=general&tabView=events&_a=(filters:[]...

Threat Hunting

Dashboard Events

eicar.com DQL

manager.name: wazuh-server Add filter

Count

timestamp

Mar 5, 2026 @ 17:14:34.628

Export Formatted Reset view 690 available fields Columns

timestamp	agent.name	rule.description
Mar 5, 2026 @ 17:29:06.305	Windows10	File delete
Mar 5, 2026 @ 17:27:13.807	Windows10	Integrity c
Mar 5, 2026 @ 17:26:58.638	Windows10	Integrity c

Document Details

View surrounding documents View single document

Table JSON

_index	wazuh-alerts-4.x-2026.03.05
agent.id	001
agent.ip	10.0.0.10
agent.name	Windows10
decoder.name	syscheck_deleted
full_log	File 'c:\\hacking\\eicar.com' deleted Mode: realtime
id	1772728146.8723776
input.type	log
location	syscheck
manager.name	wazuh-server
rule.description	File deleted.
rule.firedtimes	1
rule.gdpr	II_5.1.f
rule.gpg13	4.11
rule.groups	ossec, syscheck, syscheck_entry_delet ed, syscheck_file

Activar Windows
Ve a Configuración para activar Windows

Escribe aquí para buscar.

17°C 17:34 05/03/2026

Wazuh

Wazuh

No es seguro https://10.0.0.11/app/discover#/doc/wazuh-alerts-*/wazuh-alerts-4.x-2026.03.05?id=FbfTvpw...

Discoverwazuh-alerts-4.x-2026.03.05#FbfTvpwBb9fhKf2mRh5F

TableJSON

@timestamp	Mar 5, 2026 @ 17:27:13.887
_index	wazuh-alerts-4.x-2026.03.05
agent.id	001
agent.ip	10.0.0.10
agent.name	Windows10
decoder.name	syscheck_integrity_changed
full_log	File 'c:\hacking\leicar.com' modified Mode: realtime Changed attributes: size mtime md5 sha1 sha256 Size changed from '73' to '78' Old modification time was: '1772726685', now it is '1772726701' Old md5sum was: '30378988c6fac293a7e5b21f2042fe66' New md5sum is : 'abb7c3a531fbf9388815126108468ec9' Old sha1sum was: '93af34facde1e2ba8711d13940d3c82092dd7d4e' New sha1sum is : 'be086a7c05a6de6b2990c9851da5382ebd8cf98d' Old sha256sum was: 'febb79b4779f2e7b8d8bbc1e2bca512404f84acddb859e3c296a396b763100fd' New sha256sum is : '5816ea61f435291f0cb149b6bca77e1541141eef2a3013f40495f7878f43b7dd'
id	1772728033.8720490
input.type	log
location	syscheck
manager.name	wazuh-server
rule.description	Integrity checksum changed.
rule.firedtimes	2

Activar Windows
Ve a Configuración para activar Windows.

7. Pruebas y resultado

Tras lograr la detección exitosa en la fase anterior, nuestro objetivo final ha sido automatizar la contención de la amenaza. Para ello, hemos configurado un mecanismo de *Active Response* que elimina el *malware* de forma autónoma en el instante en que es identificado y confirmado por VirusTotal.

5.1 CONFIGURACIÓN DEL SERVIDOR (WAZUH MANAGER)

En primer lugar, definimos el "comando" (la acción a ejecutar) y el "gatillo" (la condición que dispara la acción) en el cerebro del SIEM.

Accedimos a la consola de Ubuntu y editamos el archivo de configuración global (sudo nano /var/ossec/etc/ossec.conf. Al final del documento, añadimos los siguientes bloques de código:

```
<command>
  <name>remove-threat</name>
  <executable>remove-threat.cmd</executable>
  <timeout_allowed>no</timeout_allowed>
</command>

<active-response>
  <disabled>no</disabled>
  <command>remove-threat</command>
  <location>local</location>
  <rules_id>87105</rules_id>
</active-response>
```

Con esta configuración, le indicamos al servidor que, siempre que se dispare la regla 87105 (la alerta crítica de Nivel 12 de VirusTotal), debe ejecutar localmente en la máquina afectada el *script* remove-threat.cmd. Finalmente, reiniciamos el *manager* para aplicar los cambios (sudo systemctl restart wazuh-manager).

```

Wazuh v4.14.2 [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
GNU nano 8.3 /var/ossec/etc/ossec.conf Modified

</ossec_config>
<ossec_config>
  <localfile>
    <log_format>journald</log_format>
    <location>journald</location>
  </localfile>

  <localfile>
    <log_format>audit</log_format>
    <location>/var/log/audit/audit.log</location>
  </localfile>

  <localfile>
    <log_format>syslog</log_format>
    <location>/var/ossec/logs/active-responses.log</location>
  </localfile>

  <localfile>
    <log_format>syslog</log_format>
    <location>/var/ossec/logs/active-responses.log</location>
  </localfile>

  <integration>
    <name>virstotal</name>
    <api_key>[REDACTED]</api_key>
    <group>syscheck</group>
    <alert_format>json</alert_format>
  </integration>

  <command>
    <name>remove-threat</name>
    <executable>remove-threat.cmd</executable>
    <timeout_allowed>no</timeout_allowed>
  </command>

  <active-response>
    <disabled>no</disabled>
    <command>remove-threat</command>
    <location>local</location>
    <rules_id>87105</rules_id>
  </active-response>
</ossec_config>

Help  Write Out  Where Is  Cut  Execute  Location  Undo  Set Mark  To Bracket  Previous
Exit  Read File  Replace  Paste  Justify  Go To Line  Redo  Copy  Where Was  Next
  
```

5.2 DESPLIEGUE DEL SCRIPT EN EL ENDPOINT (WINDOWS 10)

Debido a las estrictas restricciones de permisos que aplica Windows sobre el directorio de instalación del agente, generamos el *script* de borrado directamente desde una consola de PowerShell con privilegios de Administrador, apuntando a la ruta de binarios de Wazuh.

Primero, creamos el archivo `remove-threat.cmd` inyectando la instrucción `@echo off`. Esto sirve para que, al ejecutarse, solo se procese el resultado del *script* y se evite generar "ruido" o salidas innecesarias en la consola:

```
Set-Content -Path "C:\Program Files (x86)\ossec-agent\active-response\bin\remove-threat.cmd" -Value '@echo off'
```

A continuación, añadimos la orden que ejecutará la eliminación real del vector de ataque:

```
Add-Content -Path "C:\Program Files (x86)\ossec-agent\active-response\bin\remove-threat.cmd" -Value 'del /F /Q "C:\HACKING\eicar.com"'
```

Explicación de los parámetros utilizados:

- **del:** Comando del sistema para borrar (*delete*).
- **/F:** Fuerza el borrado, incluso si el archivo está marcado como de solo lectura.
- **/Q:** Activa el modo *Quiet* (silencioso). No pide confirmación al usuario por pantalla, lo que es vital para que Wazuh pueda actuar de forma 100% autónoma en segundo plano.

Por último, reiniciamos el servicio del agente local para asimilar el nuevo ejecutable:

```
Restart-Service -Name WazuhSvc
```

```
PS C:\Windows\system32> Set-Content -Path "C:\Program Files (x86)\ossec-agent\active-response\bin\remove-threat.cmd" -Value 'echo off'
PS C:\Windows\system32> Add-Content -Path "C:\Program Files (x86)\ossec-agent\active-response\bin\remove-threat.cmd" -Value 'del /F /Q "C:\HACKING\eicar.com"'
PS C:\Windows\system32> Restart-Service -Name WazuhSvc
```

5.3 PRUEBA DE CONCEPTO Y VERIFICACIÓN

Con todo el ecosistema preparado, lanzamos el ataque simulado definitivo inyectando nuevamente la firma EICAR en el directorio monitorizado:

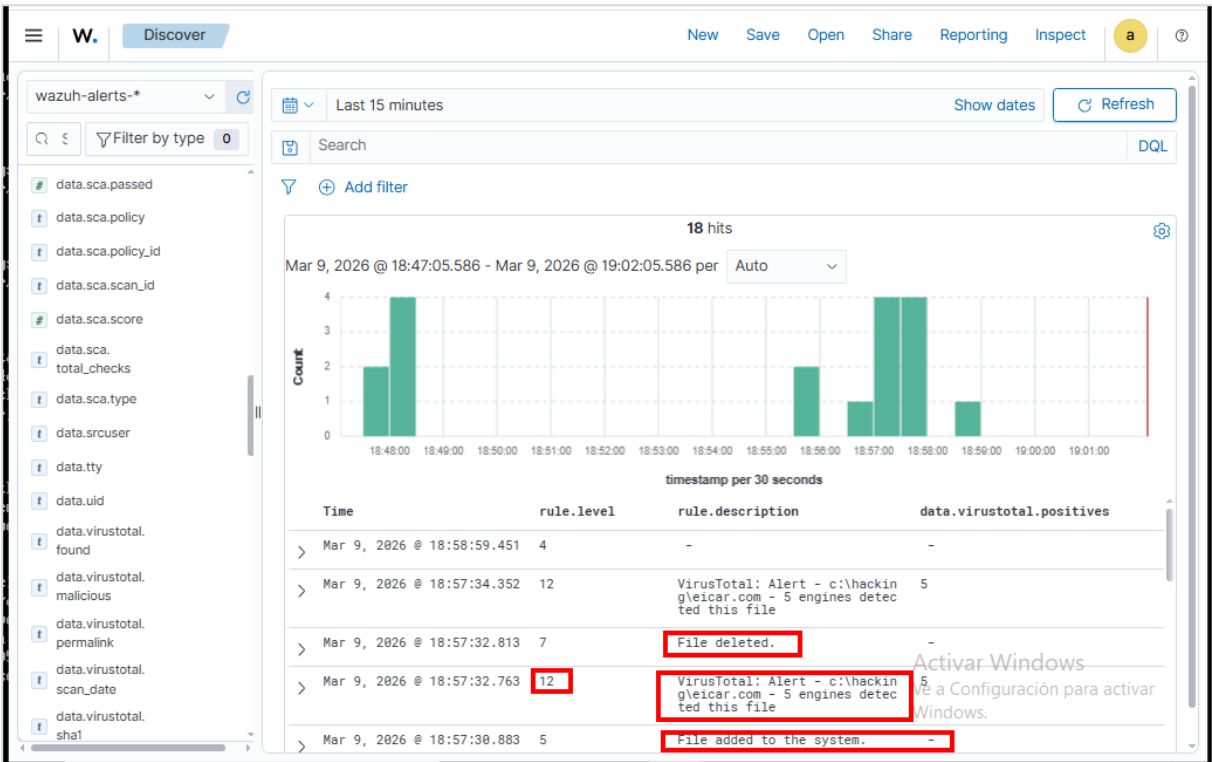
```
Set-Content -Path "C:\HACKING\eicar.com" -Value 'X50!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*'
```

```
PS C:\Windows\system32> Set-Content -Path "C:\HACKING\eicar.com" -Value 'X50!PN@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*'
PS C:\Windows\system32>
```

Análisis de resultados: Al monitorizar la carpeta C:\HACKING, observamos que recibió el archivo eicar.com de forma inmediata, pero acto seguido fue fulminado del sistema, confirmando la perfecta funcionalidad de nuestra Respuesta Activa.

Acudimos al panel de eventos para extraer la trazabilidad forense del incidente, obteniendo la siguiente línea temporal:

Tiempo	Evento	Nivel	Descripción
18:57:30	Added	5	El módulo FIM detecta la creación del archivo malicioso.
18:57:32	Alert	12	VirusTotal confirma la amenaza (Rule 87105).
18:57:32	Deleted	7	Active Response ejecuta el <i>script</i> y elimina el archivo.



7. Conclusiones y trabajos futuro

6.1 Conclusiones

La implementación de este laboratorio nos ha permitido validar empíricamente la viabilidad y eficacia de un Centro de Operaciones de Seguridad (SOC) completamente automatizado utilizando tecnologías de código abierto.

Como conclusión principal, hemos logrado cerrar con éxito el ciclo de vida de un incidente informático, superando las limitaciones de los modelos de seguridad tradicionales. Mediante la configuración del módulo *File Integrity Monitoring* (FIM), pasamos de una monitorización pasiva a obtener visibilidad granular y en tiempo real sobre los *endpoints*. Además, la integración de la API de VirusTotal dotó al SIEM de una capacidad analítica avanzada, permitiendo al sistema discernir de forma autónoma entre alteraciones legítimas y vectores de ataque reales.

El mayor logro de este proyecto ha sido demostrar la eficacia de las políticas de *Active Response*. Hemos comprobado que el tiempo de exposición de la infraestructura ante una amenaza confirmada se puede reducir a un margen de apenas dos segundos. Esto elimina el cuello de botella que supone el factor de reacción humano, reduce drásticamente el Tiempo Medio de Respuesta (MTTR) y demuestra que es posible desplegar una arquitectura capaz de autodefenderse y garantizar la continuidad del negocio.

6.2 Trabajos futuros

Este proyecto constituye una base arquitectónica sólida y funcional, pero el ámbito de la ciberseguridad defensiva está en constante evolución. Para futuras iteraciones y con el objetivo de acercar esta solución a un entorno de producción a gran escala, planteamos las siguientes líneas de desarrollo continuo:

- **Integración SOAR (Security Orchestration, Automation and Response):** Vincular las alertas críticas de Wazuh con plataformas de comunicación corporativa (como Telegram o Slack) y sistemas de *ticketing* (como Jira). De este modo, el equipo humano recibiría una notificación instantánea en su móvil en el mismo momento en que el sistema neutraliza la amenaza, mejorando la auditoría del incidente.
- **Expansión de Sensores y Reglas:** Ampliar las políticas de monitorización para que el agente no solo vigile directorios locales, sino también claves críticas del Registro de Windows y tráfico de red anómalo. Esto permitiría al sistema detectar y neutralizar ataques más sofisticados, como el *malware Fileless* (aquel que reside exclusivamente en la memoria RAM y no deja rastro en el disco duro).
- **Escalabilidad y Alta Disponibilidad:** Migrar la arquitectura central, actualmente alojada en un entorno de red local virtualizado, hacia una infraestructura en la nube (AWS o Azure). La implementación de un clúster de servidores Wazuh permitiría soportar el registro de cientos de *endpoints* simultáneos, garantizando el balanceo de carga y la tolerancia a fallos en redes corporativas complejas.

BIBLIOGRAFÍA

WAZUH 01. INSTALACIÓN de WAZUH en UBUNTU SERVER

: <https://www.youtube.com/watch?v=jv3Z6zSq4gU>

VirusTotal. (2024). VirusTotal API v3 Documentation. Recuperado de <https://docs.virustotal.com/reference/overview>

Wazuh, Inc. (2024). Wazuh Documentation: Active Response. Recuperado de <https://documentation.wazuh.com/current/user-manual/capabilities/active-response/index.html>

European Institute for Computer Antivirus Research (EICAR). (2024). Download Anti Malware Testfile. Recuperado de <https://www.eicar.org/download-anti-malware-testfile/>

Google. (2024). *Gemini* [Modelo de lenguaje grande]. Recuperado de <https://gemini.google.com/>