Risk mitigation in vulnerable Linux servers: the role of OSSEC in attack prevention

Smulever Federico Manuel, Pinat Juan Matías, Gonzalez Rodrigo Sebastián, Ríos Leopoldo José[1]

[1]Facultad de Ciencias Exactas, Naturales y Agrimensura. Universidad Nacional del Nordeste.
Corrientes, Argentina.

federicosmulever@gmail.com, gonzalezrodrigo0020@gmail.com, juanmatiaspinat@gmail.com, ljr@comunidad.unne.edu.ar

Abstract. The protection of vulnerable Linux servers is essential in critical infrastructure environments. This study analyzes the impact of OSSEC on threat detection and mitigation in Linux servers with limited security configurations. Through practical implementation, its capabilities to identify unauthorized access and suspicious patterns are evaluated, as well as the effectiveness of its real-time alerts to strengthen defense against attacks. This work also addresses initial configuration, active monitoring, and observed limitations in OSSEC, highlighting its role as an effective solution in environments with minimal protection resources.

Keywords. OSSEC. Security on Linux servers. Intrusion detection. Attack prevention. Proactive monitoring

Mitigación de riesgos en servidores Linux vulnerables: el rol de OSSEC en la prevención de ataques

Abstract. La protección de servidores Linux vulnerables es esencial en entornos de infraestructura crítica. Este estudio analiza el impacto de OSSEC en la detección y mitigación de amenazas en servidores Linux con configuraciones de seguridad limitadas. A través de una implementación práctica, se evalúan sus capacidades para

identificar accesos no autorizados y patrones sospechosos, así como la efectividad de sus alertas en tiempo real para fortalecer la defensa ante ataques. Este trabajo aborda además la configuración inicial, el monitoreo activo y las limitaciones observadas en OSSEC, resaltando su rol como solución efectiva en entornos con recursos de protección mínimos.

Keywords. OSSEC. Seguridad en servidores Linux. Detección de intrusiones. Prevención de ataques. Monitoreo proactivo.

1- Introducción

Durante el transcurso de la materia Redes de datos, de la carrera de Licenciatura en Sistemas de Información, dictada en la Universidad Nacional del Nordeste en el año 2024, hemos trabajado con servidores Linux y hemos estudiado las diferentes funcionalidades y servicios que estos pueden prestar. En sintonía con los contenidos dictados, y por incentivo del claustro docente, este trabajo es resultado consecuente.

Los servidores Linux han cimentado su importancia como soporte esencial de infraestructuras digitales en sectores empresariales, académicos y gubernamentales. Su flexibilidad, estabilidad y escalabilidad los convierten en opción preferida para administrar datos y para aplicaciones críticas. Sin embargo, suelen configurarse rápidamente con prácticas de seguridad mínimas, dejando brechas que atacantes explotan.

Los principales riesgos incluyen ataques de fuerza bruta (forzar credenciales), inyecciones de código (ejecutar comandos maliciosos) y ataques DoS (sobrecargar el servidor). La defensa efectiva es crucial para preservar integridad y disponibilidad.

OSSEC (Open Source Security Event Correlator) surge como respuesta específica. Es una herramienta de código abierto diseñada para monitorear y proteger en tiempo real servidores Linux frente a actividades sospechosas. Funciona en entornos críticos analizando eventos y correlacionando patrones para emitir alertas inmediatas y respuestas automáticas. Es adaptable, escalable y ligero, sin requerir grandes recursos de hardware, convirtiéndolo en solución confiable y accesible para abordar deficiencias de seguridad en configuraciones vulnerables.

2- Estructura y Operatividad del Sistema OSSEC

Servidor OSSEC y Agentes. El servidor OSSEC centraliza el monitoreo y análisis de seguridad, actuando como núcleo de la infraestructura de detección. Recibe datos de agentes instalados en dispositivos monitoreados, procesa información mediante correlación de eventos y aplica reglas que identifican patrones de amenazas, generando alertas ante comportamientos anómalos[5][8]. En configuración sin agentes, opera independientemente ejecutando comprobaciones locales[5].

Los agentes son programas ligeros instalados en dispositivos a monitorear. Recopilan datos de registros del sistema y los envían al servidor central. Funcionan en diversos sistemas operativos, permitiendo supervisar entornos heterogéneos[7]. En modo local, OSSEC no requiere agentes y recopila datos directamente en el servidor.

Monitor de Integridad de Archivos (FIM). Realiza comprobaciones periódicas en archivos críticos del sistema, detectando cambios inesperados en configuraciones y archivos importantes. Genera alertas ante modificaciones sospechosas, siendo esencial para identificación temprana de accesos no autorizados o modificaciones maliciosas[6].

Sistema de Correlación de Eventos. Aplica reglas predefinidas y personalizables para analizar eventos de seguridad de múltiples sistemas. Correlaciona eventos según patrones específicos de actividad sospechosa, clasificándolos por nivel de riesgo y activando alertas automáticas. Es el eje central de la capacidad de detección en tiempo real[5].

Alertas y Notificaciones. Permite configurar notificaciones automáticas para eventos críticos con distintos niveles de gravedad. Soporta alertas por correo electrónico y otras plataformas, configurables para desencadenar respuestas automáticas como bloqueo de IPs maliciosas o finalización de sesiones sospechosas[8]. Estas respuestas automáticas ayudan a contener amenazas inmediatamente.

Bases de Datos de Firmas y Reglas Personalizables. Contienen reglas de detección de amenazas personalizables según necesidades específicas. Permiten identificar comportamientos maliciosos y reaccionar según estructura de amenazas detectadas. Administradores pueden agregar o modificar reglas para mejorar efectividad[5][6].

Lista Blanca (Whitelist). Configura direcciones IP y usuarios autorizados para evitar alertas o respuestas automáticas, minimizando falsos positivos de sistemas confiables. Permite acceso sin restricciones solo a dispositivos y usuarios verificados[5].

Interacción de los Componentes y Flujo de Información. El flujo inicia con agentes recopilando datos que envían al servidor OSSEC para procesamiento mediante

reglas de bases de datos. FIM verifica archivos clave periódicamente, activando alertas ante modificaciones inesperadas[5][7][8].

Cuando el sistema de correlación identifica amenazas, genera alertas enviadas al administrador mediante sistema de notificaciones. Según gravedad, ejecuta respuestas automáticas para mitigar riesgos. Todos los eventos se almacenan para seguimiento histórico y análisis de incidentes.

3- Metodología de implementación

Preparación del Entorno. Antes de instalar OSSEC, es necesario asegurarse de que el servidor Linux cuente con las dependencias requeridas. Para ello, se ejecutan los siguientes comandos que actualizan el sistema e instalan las bibliotecas y herramientas necesarias: 1- *sudo apt update*, 2- *sudo apt install gcc make curl libevent-dev unzip -y*, 3- *sudo apt install libpcre2-dev*, 4- *sudo apt install zlib1g-dev*, 5- *sudo apt install libssl-dev* y 6- *sudo apt install libsystemd-dev*

Instalación del Software. Una vez preparado el entorno, descargamos OSSEC utilizando el siguiente comando para obtener el paquete desde el repositorio oficial: *wget https://github.com/ossec/ossec-hids/archive/master.zip*

Desempaquetado e Instalación de OSSEC. Con el archivo descargado, procedemos a descomprimirlo: *unzip master.zip* y acceder al directorio OSSEC: *cd ossec-hids-master* para iniciar la instalación. Para iniciar la instalación, utilizamos el script: *$ sudo ./install.sh*, que guía al usuario a través de una serie de preguntas para configurar el sistema de manera personalizada. Durante la instalación, se presentan opciones importantes, que permiten configurar OSSEC según las necesidades del entorno. (fig.1)

Tipo de instalación. El instalador solicita seleccionar el tipo de instalación. Las opciones disponibles son: 1- Servidor (Server): Configura OSSEC como servidor central para recibir y procesar datos de otros agentes. 2- Agente (Agent): Configura OSSEC como agente, permitiendo enviar registros y eventos al servidor central. 3- Local: Permite ejecutar OSSEC en modo local sin necesidad de comunicación con un servidor.

4- Simulación en OSSEC: Monitoreo y Respuesta ante Amenazas

Contexto del Escenario. Trabajamos con un servidor Linux autoalojado en una pequeña oficina o en casa, configurado para ofrecer servicios básicos de Apache y acceso SSH para administración remota. Este servidor no está en la nube y carece de una IP pública, por lo que solo es accesible desde la red local. El servidor utiliza una IP

privada y permite conexiones SSH en el puerto 22, facilitando el mantenimiento remoto del administrador y el acceso web para los dispositivos de la red. Sin embargo, con medidas de seguridad mínimas, este servidor es vulnerable a un atacante que obtenga acceso a la red local y pueda realizar un ataque de fuerza bruta al servicio SSH, una práctica común en redes internas.

Ataque por fuerza bruta. Un ataque de fuerza bruta es un método de hackeo en el que un atacante intenta repetidamente diferentes combinaciones de contraseñas hasta encontrar la correcta para acceder a un sistema o servicio.

No se deben subestimar los ataques de fuerza bruta porque, aunque parecen simples, pueden ser efectivos si las contraseñas son débiles o comunes (fig 24)[11]. Además, herramientas automatizadas permiten probar millones de combinaciones rápidamente, aumentando las probabilidades de éxito.

Un artículo de Minery Report del año 2025 subraya la intensidad de estos ataques al mencionar que los sistemas automáticos actuales son capaces de lanzar "cientos de miles de intentos por minuto"[9]. También pueden ser muy efectivo, así lo refleja otro artículo de Infobae, el cual, relata que ataques de fuerza bruta a gran escala han comprometido millones de dispositivos VPN, firewalls y otros dispositivos de seguridad perimetral.[10]

Preparación del Entorno de Simulación. Se configuró un entorno controlado con dos máquinas virtuales para simular un ataque de fuerza bruta realista: máquina atacante con Kali Linux (Debian Testing) y máquina víctima con Ubuntu 14.04, replicando un servidor Linux con vulnerabilidades específicas. Ambas se ejecutan en VMware con configuración de red "host only" para crear red privada entre máquinas virtuales y host físico.

Para configurar el entorno vulnerable, se eliminaron todas las reglas del firewall en Ubuntu mediante *sudo iptables -F*, permitiendo observar el potencial completo de OSSEC en entornos con medidas de seguridad mínimas. Se verificó la eliminación mediante *sudo iptables -S* (fig 2), también que SSH estuviera ejecutándose en ambas máquinas (fig 3) y la conectividad bidireccional mediante pruebas de ping (fig 4).

Recopilación Activa de Información. Antes de ejecutar el ataque, se simuló un escenario realista donde el atacante debe identificar el objetivo en la red. Utilizando Nmap desde Kali Linux, se realizó un escaneo de descubrimiento de nodos en la red mediante el comando *nmap -sS -O 192.168.137.0/24*. El reporte resultante identificó la máquina objetivo con dirección IP 192.168.137.131, estimando que su sistema operativo es Linux 3.4-4.14 con una confianza del 98%. El escaneo reveló múltiples puertos abiertos, confirmando que el puerto 22 está habilitado con el servicio SSH en

ejecución, estableciendo así las condiciones ideales para un ataque de fuerza bruta. (fig 5)

Preparación del Ataque por Fuerza Bruta. Se utilizó Hydra para ejecutar el ataque automatizado contra SSH. OpenSSH presenta barrera "informativa" significativa al responder siempre "permission denied" ante intentos fallidos, sin distinguir entre usuarios válidos e inválidos. Esta limitación obliga a implementar una estrategia con dos diccionarios especializados: unix_users.txt (usuarios comunes Unix/Linux como root, ubuntu, admin) y rockyou.txt (millones de contraseñas reales de filtraciones ordenadas por frecuencia) , ya que no es posible determinar previamente qué usuarios son válidos en el sistema objetivo. Hydra prueba sistemáticamente todas las combinaciones hasta encontrar credenciales válidas. Se verificó que unix_users.txt contuviera el usuario 'vagrant' (sabemos que ese es un nombre de usuario en el servidor Linux) y rockyou.txt incluyera variaciones de esta palabra, anticipando posible correlación usuario-contraseña (fig 6).

Lanzamiento del Ataque Inicial. El primer ataque se ejecutó mediante el comando *hydra -L /home/kali/usuarios.txt -P /home/kali/diccionario.txt -t 4 -V -f ssh://192.168.137.131 2>&1*, configurado para utilizar ambos diccionarios con 4 hilos simultáneos en modo verbose y detenerse al encontrar la primera credencial válida. (fig 7)

Durante la ejecución, Hydra comenzó a probar sistemáticamente diferentes combinaciones de credenciales contra el objetivo 192.168.137.131. El proceso se caracterizó por intentos repetitivos hacia el servicio SSH. (fig 8)

Paralelamente, la máquina Ubuntu registraba cada intento fallido en */var/log/auth.log*, generando entradas detalladas que incluían la fecha y hora del intento, el nombre del usuario, la dirección IP de origen (192.168.137.130) y una descripción del evento de fallo de autenticación. (fig 9)

Análisis del Tráfico de Red. Mediante una captura de tráfico con Wireshark desde Kali Linux, se observó el comportamiento característico del ataque de fuerza bruta. El análisis reveló comunicación constante entre 192.168.137.130 (Kali Linux - atacante) y 192.168.137.131 (Ubuntu 14.04 - objetivo) utilizando el puerto 22 (SSH) como destino constante y el protocolo SSHv2 predominante. Se observan conexiones extremadamente frecuentes que ocurren cada pocos milisegundos con una duración muy corta por cada intento, creando un patrón repetitivo y sistemático característico de herramientas automatizadas. (fig 10)

Finalización del Ataque Exitoso. Después de aproximadamente 40 minutos de ejecución, el ataque concluyó exitosamente con las siguientes estadísticas: inicio a las

17:23:53 y finalización a las 18:03:05 del 27 de junio de 2025, con un total de 2010 combinaciones usuario/contraseñas probadas. La herramienta operó con 4 hilos simultáneos, procesando aproximadamente 503 intentos por tarea. (fig 11)

El resultado final reveló las credenciales válidas: usuario 'vagrant' con contraseña 'vagrant'. La verificación de estas credenciales se realizó mediante una conexión SSH exitosa, confirmando el acceso completo al sistema Ubuntu 14.04 y demostrando la vulnerabilidad crítica del servidor ante ataques de fuerza bruta automatizados. (fig 12)

Empleando OSSEC para Mitigar Ataques. Tras establecer la vulnerabilidad del sistema, se procedió a implementar OSSEC en el servidor Linux vulnerable. La puesta en funcionamiento se realizó mediante el comando *sudo /var/ossec/bin/ossec-control start*. (fig 13)

Respuestas Activas Automatizadas. Una de las capacidades más destacadas y poderosas de OSSEC radica en su sistema de respuestas activas automatizadas, que representa el núcleo de su efectividad defensiva. Para examinar esta funcionalidad crítica, se empleó el comando *sudo grep -A 5 -B 5 "active-response" /var/ossec/etc/ossec.conf*, que permite visualizar las configuraciones de respuesta automática junto con su contexto en el archivo de configuración principal.

El sistema está configurado para ejecutar respuestas inmediatas y automatizadas cuando detecta eventos de seguridad de nivel 3 o superior, sin requerir intervención humana alguna. Esta capacidad transforma a OSSEC de un simple monitor pasivo a un sistema de defensa activa y proactiva.

La configuración implementa dos respuestas activas simultáneas y complementarias que actúan en capas defensivas distintas. La primera respuesta activa ejecuta el comando host-deny, diseñado específicamente para añadir automáticamente la IP atacante al archivo /etc/hosts.deny, efectuando un bloqueo a nivel sistema que impide cualquier tipo de acceso desde esa dirección IP. La segunda respuesta implementa firewall-drop, que bloquea la IP directamente en las reglas de iptables del sistema, creando una barrera adicional a nivel de firewall por el mismo período de tiempo. (fig 14)

Reglas de Detección de Ataques SSH. La efectividad de OSSEC para detectar ataques de fuerza bruta reside en dos reglas específicas contenidas en /var/ossec/rules/sshd_rules.xml. La regla 5716 actúa como detector individual con nivel 5, analizando mensajes SSH que contengan patrones como "Failed" o "error: PAM: Authentication", clasificando cada evento como "SSHD authentication failed" dentro del grupo authentication_failed. Esta regla utiliza la expresión <match>^Failed|^error: PAM: Authentication</match> para identificar fallos de autenticación. (fig 15)

La regla 5720 funciona como escalador automático con nivel 10, monitoreando la ocurrencia de la regla 5716 mediante <if_matched_sid>5716</if_matched_sid> y <same_source_ip /> con <frequency>6</frequency>. Esta regla se activa cuando detecta 6 eventos consecutivos de la regla 5716 desde la misma dirección IP, describiéndolos como "Multiple SSHD authentication failures" bajo el grupo authentication_failures. (fig 16)

OSSEC supervisa específicamente /var/log/auth.log en Ubuntu y /var/log/syslog donde SSH registra todos los intentos de autenticación. El sistema analiza estos archivos en tiempo real, detectando patrones maliciosos como múltiples fallos de login. Cuando Hydra intenta credenciales incorrectas generando logs SSH del tipo "Failed password for user", cada intento activa la regla 5716 individualmente hasta que el sexto fallo consecutivo desde la misma IP dispara la regla 5720, cuyo nivel 10 supera el umbral configurado de nivel 3, activando automáticamente las respuestas de bloqueo host-deny y firewall-drop que protegen el sistema bloqueando la IP atacante por 600 segundos.

Ataque Mitigado por OSSEC. Para demostrar la efectividad de OSSEC en un escenario más crítico, se ejecutó un segundo ataque proporcionando directamente el username correcto 'vagrant', representando una situación donde el atacante posee información parcial del sistema objetivo, lo cual incrementa significativamente las probabilidades de éxito. El comando utilizado fue *hydra -l vagrant -P /home/kali/diccionario.txt -t 4 -V -f ssh://192.168.137.131 2>&1*, configurado para atacar específicamente al usuario conocido con el diccionario completo de contraseñas. (fig 17)

Los resultados del ataque fueron reveladores: Hydra intentó una conexión SSH con las credenciales vagrant/vagrant (credenciales que sabemos son correctas del ataque anterior), sin embargo, esta vez no logró identificar una contraseña válida. El ataque finalizó reportando "0 of 1 target completed, 0 valid password found", confirmando que esta vez el ataque no ha tenido éxito. (fig 18)

Monitoreo en Tiempo Real de la Defensa. Durante la ejecución del segundo ataque, se monitoreó simultáneamente la respuesta de OSSEC mediante la observación en tiempo real del archivo de alertas usando *sudo tail -f /var/ossec/logs/alerts/alerts.log*. Este log mostró OSSEC detectando inmediatamente el ataque de Hydra contra el usuario "vagrant" desde la IP 192.168.137.130. Cada entrada representaba un intento fallido de SSH que activaba la regla 5716 (nivel 5) clasificado como "SSHD authentication failed". Los múltiples intentos registrados en milisegundos desde diferentes puertos (56500, 56512, 56486, 56502) confirmaron el patrón automatizado típico de Hydra. (fig 19)

Paralelamente, el monitoreo de los registros del sistema mediante *sudo tail -f /var/log/auth.log* reveló la fuente de datos que OSSEC estaba analizando. El archivo /var/log/auth.log contiene los eventos brutos de autenticación sin procesar, mostrando mensajes como "Failed password for vagrant from 192.168.137.130", mientras que alerts.log representa el resultado procesado de OSSEC que interpreta y clasifica esos eventos según sus reglas de detección. (fig 20)

Bloqueo de la dirección ip del atacante. Cuando OSSEC detectó el sexto intento fallido consecutivo desde la IP 192.168.137.130, la regla 5720 con nivel 10 superó el umbral configurado, activando automáticamente el sistema de active-response previamente configurado. La verificación mediante *sudo grep -i "192.168.137.130" /var/ossec/logs/active-responses.log* confirmó la ejecución de ambas respuestas: host-deny.sh add - 192.168.137.130 para el bloqueo a nivel sistema y firewall-drop.sh add - 192.168.137.130 para el bloqueo en firewall, materializando prácticamente el mecanismo de defensa automática analizado anteriormente. (fig 21)

Verificación de la Efectividad del Bloqueo. La efectividad del sistema de respuestas activas se confirmó mediante múltiples métodos de verificación. Las pruebas de conectividad posteriores usando ping desde Kali Linux hacia Ubuntu demostraron pérdida completa de conectividad: 40 paquetes transmitidos, 0 recibidos, 100% de pérdida de paquetes. La captura de tráfico ICMP con Wireshark confirmó que Kali (192.168.137.130) enviaba echo requests hacia Ubuntu (192.168.137.131), pero todos los paquetes indicaban "no response" sin generar echo replies de retorno. (fig 22)

5- Otras funcionalidad de OSSEC

Ya hemos visto lo práctico que resulta esta herramienta para proteger nuestros servidores de ataques externos, aún así, no queremos dejar de darle su debida impronta a las otras funcionalidades que OSSEC nos proporcionará. El sistema ofrece un amplio espectro de capacidades defensivas que consolidan una solución integral de seguridad.

OSSEC proporciona detección de escalamiento de privilegios, monitoreo de integridad de archivos críticos y control de acceso y permisos, protegiendo contra modificaciones maliciosas y manipulación no autorizada. En el ámbito de seguridad de red, implementa detección de escaneos de puertos, protección contra inyecciones SQL y supervisión de actividad de usuarios privilegiados, evitando tanto amenazas externas como abusos internos.

La herramienta extiende su protección mediante alertas sobre dispositivos USB no autorizados, detección de ataques de denegación de servicio (DoS) y control de accesos remotos por ubicación geográfica. Finalmente, OSSEC realiza supervisión continua de

configuraciones de seguridad, asegurando el cumplimiento con políticas establecidas. Esta diversidad funcional convierte a OSSEC en una solución comprehensiva que aborda múltiples vectores de amenaza desde una plataforma unificada [5][6][7].

6- Comparación de OSSEC con otras herramientas de monitoreo de seguridad

[5][6][7][OSSEC destaca por su capacidad de detección de intrusos en tiempo real, análisis de logs y respuesta activa, pero presenta limitaciones comparado con alternativas como Splunk, Snort, Suricata y Wazuh[5][6][7]. A diferencia de Splunk, conocido por análisis avanzado de datos y visualización intuitiva, OSSEC carece de interfaz gráfica integrada, dificultando la interpretación de datos. Sin embargo, esta ausencia lo hace más ligero y sencillo de instalar, ideal para entornos con recursos limitados.

Wazuh, basado en OSSEC, incluye mejoras significativas como interfaz gráfica mediante Elastic Stack para visualizar y gestionar alertas intuitivamente, además de herramientas específicas para cumplimiento de políticas de seguridad (PCI-DSS y GDPR), ofreciendo ventajas en entornos corporativos. OSSEC mantiene simplicidad sin dependencias externas, ideal para configuraciones rápidas en entornos pequeños.

Comparado con herramientas de detección de red como Snort y Suricata, que se especializan en inspección de tráfico, OSSEC se enfoca en detección a nivel de host, limitando su alcance en monitoreo de red en tiempo real. Esta especialización le permite detectar cambios en archivos y actividades anómalas que herramientas orientadas a red podrían pasar por alto. Aunque Wazuh ofrece capacidades ampliadas, OSSEC mantiene la ventaja de ser más ligero y fácil de implementar, siendo opción sólida para quienes buscan herramienta efectiva y adaptable sin complejidades de infraestructuras más avanzadas.

7- Conclusiones y propuestas de mejoras

OSSEC se consolida como herramienta de seguridad poderosa y versátil en detección y respuesta ante amenazas a nivel de host. Su arquitectura ligera y de código abierto permite adaptación a entornos con recursos limitados, proporcionando funcionalidades clave como monitoreo de integridad de archivos, detección de accesos no autorizados y respuesta activa eficaz. Es opción confiable para quienes buscan solución efectiva y adaptable sin complejidades de herramientas más pesadas.

Para potenciar OSSEC manteniendo simplicidad, podrían explorarse mejoras: interfaz de visualización básica opcional, módulos específicos para verificar

configuraciones según normativas comunes, y optimización de integración con sistemas externos de análisis para interactuar con plataformas de monitoreo centralizado sin comprometer su estructura ligera.

Sin embargo, OSSEC presenta limitaciones contra ataques distribuidos de botnet debido a su arquitectura basada en detección por IP individual. Carece de análisis comportamental avanzado, threat intelligence actualizada y correlación de patrones distribuidos complejos. Para protección efectiva contra amenazas distribuidas, debe complementarse con SIEM avanzados con machine learning, servicios cloud con behavioral analytics, o plataformas especializadas que identifiquen patrones operando bajo límites de detección tradicionales (fig 22).

8- Referencias

1- OSSEC Documentation. (2024). *OSSEC User Manual and Documentation*. Retrieved from https://www.ossec.net/documentation
2- Bejtlich, R. (2013). *The Practice of Network Security Monitoring: Understanding Incident Detection and Response*. No Starch Press.
3- Rash, M. (2017). *Linux Firewalls: Enhancing Security with nftables and Beyond*. No Starch Press.
4- Skoudis, E., & Liston, T. (2003). *Counter Hack Reloaded: A Step-by-Step Guide to Computer Attacks and Effective Defenses*. Prentice Hall.
5- Open Source Security, Inc. (2023). *OSSEC Documentation*.
6- Baeza-Yates, R., & Figueiredo, D. (2021). *Cybersecurity Threats, Malware, Vulnerabilities and their Defense Mechanisms: Practical Approaches to Improve Security Posture*.
7- Popescu, I., & Panait, A. (2020). "Evaluating Open-Source Host-Based Intrusion Detection Systems for Operational Security." *Journal of Cybersecurity and Privacy*.
8- Wazuh, Inc. (2023). *Wazuh Documentation: OSSEC-Based Architecture*.
9- "Ataques de fuerza bruta en 2025: el error más simple que aún destruye empresas" - Minery Report (Junio 2025)
10- "Un ataque masivo de fuerza bruta compromete millones de dispositivos VPN en todo el mundo" - Seguridad Informática / Infobae (Febrero 2025)
11- “Get our 2025 Hive Systems. Password Table” – Hive System (2025)

9- Lista de figura

Figura 1: instalador de OSSEC

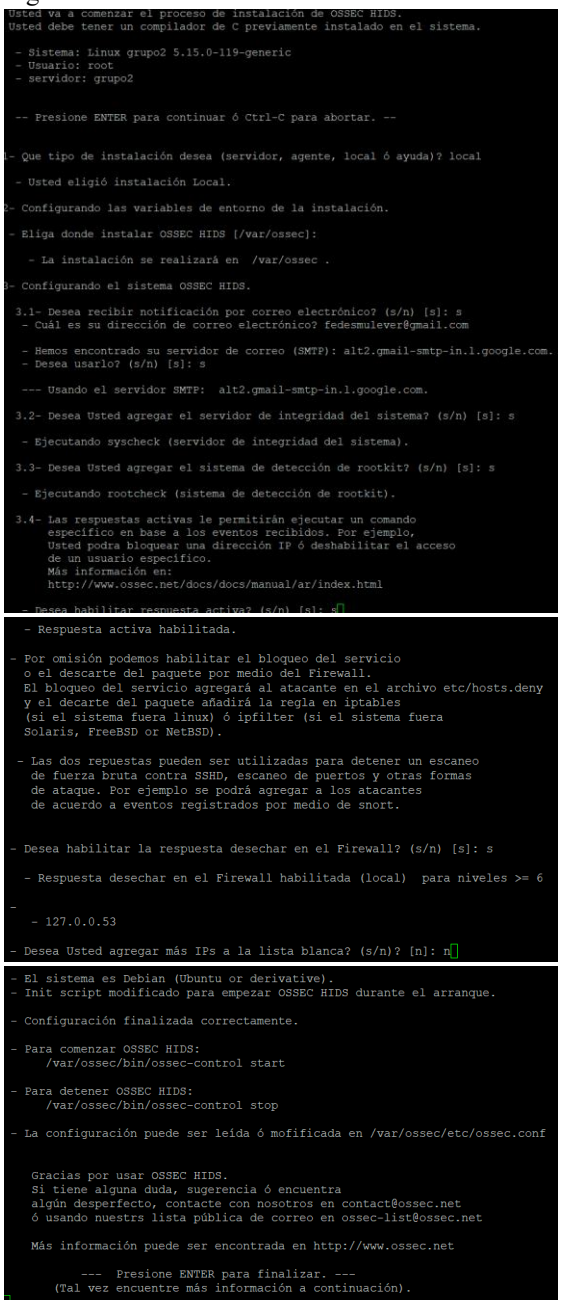

Figura 2: desactivar las reglas del firework

Figura 3: servicio SSH ejecutándose en ambas mv.

Figura 4: conectividad bidireccional

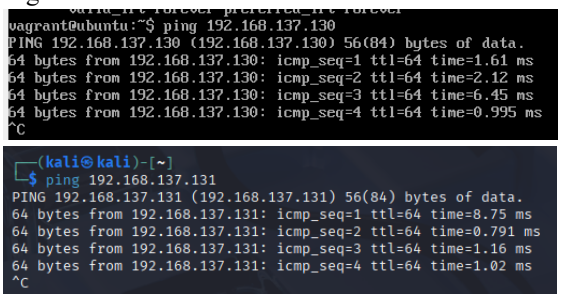

Figura 5: reporte de nmap

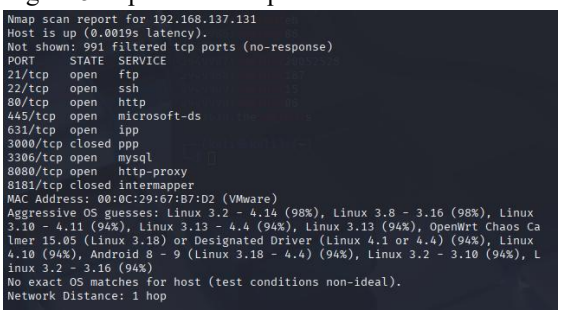

Figura 6: analizamos los diccionarios

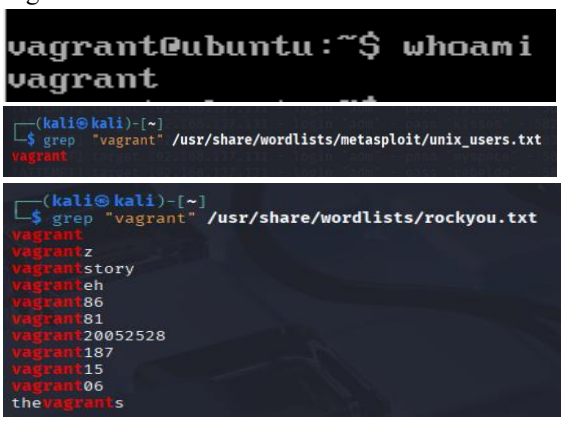

Figura 7: lanzamiento del ataque al servidor Linux

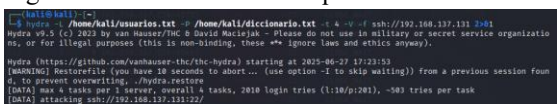

Figura 8: Hydra prueba diferentes combinaciones

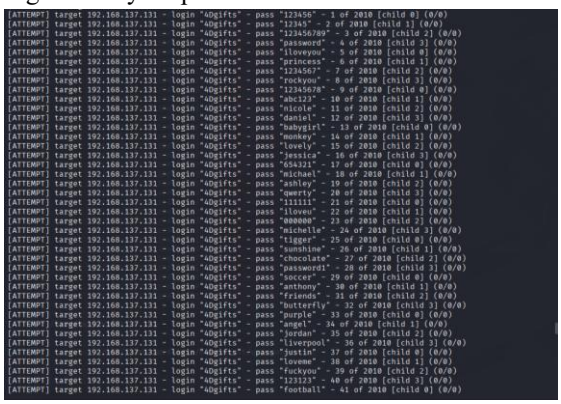

Figura 9: auth.log del servidor Linux:

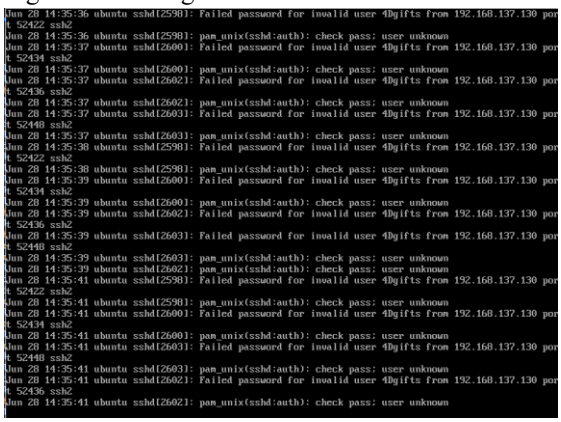

Figura 10: análisis de tráfico con Wireshark

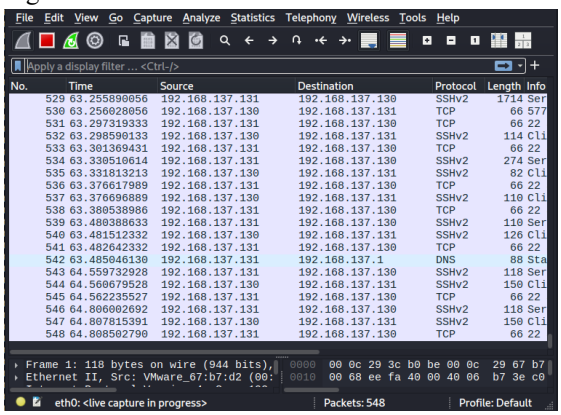

Figura 11: concreción exitosa del ataque

Figura 12: probamos las credenciales

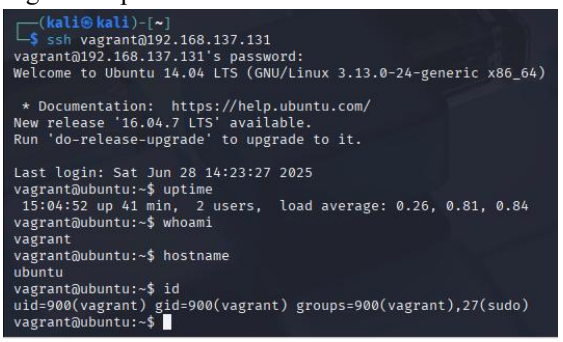

Figura 13: ponemos en funcionamiento OSSEC

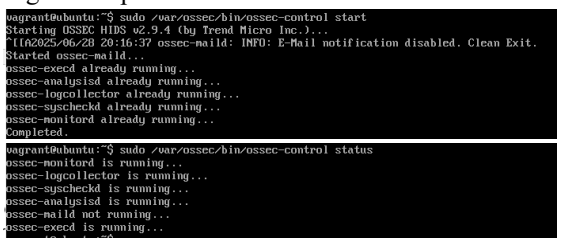

Figura 14: respuestas activas de OSSEC

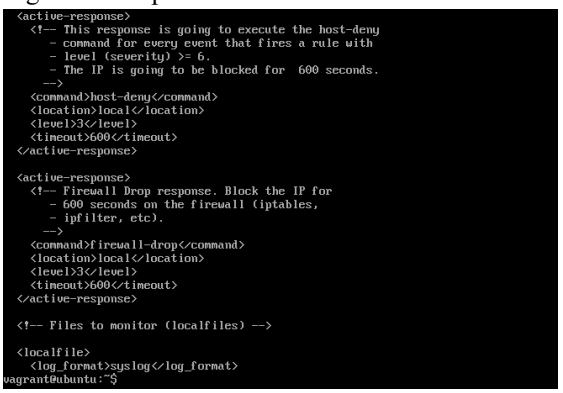

Figura 15: Regla 5716

Figura 16: Regla 5720

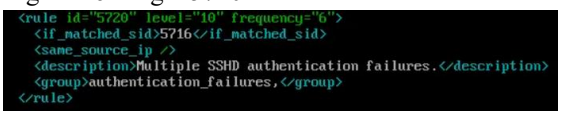

Figura 17: nuevo ataque con Hydra

Figura 18: resultados del ataque

Figura 19: alertas generadas por OSSEC

```
** Alert 1751165212.45460: - syslog,sshd,authentication_failed,
2025 Jun 29 02:46:52 ubuntu->/var/log/auth.log
Rule: 5716 (level 5) -> 'SSHD authentication failed.'
Src IP: 192.168.137.130
User: vagrant
Jun 29 02:46:52 ubuntu sshd[34088]: Failed password for vagrant from 192.168.137.130 port 56500 ssh2

** Alert 1751165212.45764: - syslog,sshd,authentication_failed,
2025 Jun 29 02:46:52 ubuntu->/var/log/auth.log
Rule: 5716 (level 5) -> 'SSHD authentication failed.'
Src IP: 192.168.137.130
User: vagrant
Jun 29 02:46:52 ubuntu sshd[34090]: Failed password for vagrant from 192.168.137.130 port 56512 ssh2

** Alert 1751165212.46068: - syslog,sshd,authentication_failed,
2025 Jun 29 02:46:52 ubuntu->/var/log/auth.log
Rule: 5716 (level 5) -> 'SSHD authentication failed.'
Src IP: 192.168.137.130
User: vagrant
Jun 29 02:46:52 ubuntu sshd[34087]: Failed password for vagrant from 192.168.137.130 port 56486 ssh2

** Alert 1751165212.46372: - syslog,sshd,authentication_failed,
2025 Jun 29 02:46:52 ubuntu->/var/log/auth.log
Rule: 5716 (level 5) -> 'SSHD authentication failed.'
Src IP: 192.168.137.130
User: vagrant
Jun 29 02:46:52 ubuntu sshd[34091]: Failed password for vagrant from 192.168.137.130 port 56502 ssh2
```

Figura 20: registros del sistema

```
vagrant@ubuntu:~$ sudo tail -f /var/log/auth.log
Jun 29 02:46:50 ubuntu sshd[3490]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=
0 tty=ssh ruser= rhost=192.168.137.130  user=vagrant
Jun 29 02:46:50 ubuntu sshd[34089]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=
0 tty=ssh ruser= rhost=192.168.137.130  user=vagrant
Jun 29 02:46:50 ubuntu sshd[34071]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=
0 tty=ssh ruser= rhost=192.168.137.130  user=vagrant
Jun 29 02:46:52 ubuntu sshd[34088]: Failed password for vagrant from 192.168.137.130 port 56500 ssh2
Jun 29 02:46:52 ubuntu sshd[34090]: Failed password for vagrant from 192.168.137.130 port 56512 ssh2
Jun 29 02:46:52 ubuntu sshd[34071]: Failed password for vagrant from 192.168.137.130 port 56486 ssh2
Jun 29 02:46:52 ubuntu sshd[34091]: Failed password for vagrant from 192.168.137.130 port 56502 ssh2
Jun 29 02:51:53 ubuntu sudo: pam_unix(sudo:session): session closed for user root
Jun 29 02:52:11 ubuntu sudo:  vagrant : TTY=tty1 ; PWD=/home/vagrant ; USER=root ; COMMAND=/usr/bin/
tail -f /var/log/auth.log
Jun 29 02:52:11 ubuntu sudo: pam_unix(sudo:session): session opened for user root by vagrant(uid=0)
```

Figura 21: bloqueo de la ip atacante

```
Sun Jun 29 00:33:31 UTC 2025 /var/ossec/active-response/bin/host-deny.sh add - 192.168.137.130 17511
57211.19520 5503
Sun Jun 29 00:33:32 UTC 2025 /var/ossec/active-response/bin/firewall-drop.sh add - 192.168.137.130 1
751157211.19520 5503
vagrant@ubuntu:~$
```

Figura 22: pérdida de tráfico

```
┌──(kali㉿kali)-[~]
└─$ ping 192.168.137.131
PING 192.168.137.131 (192.168.137.131) 56(84) bytes of data.
^C
── 192.168.137.131 ping statistics ──
40 packets transmitted, 0 received, 100% packet loss, time 39927ms
```

Con Wireshark:

No.	Time	Source	Destination	Protocol	Ler
1	0.000000000	192.168.137.130	192.168.137.131	ICMP	98
10	1.016381985	192.168.137.130	192.168.137.131	ICMP	98
11	2.040263075	192.168.137.130	192.168.137.131	ICMP	98
12	3.064321201	192.168.137.130	192.168.137.131	ICMP	98
13	4.089327297	192.168.137.130	192.168.137.131	ICMP	98
22	5.112242634	192.168.137.130	192.168.137.131	ICMP	98
25	6.136167570	192.168.137.130	192.168.137.131	ICMP	98
26	7.160570274	192.168.137.130	192.168.137.131	ICMP	98
27	8.184541236	192.168.137.130	192.168.137.131	ICMP	98
28	9.208345838	192.168.137.130	192.168.137.131	ICMP	98
29	10.232631071	192.168.137.130	192.168.137.131	ICMP	98
30	11.256573125	192.168.137.130	192.168.137.131	ICMP	98
31	12.283105587	192.168.137.130	192.168.137.131	ICMP	98
32	13.304634574	192.168.137.130	192.168.137.131	ICMP	98
33	14.328804166	192.168.137.130	192.168.137.131	ICMP	98
34	15.352362766	192.168.137.130	192.168.137.131	ICMP	98
35	16.376312272	192.168.137.130	192.168.137.131	ICMP	98
36	17.400262283	192.168.137.130	192.168.137.131	ICMP	98
37	18.424463994	192.168.137.130	192.168.137.131	ICMP	98
38	19.448251612	192.168.137.130	192.168.137.131	ICMP	98
39	20.474814118	192.168.137.130	192.168.137.131	ICMP	98

Info
Echo (ping) request id=0x000c, seq=1/256, ttl=64 (no respons…
Echo (ping) request id=0x000c, seq=2/512, ttl=64 (no respons…
Echo (ping) request id=0x000c, seq=3/768, ttl=64 (no respons…
Echo (ping) request id=0x000c, seq=4/1024, ttl=64 (no respon…
Echo (ping) request id=0x000c, seq=5/1280, ttl=64 (no respon…
Echo (ping) request id=0x000c, seq=6/1536, ttl=64 (no respon…
Echo (ping) request id=0x000c, seq=7/1792, ttl=64 (no respon…
Echo (ping) request id=0x000c, seq=8/2048, ttl=64 (no respon…
Echo (ping) request id=0x000c, seq=9/2304, ttl=64 (no respon…
Echo (ping) request id=0x000c, seq=10/2560, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=11/2816, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=12/3072, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=13/3328, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=14/3584, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=15/3840, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=16/4096, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=17/4352, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=18/4608, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=19/4864, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=20/5120, ttl=64 (no respo…
Echo (ping) request id=0x000c, seq=21/5376, ttl=64 (no respo…

Figura 23: ataques distribuidos geográficamente

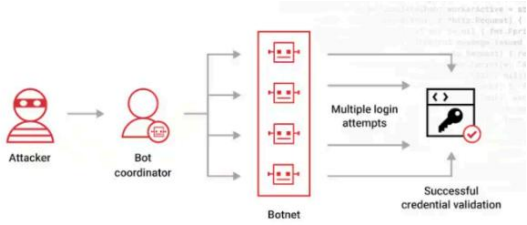

Figura 24: eficacia de los ataques

Hardware: 12 x RTX 5090 | Password hash: bcrypt (10)

Number of Characters	Numbers Only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instantly	Instantly	Instantly	Instantly	Instantly
5	Instantly	Instantly	57 minutes	2 hours	4 hours
6	Instantly	46 minutes	2 days	6 days	2 weeks
7	Instantly	20 hours	4 months	1 year	2 years
8	Instantly	3 weeks	15 years	62 years	164 years
9	2 hours	2 years	791 years	3k years	11k years
10	1 day	40 years	41k years	238k years	803k years
11	1 weeks	1k years	2m years	14m years	56m years
12	3 months	27k years	111m years	917m years	3bn years
13	3 years	705k years	5bn years	56bn years	275bn years
14	28 years	18m years	300bn years	3tn years	19tn years
15	284 years	477m years	15tn years	218tn years	1qd years
16	2k years	12bn years	812tn years	13qd years	94qd years
17	28k years	322bn years	42qd years	840qd years	6qn years
18	284k years	8tn years	2qn years	52qn years	463qn years