



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

– **TELECOM** ESCUELA
TÉCNICA **VLC** SUPERIOR
DE INGENIERÍA DE
TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de
Telecomunicación

Implementación y Evaluación de Sistemas IDS/IPS para
mejorar la Ciberseguridad en Entornos Industriales

Trabajo Fin de Grado

Grado en Ingeniería de Tecnologías y Servicios de
Telecomunicación

AUTOR/A: Corbella Alcántara, Valentino

Tutor/a: Sempere Paya, Víctor Miguel

Cotutor/a externo: Cano Sáez, José Ramón

CURSO ACADÉMICO: 2023/2024



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

– **TELECOM** ESCUELA
TÉCNICA **VLC** SUPERIOR
DE INGENIERÍA DE
TELECOMUNICACIÓN

NO PONER PORTADA

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

VLC/
CAMPUS
VALENCIA, INTERNATIONAL
CAMPUS OF EXCELLENCE



Resumen

Este trabajo, centrado en la mejora de la ciberseguridad en entornos industriales, aborda la implementación de sistemas NIDS/NIPS (Sistemas de Detección y Prevención de Intrusiones en Redes). En primer lugar, se analiza la necesidad del mercado de herramientas de ciberseguridad que permitan prevenir y alertar sobre posibles ataques. A continuación, se evalúa el rendimiento de las soluciones más utilizadas en el mercado, utilizando tanto criterios objetivos como subjetivos, para seleccionar la opción más adecuada. Posteriormente, se explora la integración con la pila de herramientas de ElasticSearch, con el fin de facilitar la visualización y el filtrado de datos. El principal objetivo es crear un entorno de red industrial simulado para evaluar la eficacia de estas herramientas ante diversos tipos de ataques. Los resultados obtenidos permitirán proponer mejoras y realizar ajustes en la implementación, así como evaluar la conveniencia de desplegar este tipo de soluciones en un entorno industrial real.

Palabras clave: ciberseguridad, ciberataques, red industrial, NIDS/NIPS, ElasticSearch, detección de intrusiones, prevención de intrusiones, Suricata, reglas, DoS, fuerza bruta, escaneo.

Resum

Aquest treball, centrat en la millora de la ciberseguretat en entorns industrials, aborda la implementació de sistemes NIDS/NIPS. En primer lloc, s'analitzarà la necessitat del mercat d'eines de ciberseguretat que prevenen i alerten dels possibles atacs. A continuació, s'avaluarà el rendiment de les opcions més populars del mercat mitjançant criteris objectius i subjectius, i es seleccionarà la millor opció. Posteriorment, s'explorarà la integració amb el *stack* d'ElasticSearch per tal de poder visualitzar i filtrar les dades amb més facilitat. L'objectiu és crear un entorn de xarxa industrial amb un switch i dos PLCs per avaluar l'eficàcia de les eines davant de diversos tipus de atacs. Les conclusions resultants guiaran les millores i possibles ajustaments en la implementació proposada, així com l'avaluació sobre la necessitat d'una implementació a aquest nivell.

Paraules clau: ciberseguretat, ciberatacs, xarxa industrial, NIDS/NIPS, ElasticSearch, detecció d'intrusions, prevenció d'intrusions, Suricata, regles, DoS, força bruta, escaneig.

Abstract

This work it is focused on improving cybersecurity in industrial environments, addresses the implementation of NIDS/NIPS systems. The market need for cybersecurity tools that prevent and alert on attacks will be analyzed. Next, the performance of the most popular market options will be evaluated using both objective and subjective criteria, and the best option will be selected. Subsequently, integration with the ElasticSearch stack will be explored to enable easier data visualization and filtering. The goal is to create an industrial network environment with a switch and two PLCs to evaluate the effectiveness of the tools against various attacks. The resulting conclusions will guide improvements and possible adjustments in the proposed implementation, as well as the assessment of the need for an implementation at this level.

Keywords: cybersecurity, cyberattacks, industrial network, NIDS/NIPS, ElasticSearch, intrusion detection, intrusion prevention, Suricata, rules, DoS, brute force, scanning.

RESUMEN EJECUTIVO

La memoria del TFG del Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación debe desarrollar en el texto los siguientes conceptos, debidamente justificados y discutidos, centrados en el ámbito de la ingeniería de telecomunicación

CONCEPT (ABET)	CONCEPTO (traducción)	¿Cumple? (S/N)	¿Dónde? (páginas)
1. IDENTIFY:	1. IDENTIFICAR:	S	1-17
1.1. Problem statement and opportunity	1.1. Planteamiento del problema y oportunidad	S	1-3
1.2. Constraints (standards, codes, needs, requirements & specifications)	1.2. Toma en consideración de los condicionantes (normas técnicas y regulación, necesidades, requisitos y especificaciones)	S	4-15
1.3. Setting of goals	1.3. Establecimiento de objetivos	S	16-17
2. FORMULATE:	2. FORMULAR:	S	17-40
2.1. Creative solution generation (analysis)	2.1. Generación de soluciones creativas (análisis)	S	17-22
2.2. Evaluation of multiple solutions and decision-making (synthesis)	2.2. Evaluación de múltiples soluciones y toma de decisiones (síntesis)	S	23-40
3. SOLVE:	3. RESOLVER:	S	41-42
3.1. Fulfilment of goals	3.1. Evaluación del cumplimiento de objetivos	S	41
3.2. Overall impact and significance (contributions and practical recommendations)	3.2. Evaluación del impacto global y alcance (contribuciones y recomendaciones prácticas)	S	42



Índice

Capítulo 1.	Introducción. Necesidad de mercado y definición de las soluciones	1
1.1	Contexto actual	1
1.2	Soluciones de defensa.....	2
Capítulo 2.	Sistemas IDS/IPS	4
2.1	Objetivo del IDS	4
2.2	Tipos de sistemas IDS	4
2.3	Sistemas de Detección.....	5
2.4	Tácticas de evasión de IDS [5][15]	6
2.5	Métodos de prevención del IPS [6].....	6
2.6	Estructura de un NIDS/NIPS [17].....	7
2.7	Ejemplo clásico de una topología de red industrial completa [14]	8
Capítulo 3.	Evaluación y elección del software NIDS/NIPS	10
3.1	Softwares a evaluar	10
3.2	Puntos no técnicos a evaluar	10
3.2.1	Financiación, sostenibilidad, reputación y credibilidad.....	10
3.2.2	Estrategia de desarrollo y futuro.....	11
3.3	Puntos técnicos a evaluar.....	12
3.3.1	Capacidad de procesamiento [17]	12
3.3.2	Reglas, personalización y actualizaciones	13
3.3.3	Interfaz de monitorización	13
3.3.4	Adaptabilidad con protocolos y tecnologías emergentes.....	14
3.4	Conclusión y elección del software IDS/IPS.....	15
Capítulo 4.	Planteamiento de la red y objetivos.....	16
4.1	Condicionantes	16
4.2	Configuración de los equipos	17
Capítulo 5.	Aplicación de Suricata en la red industrial.....	20
5.1	Reglas en Suricata [44].....	20
5.1.1	La acción	20
5.1.2	La cabecera.....	20
5.1.3	Las opciones	21



Capítulo 6.	Ataques en la red industrial.....	23
6.1	Sondeo de puertos con la herramienta Nmap.....	23
6.1.1	Sondeo de descubrimiento de hosts en la red con protocolo TCP. [48].....	24
6.1.2	Sondeo de descubrimiento de hosts en la red con protocolo UDP. [48][49]	27
6.1.3	Sondeo de puertos sobre un host con técnicas de evasión IDS. [48]	30
6.2	Ataque DoS.....	33
6.2.1	Resultados del ataque DoS.....	34
6.2.2	Cómo proteger a la subred de un ataque DoS. [44].....	34
6.3	Ataque del tipo ICMP Redirect (Man In The Middle).	35
6.3.1	Resultado del ataque ICMP Redirect	36
6.3.2	Cómo proteger a la subred de un ataque ICMP Redirect [44].....	37
6.4	Ataque de fuerza bruta con diccionario sobre la conexión SSH / Telnet	38
6.4.1	Resultados del ataque de fuerza bruta con diccionario.....	38
6.4.2	Cómo proteger al switch industrial de un ataque de fuerza bruta por diccionario. [44]	39
Capítulo 7.	Conclusiones de la implementación de un IDPS en un entorno de red industrial 41	
7.1	Futuras líneas de acción.....	42
Capítulo 8.	Bibliografía	43
Capítulo 9.	Anexo de instalación de los softwares.....	47
9.1	Instalación y configuración de Suricata en Ubuntu 20.04.	47
9.2	Instalación y configuración de ElasticSearch y Kibana.....	49
9.3	Configuración del Stack Elastic para Suricata mediante los Elastic Agents y Fleet Servers. 52	



Índice de figuras

Figura 1. Diferencia porcentual de ciberataques en 2022 y 2021 por región. [1].....	1
Figura 2. Cambios en el primer semestre del 2023 en porcentajes sobre las acciones maliciosas bloqueadas, por región. [2].....	2
Figura 3. Topología básica de una red con la implementación de un NIDS y de varios HIDS. [18]	5
Figura 4. Estructura básica de un NIDS/NIPS [17].....	7
Figura 5. Topología de una red industrial con sistemas IDS/IPS y SIEM incorporados. [14]	8
Figura 6. Topología de la red industrial planteada en este TFG.....	17
Figura 7. Resultado del escaneo Nmap -A desde la máquina atacante dirigido a la red 192.168.1.0/24	24
Figura 8. Resultado del escaneo Nmap -A desde la máquina atacante dirigido a la red 192.168.1.0/24	25
Figura 9. Captura de Wireshark desde el Ubuntu del escaneo Nmap -A dirigido a la red 192.168.1.0/24	25
Figura 10. Captura de Kibana del escaneo Nmap -A dirigido a la red 192.168.1.0/24.....	27
Figura 11. Captura del resultado del Nmap -A dirigido a la red 192.168.1.0/24 tras incorporar las reglas de Suricata.....	27
Figura 12. Resultado del escaneo Nmap -sU desde la máquina atacante dirigido a la red 192.168.1.0/24	28
Figura 13. Captura Wireshark desde el Ubuntu del escaneo Nmap -sU dirigido a la red 192.168.1.0/24	29
Figura 14. Captura de Kibana del escaneo Nmap -sU dirigido a la red 192.168.1.0/24.....	30
Figura 15. Resultado del escaneo Nmap -sU desde la máquina atacante dirigido a la red 192.168.1.0/24	31
Figura 16. Captura de Wireshark del escaneo de puertos Nmap -sS -D RND:10 -f -T2 -p 1-1024 192.168.1.21	31
Figura 17. Captura de Kibana del escaneo de puertos Nmap -sS -D RND:10 -f -T2 -p 1-1024 192.168.1.21	32
Figura 18. Captura de Wireshark del escaneo de puertos Nmap -sS -D RND:10 -f -T2 -p 1-1024 192.168.1.21	33
Figura 19. Captura de Wireshark del ataque hping3 --flood -S -p 102 192.168.1.21	34
Figura 20. Captura del resultado de los pings mientras sucede el ataque DoS.....	34
Figura 21. Captura de Kibana del ataque DoS	35
Figura 22. Diagrama de funcionamiento del ataque ICMP Redirect [53]	36
Figura 23. Captura de Wireshark del ataque ICMP Redirect	37
Figura 24. Captura de Kibana tras el ataque ICMP Redirect	38
Figura 25. Resultado del ataque de fuerza bruta desde la máquina atacante.....	39



Figura 26. Resultado del ataque de fuerza bruta desde la máquina atacante tras la activación de las reglas de Suricata.....	39
Figura 27. Captura de Kibana tras el ataque de fuerza bruta.....	40
Figura 28. Resultado del comando whereis suricata	47
Figura 29. Resultado del comando ip addr.....	47
Figura 30. Captura del archivo suricata.yaml en la sección de interfaz	48
Figura 31. Resultado del comando suricata-update.....	48
Figura 32. Captura del archivo suricata.yaml en la sección de reglas.....	49
Figura 33. Resultado del comando service suricata status.....	49
Figura 34. Captura del archivo elasticsearch.yml.....	50
Figura 35. Resultado del comando elasticsearch-setup-passwords interactive	51
Figura 36. Captura del archivo kibana.yml.....	51
Figura 37. Captura de la integración Suricata dentro de la interfaz web de Kibana	52
Figura 38. Captura de la configuración de Suricata como Fleet Agent desde Kibana	53
Figura 39. Captura de los pasos a seguir para instalar el Fleet Server de Suricata.....	53
Figura 40. Captura del funcionamiento del Suricata Fleet Server desde Kibana	54

Índice de tablas

Tabla 1. Comparación de capacidad de procesamiento entre Snort y Suricata en un entorno estándar [17].....	12
Tabla 2. Comparación de capacidad de procesamiento entre Snort y Suricata variando el ancho de banda en modo IDS [17].....	12
Tabla 3. Comparación de capacidad de procesamiento entre Snort y Suricata variando el ancho de banda en modo IPS [17]	13
Tabla 4. Comparación de los tipos de paquetes y protocolos soportados por Snort y Suricata. [32]	14
Tabla 5. Puntuaciones obtenidas por Snort y Suricata en los puntos abarcados en las secciones 3.2 y 3.3.....	15

Capítulo 1. Introducción. Necesidad de mercado y definición de las soluciones

1.1 Contexto actual

Se ha evidenciado que el mundo está experimentando una creciente digitalización. No solamente en el contexto de más o mejores teléfonos u ordenadores sino en la incursión en la vida diaria que han supuesto los aparatos IoT y el avance tecnológico en distintos gadgets como relojes o pulseras.

Concretamente, en el sector industrial, ha sido bien recibido el uso de los avances tecnológicos en software, IoT y ciencia de datos, entre otros. Por ejemplo, una mayor monitorización y recolecta de datos repercute en poder hacer análisis predictivos tanto de mantenimiento como de detección precoz de defectos. Por otra parte, ha habido mejoría en la organización gracias a los softwares ERP (Enterprise Resource Planning), softwares de planificación de recursos empresariales.

Sin embargo, con la introducción de estas tecnologías también se abre la veda a posibles nuevos ataques. Al exponer las redes de control a internet, repercute en una mayor cantidad de puntos potencialmente vulnerables.

Un ejemplo serían los ransomwares, ataques basados en secuestrar los datos de la víctima impidiéndole el acceso a ellos y pidiendo un pago económico por devolverlos.

Y otro ejemplo conocido es el ataque de denegación de servicio, con el objetivo de sobrecargar los sistemas causando el fallo en su funcionamiento.

Esta situación hace que la filtración de datos, que ya no se limita únicamente a información de carácter personal, sino también a datos empresariales y de producción, pueda suponer problemas graves para una industria. Sin mencionar los perjuicios aún peores que podría implicar que un atacante tuviera acceso remoto al control de los dispositivos conectados a Internet.

Según indica Check Point Research, en 2022, la media semanal de ataques cibernéticos aumentaron un 38% en media, suponiendo un aumento del 26% en Europa y un 52% en Norteamérica. [1]

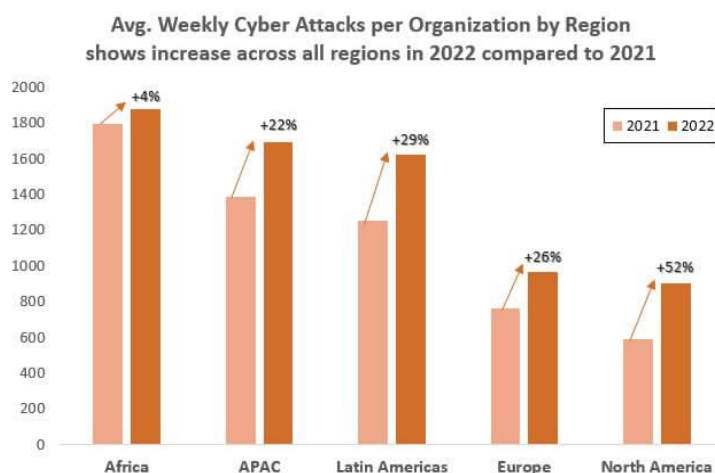


Figura 1. Diferencia porcentual de ciberataques en 2022 y 2021 por región. [1]

También, Kaspersky en su último reporte indica que hubo un incremento del 4.6% de ciberataques bloqueados en los ordenadores industriales en Europa Occidental. [2]

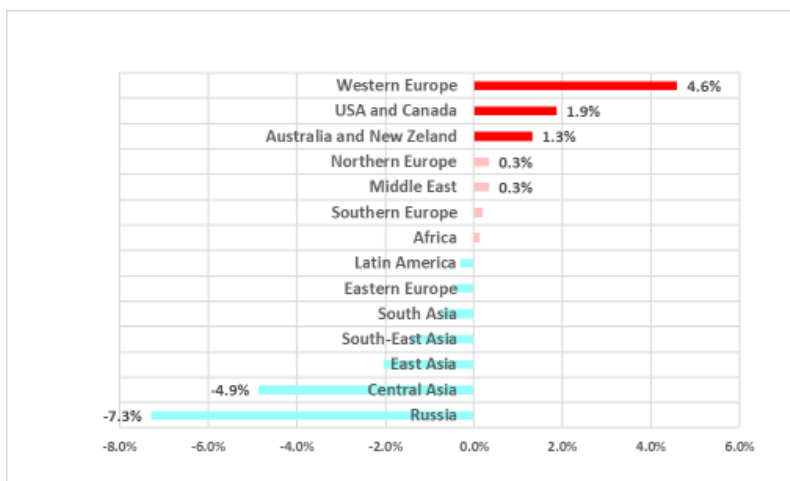


Figura 2. Cambios en el primer semestre del 2023 en porcentajes sobre las acciones maliciosas bloqueadas, por región. [2]

Sin contar que el gran último avance tecnológico, la inteligencia artificial, ha sido uno de los motivos por los que en 2023 ha habido un incremento significativo de los ataques.

Los ciberdelincuentes han adoptado un enfoque más sofisticado, utilizando la IA para desarrollar y ejecutar ataques más precisos y difíciles de detectar. Por ejemplo, han comenzado a utilizar IA para crear anuncios, correos electrónicos y sitios web falsos que imitan a canales y plataformas legítimas de instituciones financieras, lo que dificulta la distinción entre contenido auténtico y fraudulento.

La IA está lo suficientemente desarrollada como para llevar a cabo fraudes con interacciones directas. Tal como en El Confidencial reportan, unos ciberdelincuentes consiguieron defraudar más de 567.000€ suplantando la identidad de un íntimo amigo del CEO de la empresa a través de una videollamada, esto quiere decir suplantando imagen y voz. [3]

1.2 Soluciones de defensa

Debido a la creciente amenaza de ataques cibernéticos en entornos industriales, la implementación de redes OT (Operational Technology) seguras se ha vuelto fundamental.

Históricamente se ha estandarizado el uso de air gaps, técnica que consiste en aislar físicamente los sistemas críticos de cualquier otra red, incluyendo Internet, para evitar accesos no autorizados. [4]

Junto a los air gaps también se ha usado frecuentemente los diodos de datos. Dispositivos electrónicos unidireccionales que permiten el flujo de información en una sola dirección, de la red OT a la IT, previniendo así cualquier posibilidad de un ataque desde el exterior a la red OT. [4]

Pese a ser dispositivos que llevan a cabo su función correctamente y son infranqueables debido a que conllevan una barrera física a superar, se han ido sustituyendo por Firewalls en los casos más convenientes. Esta decisión es tomada porque tener una seguridad tan alta como la de un air gap o diodo de datos supone además que cuando la empresa necesita transferir datos desde la red o dispositivo seguro a otro lugar, también debe pasar todas estas barreras, haciendo más ineficiente su transferencia.



En la evolución tecnológica se ha visto como una combinación de Firewall sumado a un IDS/IPS o SIEM, forman una barrera robusta contra las amenazas cibernéticas, salvaguardando la integridad y continuidad operacional de la industria y haciendo más eficiente la transferencia voluntaria de datos.

Es en los sistemas IDS/IPS en los que estará centrado el trabajo de fin de grado.

Un IDS (Intrusion Detection System) se trata de una herramienta de seguridad que monitoriza la red para alertar de tráfico malicioso, actividad sospechosa y violaciones de la política de seguridad. [5]

Un IPS (Intrusion Prevention System) es una herramienta de seguridad muy similar al IDS ya que también monitoriza la red en busca de potenciales amenazas pero además de alertar también puede estar programado para tomar acción de forma automática ante estas, eliminando el tráfico de la red que active las condiciones configuradas. [6]

Ha de entenderse que un IDS/IPS en sí mismo es un sistema de seguridad, por tanto depende también del contexto donde se decida utilizar. Es relevante comprender que se puede establecer un IDS/IPS en una red o en un host. Serían conocidos como NIDS/NIPS e HIDS/HIPS respectivamente.

Además, también es posible usar un software adicional con el que gestionar las reglas de prevención y dónde poder visualizar los eventos notificados por el IDS. Este software se trata del SIEM (Security Information & System Event Management).

El SIEM se trata de una solución híbrida centralizada que maneja la gestión de la información de seguridad y la gestión de eventos. El objetivo es proporcionar un análisis en tiempo real de las alertas de seguridad de los dispositivos dónde se hayan implantado sistemas de detección de intrusión. Recopila información sobre registros de actividad de los sistemas para buscar actividades sospechosas. [7]

Por tanto, se puede definir que el SIEM es la herramienta que unifica las posibilidades de un IPS con una gestión centralizada y detallada de la actividad de la red.

A pesar de ser una herramienta muy avanzada, su uso puede ser desaconsejado en entornos que no requieran un gran nivel de seguridad por el hecho de que su instalación, mantenimiento y personal para utilizarla tiene un alto coste debido a la complejidad que requiere.

En el siguiente capítulo, se hará una descripción más precisa de las tareas que desempeñan cada uno de los conceptos definidos y su funcionamiento.

Capítulo 2. Sistemas IDS/IPS

En este punto, se procederá a definir para qué sirve el IDS de forma más detallada, los tipos de sistemas IDS, cómo funcionan, qué técnicas usan los atacantes para evadirlos y por último los métodos de prevención del IPS.

2.1 Objetivo del IDS

El objetivo principal del IDS es poder hacer más eficiente la tarea de detectar las amenazas o ataques que puedan haber en una red, ya sea mejorando la velocidad de detección o automatizando ciertos procesos. [5]

El sistema IDS se puede complementar con otras herramientas, como el SIEM, funcionan en conjunto de forma que una vez se halla una amenaza se alerta mediante el SIEM para que el personal del SOC (Security Operations Center) tenga a mano la mayor cantidad posible de información para poder tomar una decisión sobre la red. [5]

Por otro lado, está el IPS. Que en este caso sería ampliar las posibilidades del software IDS para que también pueda ejecutar acciones de prevención automáticas ante determinadas alertas. [5]

Otro aliado del IDS sería el cortafuegos que en este caso serviría para complementar las reglas que pudiera tener y hacer una mejor gestión. Los NGFW (Next-Generation Firewall) tienen funciones IDS/IPS ya incorporadas. [5]

Es importante recalcar que el IDS únicamente detecta amenazas, no ejecuta ninguna acción sobre la red o host en el que esté alojado.

2.2 Tipos de sistemas IDS

- NIDS (Network Intrusion Detection System)

Los sistemas NIDS deben vigilar la totalidad tráfico de los dispositivos de la red a proteger para tomar las decisiones basándose en los metadatos y contenido del paquete. Se deben colocar en puntos estratégicos como seguidamente del cortafuegos para poder detectar lo antes posible cualquier tráfico malicioso que pueda haber. [5] [13]

Como se podría imaginar, si pusiéramos el NIDS como punto por donde deben pasar todos los paquetes podríamos empeorar la situación de la red o llegar a tener cuellos de botella. Es por eso que la red no se hace pasar directamente por el sistema sino que va analizando copias de los paquetes de red en lugar de los paquetes de red en sí mismos. [5]

- HIDS (Host Intrusion Detection System)[5]

Un sistema HIDS se instala directamente en un punto (host) específico como un ordenador o un servidor. Su tarea es supervisar la actividad del dispositivo tanto a nivel de cambios del sistema como del tráfico que pase por él.

Funciona haciendo instantáneas periódicas de los ficheros más importantes del equipo en busca de posibles cambios que hayan podido haber a causa de un ataque. En ese caso emite una alerta.

Es una opción de seguridad a tener en cuenta cuando tengamos sistemas que son críticos y se necesite tener una capa de seguridad adicional.

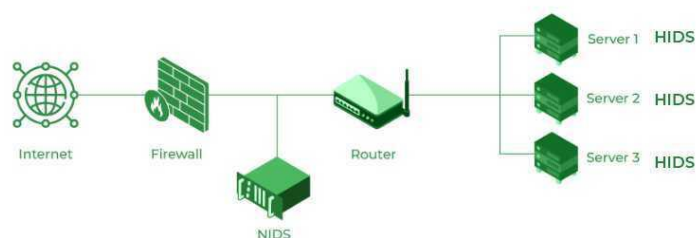


Figura 3. Topología básica de una red con la implementación de un NIDS y de varios HIDS. [18]

Como se puede observar en la Figura 3, el IDS se sitúa tras el firewall en paralelo al flujo de datos de la red ya que solo lee, si se tratara de un IPS se situaría dentro del propio flujo en serie para poder bloquear paquetes en el caso de que fuera necesario. También en los hosts de la red, en este caso servidores, vemos como se indica que tienen HIDS incorporados.

Otros sistemas IDS [5]

- PIDS: es un IDS basado en prototipos y su función es analizar el tráfico de los protocolos de conexión entre servidores y dispositivos.
Por ejemplo: un PIDS en un servidor web supervisando las conexiones HTTP. [5]
- APIDS: es un IDS basado en protocolos de aplicación que funciona a nivel de capa de aplicación analizando protocolos concretos de la aplicación.
Por ejemplo: un APIDS entre un servidor web y la base de datos SQL para detectar inyecciones SQL.

2.3 Sistemas de Detección

Generalmente los sistemas IDS presentan la capacidad de poder combinar los sistemas de detección, siendo híbridos. [14]

- Detección basada en firmas [13]

Básicamente se basa en escanear constantemente la red en busca de que en los paquetes que circulan coincida el comportamiento con la base de datos de firmas de ciberataques.

Las firmas se podrían definir como características o comportamientos que están asociados a una amenaza.

Para desempeñar bien esta función el IDS debe de tener una base de datos de firmas actualizada.

- Detección basada en anomalías [5][13][14]

Teniendo un modelo de referencia de lo que sería la actividad normal de una red, se analiza el comportamiento continuamente en búsqueda de anomalías.

Por ejemplificar podría ser que haya un proceso haciendo uso de más o menos ancho de banda de lo habitual o tener algún puerto abierto que en condiciones normales no debería estarlo.

Es con este método de detección que se podría llegar a asegurar a la red de ataques de día cero ya que al no estar aún parcheados y ser nuevos, la detección de firmas es probable que no lo detecte.

En contraposición, ante cualquier comportamiento anómalo por mera casualidad hace que este sistema de detección lo notifique. Esto nos lleva a que es más propenso a falsos positivos (alerta de amenaza que realmente no lo es).

- Detección basada en Machine Learning

Dotar a los softwares IDS/IPS y SIEM de algoritmos de ML incrementa considerablemente la capacidad que ya de por sí tenían.

Un ejemplo en el campo IDS es con un ML entrenado con una base de datos lo suficientemente buena (y grande) poder detectar firmas más rápidamente. [16]

En el caso de los IPS sería tener cierta capacidad de decisión sin tener que esperar una respuesta por parte del SOC. Ya de por sí los IPS la tienen, pero en este caso sería mejorar su capacidad disminuyendo las tasas de falsos negativos y falsos positivos.

En sistemas SIEM se trataría de mejorar las estadísticas mostradas y auxiliar en el análisis al personal que lo supervise.

Aún con estas ventajas, al introducir más tecnología en el sistema también se introducen más posibilidades de riesgo.

Es conveniente destacar dos métodos de detección menos comunes pero útiles:

- Detección basada en reputación: se trataría de detectar amenazas si el tráfico detectado proviene de un dominio/ip conocido por actividades maliciosas (comprobación de blacklist). [5]
- Análisis de protocolos con seguimiento de estado: observando el comportamiento del protocolo. Por ejemplo detectar que se han efectuado una gran cantidad de solicitudes para realizar una conexión TCP en un tiempo corto (DDOS). [5]

2.4 Tácticas de evasión de IDS [5][15]

En la actualidad hay una carrera entre hackers e IDS en las técnicas de evasión. Los hackers buscan la forma de evadir el IDS para realizar sus actividades y los sistemas IDS deben actualizarse para evitarlo.

Las tácticas más frecuentes son:

- Ataques DDoS: se trata de desbordar la red desde múltiples orígenes haciendo así que los IDS no puedan alertar correctamente de todas las amenazas.
- Suplantación: se trata de falsificar las direcciones IP y registros DNS para engañar al IDS haciendo ver que es una fuente fiable.
- Fragmentación: se fragmenta el malware en distintos paquetes más pequeños dificultando así la detección de la firma del malware.
- Cifrado: al utilizar protocolos de cifrado en la transferencia de paquetes el IDS no sabe qué contenido portan si no tiene la clave.

2.5 Métodos de prevención del IPS [6]

A continuación se muestran las acciones que típicamente toma un IPS cuando detecta una amenaza y evalúa que puede tomar decisión directamente sin pasar por el SOC.

- Bloquear tráfico malicioso

Es el método más básico y se trata de bloquear IPs o finalizar sesiones en caso de alerta. También podría bloquear el tráfico a un punto concreto.

Hay IPS que redirigen el tráfico a un *honeypot*, el cual es un sistema de seguridad informática diseñado para simular recursos de red aparentemente vulnerables con la finalidad de engañar al *hacker* haciéndole creer que ha tenido éxito en su ataque.

- Eliminar contenido malicioso

En lugar de bloquear todo el tráfico esta medida es más light y de lo que trata es de descartar únicamente los paquetes maliciosos del flujo pero sin cesar la transferencia del flujo. También se podría ejemplificar con la eliminación de un archivo adjunto de un correo electrónico

- Activar otros dispositivos de seguridad

Alerta a otros dispositivos a actuar, como por ejemplo al firewall o cambiar la configuración del router.

- Aplicación de políticas de seguridad

Bloqueo del tráfico que vaya a infringir las políticas de seguridad. Como el intento de difusión de datos sensibles de una empresa a través de internet. El IPS bloquearía la acción.

2.6 Estructura de un NIDS/NIPS [17]

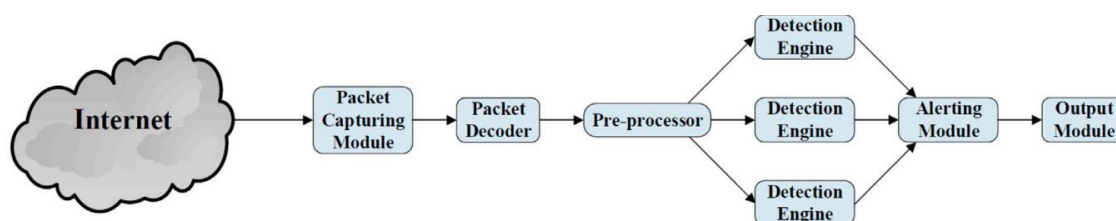


Figura 4. Estructura básica de un NIDS/NIPS [17]

1. Módulo de captura de paquetes.

El primer módulo de análisis tal como su nombre indica funciona para poder capturar los paquetes y poder analizarlos. Puede funcionar en modo pasivo para funciones exclusivamente IDS dónde el paquete se clona y se almacena en un buffer temporal para el análisis. El otro modo es denominado en línea y necesita una capacidad de procesamiento muy alta ya que se ejecuta el análisis y prevención sobre el mismo paquete que llega de internet por tanto debe ser en tiempo real.

2. Módulo decodificador de paquetes.

Aplica reglas específicas de protocolo para decodificar los paquetes y asegurar que mantiene unos estándares. Si un paquete no cumple los estándares correspondientes a su protocolo teniendo por ejemplo un tamaño distinto puede ser descartado.

3. Módulo preprocesador.

Se preprocesan paquetes de protocolos de alto y bajo nivel. A nivel bajo procesa paquetes para su fragmentación y reensamblaje. A nivel alto valida los estándares del paquete respecto a su protocolo de aplicación.

4. Motor de detección

Aquí se puede observar en la figura 3 como el flujo lineal se vuelve lineal-paralelo aumentando la cantidad de módulos en esta etapa. Esto se debe a que se trata de la etapa más importante y que requiere de más capacidades pues se aplican las reglas de detección de amenazas comparando los paquetes con las firmas, analizando la actividad del tráfico, etc.

5. Módulo de alerta

Se encarga de generar la alerta cuando se detecta una amenaza. En el modo IDS solo genera alertas, en el modo IPS también es capaz de bloquear el paquete además de generar la alerta.

6. Módulo de salida

Es el último módulo de la estructura NIDPS y básicamente genera estadísticas de los paquetes que la han atravesado.

2.7 Ejemplo clásico de una topología de red industrial completa [14]

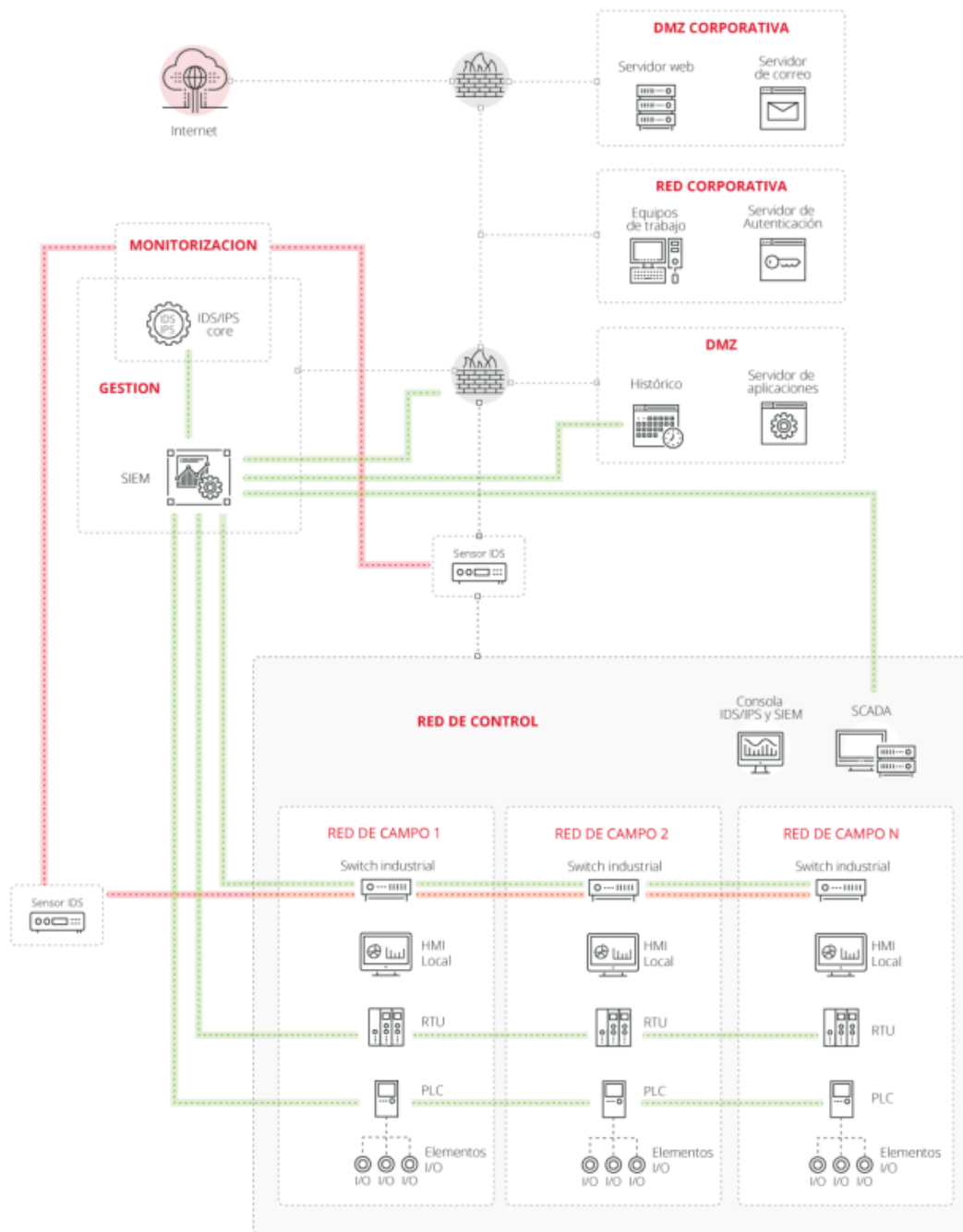


Figura 5. Topología de una red industrial con sistemas IDS/IPS y SIEM incorporados. [14]

A continuación, se procede a explicar la topología de la figura 5.

1. DMZ corporativa y red corporativa

Tras el primer cortafuegos en la DMZ se encuentran los servidores que pueden ser accesibles desde internet como el de correo o el web.

Además, también se encuentra la red corporativa dónde se encuentran los equipos de trabajos y que contiene un sistema de seguridad adicional con la gestión de la autenticación. Hay que tener en cuenta que estos hosts también pueden llegar a ser accesibles desde internet con un sistema de VPN correctamente implementado.

2. DMZ

Tras el segundo cortafuegos encontramos otra red DMZ esta vez con servidores más críticos y que deben estar mejor protegidos.

3. Red de control

Aquí tenemos, por una parte, la red SCADA (Supervisory Control and Data Acquisition) que son relativos a la gestión industrial para adquirir y controlar los datos de los procesos industriales con el fin de que los operadores puedan tomar las decisiones óptimas. [19]

También se referencia una consola IDPS/SIEM, pero no será en esta zona donde se haga la toma de decisiones.

Por otro lado, encontramos las redes de campo. Es la red más crítica para la planta pues es la encargada directa de los procesos industriales.

Dentro encontramos un switch para conectar los distintos dispositivos de la planta como por ejemplo sensores.

Los HMI Local, esta es la interfaz hombre-máquina que permite interactuar a los operadores con los sistemas de control, generalmente mediante pantallas. [20]

Los RTU's, que sirve para manejar los datos de elementos ubicados en zonas remotas. [21]

Los PLC's, pieza clave en cualquier industria, controlan los procesos y datos de los elementos I/O conectadas a estos. [21]

4. Red de Control y Monitorización.

Como se puede apreciar, hay sensores IDS en zonas críticas de la topología como es tras el segundo cortafuegos y también en cada red de campo monitoreando los datos que pasan por los switches hacia las redes de campo.

La información recolectada se muestra en la red de control y monitorización a través de las herramientas SIEM, las cuales sirven para valorar el comportamiento de la red.

También encontramos el IDS/IPS Core que sería el sistema central dónde se ejecuta el análisis automático de la red. Al ser un proceso costoso, dependiendo del tamaño de la red, puede ser rentable centralizar los datos a un servidor tanto por equipos como por el propio mantenimiento.

Capítulo 3. Evaluación y elección del software NIDS/NIPS

La finalidad de este capítulo es poder llegar a la conclusión sobre qué software utilizar en el trabajo final de grado enfocado en NIDS/NIPS.

Para ello, lo pensado es evaluar en distintos puntos tanto técnicos como no técnicos qué software es mejor opción que otro. En los aspectos técnicos nos atendemos a métricas totalmente objetivas y en los aspectos no-técnicos a métricas con sesgos subjetivos.

En cada punto se detallarán las capacidades de cada software y al finalizar el punto se evaluará qué software es superior en el campo evaluado.

Para ello se le podrá asignar 4 puntuaciones:

M = Mal – 1 Punto

R = Regular/normal – 2 Puntos

B = Bien – 3 Puntos

MB = Muy bien/excelente – 4 Puntos

Al finalizar con todos los bloques a evaluar se hará un repaso de las puntuaciones y se escogerá en base a ella.

3.1 Softwares a evaluar

Para poder evaluar un número viable de softwares hay que hacer un primer filtrado basados en la popularidad y búsquedas de Google de las herramientas más usadas. A causa de que no se obtienen resultados razonables evaluando un software con meses desde su creación que a pesar de que tenga capacidades para competir, al no estar estandarizado en el mercado puede ocasionar problemas aún no localizados y falta de soporte.

De esta forma, hay 4 competidores principales en softwares IDS/IPS:

- Snort
- Zeek
- Suricata
- OSSEC

Sobre el último no se efectuará la evaluación considerando que se trata de un HIDS [8]. En el caso de Zeek, tampoco se evaluará en profundidad puesto que se trata de una herramienta puramente IDS [9], lo cual hace innecesaria una comparativa a fondo pues no tienen las mismas capacidades que Snort y Suricata.

3.2 Puntos no técnicos a evaluar

3.2.1 Financiación, sostenibilidad, reputación y credibilidad

Evaluación sobre si se está financiando el proyecto por una entidad, del apoyo que pueda tener de la comunidad y la solidez del equipo de desarrolladores. Sacando conclusiones sobre qué tan sostenible pueda ser el proyecto en los próximos años evaluando si vale la pena utilizarlo en el presente.

En ambos casos son softwares open source, por tanto, no necesitan financiación ni dependen exclusivamente de una (o varias) empresas.

Snort

Hay que tener en cuenta que detrás de Snort se encuentra Cisco como afiliado. Cisco implementa Snort en dispositivos como los Cisco 4000 Series Integrated Services Routers y los Cisco Cloud Services Router 1000v Series. [10]

Esto desemboca en que Snort entre su comunidad open source y su integración con Cisco se pueda considerar que posee una sostenibilidad sólida en los siguientes años.

Al haber salido en 1998 cuenta con más de 600 mil usuarios reportados. [11]

Gracias a este hecho y que es open source desde un inicio, cuenta con una buena reputación en la comunidad.

Suricata

Suricata tiene un equipo definido, el OISF (Open Information Security Information) que se trata de una organización sin ánimo de lucro y figuran como principales desarrolladores de la herramienta. [12]

Surgió en 2010 con avances como el multi-threading lo cual le hizo popularizarse en la comunidad. Actualmente está muy bien reputada no tanto por su trayectoria sino por sus amplias capacidades como sus más de 40 mil reglas. [12]

Conclusión

En este punto Snort es la mejor opción ya que a igualdad de condiciones de desarrollo (al ser open source) también cuenta con la ayuda tanto operativa como financiera que pueda proveer Cisco, una de las empresas más grandes del sector.

Respecto a que Suricata haya podido ganar popularidad en la comunidad en los últimos años es un valor a tener en cuenta (sobre todo si sigue así) pero de momento como herramienta con mejor sostenibilidad y reputación destaca Snort gracias a su gran trayectoria.

3.2.2 Estrategia de desarrollo y futuro

Desgraciadamente en este apartado no se puede concretar y argumentar sobre el futuro ya que los softwares son herméticos en sus próximos pasos innovadores. Obviamente trabajan constantemente en la mejora de rendimiento y fallos.

Por ello se evaluará cuáles han sido los pasos hasta el presente y en base a ello escoger según la tendencia que tengan.

Snort

A pesar de ser del 98, Snort ha sido capaz de reinventarse en los últimos años publicando Snort 3 con mejoras sustanciales como el soporte multi-threading. Sin embargo más allá de este lanzamiento (en el 2021 de forma oficial) no se ha visto más avance.

Suricata

A lo largo de su desarrollo ha podido implementar mejoras tales como el soporte de reglas Lua y la capacidad de ejecución de scripts en tiempo real.

También algo llamativo es observar como el código fuente del software está migrando al lenguaje Rust (actualmente un 15.6%) lo cual representa el compromiso del software con ofrecer la capacidad de procesamiento más eficiente.

Conclusión

En este caso es evidente la superioridad en innovación por parte de Suricata.

3.3 Puntos técnicos a evaluar

3.3.1 Capacidad de procesamiento [17]

Ha de considerarse la capacidad que tiene el software para analizar paquetes. Con un alto tráfico se debe tener un software con capacidad de procesamiento rápida observando así el uso de los recursos del host del software y el porcentaje de pérdidas que se presenta ante situaciones de presión.

Las pruebas fueron realizadas haciendo uso del modo libpcap en el Módulo de Captura de Paquetes que presenta un mejor rendimiento que la librería por defecto AF_Packet. Del mismo modo también se usará el algoritmo Aho-Corasick AC-Split como algoritmo de coincidencia de patrones en el Motor de Detección.

3.3.1.1 Comparación en modo IDS en un entorno estándar [17]

Teniendo un aproximado de 42.000 reglas aplicadas tanto para Snort 3.1 como para Suricata no se han presentado pérdidas de paquetes. Estos son los resultados del uso de la memoria y el consumo CPU.

	Snort 3.1	Suricata
Pico máximo de uso de la memoria	40%	42%
Pico máximo de consumo de la CPU	64%	67%

Tabla 1. Comparación de capacidad de procesamiento entre Snort y Suricata en un entorno estándar [17]

En la primera comparación no se aprecian diferencias notables, una leve superioridad por parte de Snort.

3.3.1.2 Capacidad de procesamiento según la longitud de los paquetes [17]

En este punto se pretende ver diferencias variando la cantidad de paquetes modificando su longitud, pero manteniendo el ancho de banda.

Se analiza en escalones de 100Mbps de ancho de banda. Los valores corresponden al porcentaje de pérdida que sufre el software a determinado bandwidth en paquetes de 512 bytes.

Bandwidth	500 Mbps	600 Mbps	700 Mbps	800 Mbps	900 Mbps	1000 Mbps
Snort 3.1	0%	15.82%	27.85%	36.87%	43.88%	49.5%
Suricata	0%	0%	6.78%	18.43%	27.5%	34.75%

Tabla 2. Comparación de capacidad de procesamiento entre Snort y Suricata variando el ancho de banda en modo IDS [17]

En este caso sí que se distingue una clara ventaja por parte de Suricata que comienza a sufrir pérdidas a partir de un ancho de banda de 700Mbps y mantiene una distancia del 15-20% menos en pérdidas comparado con Snort 3.1.

3.3.1.3 Capacidad de procesamiento en modo IPS [17]

Bandwidth	100 Mbps	200 Mbps	300 Mbps	400 Mbps	500 Mbps	600 Mbps
Snort 3.1	9.85%	18%	25.02%	30.36%	34.19%	45.16%
Suricata	0%	0%	1.48%	7.81%	9.9%	12.9%

Tabla 3. Comparación de capacidad de procesamiento entre Snort y Suricata variando el ancho de banda en modo IPS [17]

De nuevo Suricata presenta dominancia sobre Snort teniendo alrededor de un 30% de diferencia en porcentaje de paquetes perdidos sobre la totalidad.

Conclusión

Hay una clarísima dominancia de Suricata a Snort en cuanto a capacidad de procesamiento.

Snort = R ; Suricata = MB

3.3.2 Reglas, personalización y actualizaciones

Las reglas son uno de los puntos más relevantes de un sistema IDPS. Ya que el software se basa exclusivamente en estas para evaluar los datos de la red y determinar si el flujo de paquetes puede ser una amenaza.

Este apartado evalúa las actualizaciones que estas tienen. Y adicionalmente a la personalización que se pueda hacer a las reglas para adaptarlas a nuestro caso de uso.

Snort

En lo referente a actualizaciones, Snort tiene actualizaciones de las reglas entre 2 a 7 días de forma continuada en su versión de pago proporcionada por Talos, un grupo de expertos en ciberseguridad encargados de mantener las reglas. [22]

La diferencia principal entre las reglas gratuitas y las reglas de pago es que hay un *delay* de 30 días entre que se publica en las reglas de pago y llega a las reglas gratuitas. Pero siempre las reglas que son de pago llegan a las reglas que son gratuitas. [23]

Suricata

En el caso de Suricata tiene la ventaja de que es compatible con todas las reglas de Snort y además también tiene un sistema de gestión de reglas más cómodo implementado ya que es automático llamado suricata-update mientras en Snort, la gestión de las reglas es manual. [24]

Conclusión

En ambos casos hay compatibilidad con Lua, un lenguaje de scripting, lo cual es útil para extender las posibilidades que tienen las reglas.

Por la mayor flexibilidad que tiene Suricata por la adaptación a las reglas de Snort además de las que pueda proveer la comunidad es mejor opción. [25] [26]

3.3.3 Interfaz de monitorización

A fin de facilitar la implementación y la interpretación de los datos.

En este caso no es necesario una comparativa como tal ya que en ambos casos está presente la posibilidad de implementar Kibana que por su popularidad y capacidades es un recolector de datos muy interesante para integrarlo en conjunto ya sea con Snort o Suricata. [27]

Sí es cierto que el stack ELK no es un SIEM debido a que no se toman las decisiones de bloqueo desde la interfaz, es un visualizar. No obstante, de nuevo en este apartado se valoran opciones

robustas para ambos como lo son IBM QRadar [28] o Splunk [29]. A pesar de ser opciones muy capaces, en el caso del QRadar su implementación en este proyecto costaría aproximadamente 370€ - 444€ según su calculadora de costos [30]. En el caso de Splunk serían 2000\$ anuales [31]

En este proyecto se efectuará el análisis de los datos desde el stack ELK.

3.3.4 Adaptabilidad con protocolos y tecnologías emergentes

Evaluación de qué software está adaptado a tecnologías y protocolos de red. Para ello, se desarrolla una tabla comparativa indicando directamente las diferencias entre Suricata y Snort. Es decir, aquellos tipos de paquetes que estén soportados (o no lo estén) por ambos softwares no figurarán pues no es representativo en una comparación.

Este cuadro comparativo se distingue por las capas OSI analizando capa a capa qué aspecto cubre cada software.

Componente del paquete / característica	Snort	Suricata
Capa 2 - Capa de enlace de datos		
ARP Header	No	Limitado
PPPoE	Limitado	Sí
Capa 3 – Capa de red		
IPSec Support, MPLS Label Switching	No	Limitado
Capa 4 – Capa de transporte		
ICMP Timestamp	No	Limitado
Capa 5 – Capa de sesión		
Session Establishment, Session Termination	Limitado	Sí
Capa 6 – Capa de presentación		
Data Encryption	Limitado	Sí
Data Compression, Data Conversion, MIME Type Recognition	No	Limitado
Capa 7 – Capa de aplicación		
Protocol decoding	Limitado	Más extenso
File Extraction, TLS Inspection, Deep Packet Inspection, GeoIP Lookup, SSL Certificate Info, User-Agent String, FTP Commands, SMTP Headers	Limitado	Sí
Behavioral Analysis, Anomaly Detection, HTTP/2 Support, DNS over HTTPS Recognition, SMB Commands	No	Limitado

Tabla 4. Comparación de los tipos de paquetes y protocolos soportados por Snort y Suricata. [32]

Como se puede apreciar en la Tabla 4, Suricata muestra una clara dominancia en todos los aspectos respecto a Snort. Sin embargo, cabe mencionar que, en los protocolos más comunes y usados, Snort si tiene soporte (de la misma forma que Suricata)

3.4 Conclusión y elección del software IDS/IPS

	Snort	Suricata
Puntos no técnicos		
Financiación, sostenibilidad, reputación y credibilidad	MB	B
Estrategia de desarrollo y futuro	R	B
Puntos técnicos		
Capacidad de procesamiento	R	MB
Reglas, personalización y actualizaciones	B	MB
Interfaz de monitorización	-	-
Adaptabilidad con protocolos y tecnologías emergentes	B	MB

Tabla 5. Puntuaciones obtenidas por Snort y Suricata en los puntos abarcados en las secciones 3.2 y 3.3.

Tras haber realizado un análisis detallado de los puntos más representativos a la hora de evaluar ambos softwares, se deben tener en cuenta ciertos aspectos que cobran una mayor importancia dada la relevancia que tienen sobre el resto.

Financiación, sostenibilidad, reputación y credibilidad: en este caso es crucial tener un *background* muy sólido para escoger un software ya que, a carencia de éste, podría suceder que al cabo del tiempo pueda llegar a ser un proyecto discontinuado. A pesar de que este TFG está enfocado en mejorar un sistema de red industrial, se pretende abarcar la causa desde un punto de vista lo más sostenible y realista posible. Por estas razones, este aspecto pondera el doble.

Capacidad de procesamiento: es el aspecto técnico más básico dónde el software debe hacer un uso óptimo de los recursos a disposición. Si este punto está correctamente cubierto, demuestra una mejor capacidad para escalar la red. Este aspecto pondera el doble.

Las puntuaciones quedan de la siguiente forma:

- Snort – 19 Puntos
- Suricata – 25 puntos

Por tanto, vamos a implementar Suricata como IDPS en este TFG.

Capítulo 4. Planteamiento de la red y objetivos

4.1 Condicionantes

Los condicionantes en este proyecto vienen dados por el inventario de dispositivos disponible a utilizar. En ellos, hay una serie de limitaciones que han condicionado la solución propuesta.

A continuación, se detalla el inventario de dispositivos y una explicación de su función y los condicionantes que marca en el proyecto.

- PC Ubuntu 20.04LTS – 8GB de RAM – Intel Xeon E-2224G – Intel UHD Graphics P630

Este dispositivo será el que tenga instalado el IDPS. Su función será escanear la red y bloquear el *malware*. El ordenador, siendo la pieza fundamental en la red y para este TFG, tiene un condicionante grave, se trata de poseer una única interfaz de red. Este punto, como veremos en la solución, ha llevado a que se opte por una solución concreta para alcanzar los objetivos propuestos.

Además ha de tenerse en cuenta que se trata de un dispositivo en cuanto a especificaciones técnicas justo para llevar a cabo esta función. Sin embargo, Suricata es un software, que como se ha constatado en el capítulo comparativo, capaz de aprovechar los recursos del dispositivo que lo ejecuta.

- Portátil Windows 11 – 16GB de RAM – Intel i7-12700H – RTX3060

Es el dispositivo que alberga la máquina virtual atacante. En este caso, no hay condicionante puesto que se trata de un dispositivo muy capaz en términos de especificaciones técnicas, lo cual es necesario para ejecutar una máquina virtual sin ningún problema.

- VM ParrotOS en el portátil Windows.

Desde esta máquina virtual se desarrollarán los ataques en la red. No hay ningún condicionante, la máquina virtual ejecuta las herramientas requeridas sin ningún problema.

- Router Cisco 1921

Es el router que provee conectividad a la red industrial con la red exterior. No condiciona el resultado en ningún aspecto.

- Switch Siemens Scalance X208

Switch encargado de gestionar la conexión de los PLCs a la red. El condicionante en este switch es que su configuración haya sido propuesta únicamente para otorgar conectividad a los PLCs y el correcto funcionamiento de la transmisión de datos por parte de estos. La intencionalidad en este proyecto es poder otorgar una capa de seguridad sin necesidad de configurar los dispositivos industriales.

- 2 Unidades del PLC Siemens Simatic S7-1500

Unidades de PLC destinados a la simulación de un entorno industrial. En este trabajo tienen la funcionalidad de ser las víctimas de los ataques. Dado que tienen un correcto funcionamiento, no suponen ningún condicionante negativo.

Se distinguen dos subredes: 192.168.1.0, red industrial a proteger y 192.168.20.0/24, red externa desde la cual se ejercerán los ataques.

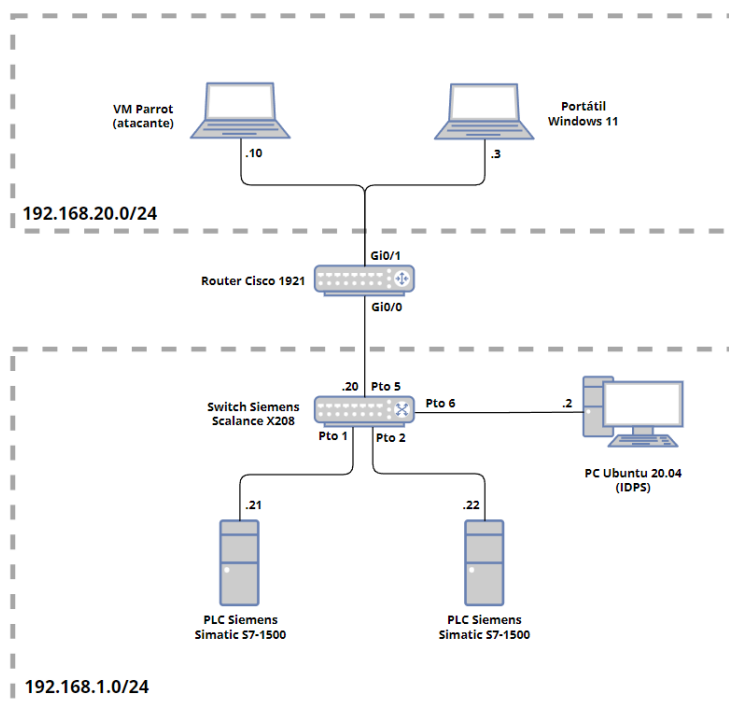


Figura 6. Topología de la red industrial planteada en este TFG.

De la forma que está estructurada la red, es visible un problema fundamental. Si el PC Ubuntu con Suricata está en paralelo con los PLCs y detrás del switch Siemens, ¿Cómo podrá actuar de IPS? Porque topológicamente hablando, no está situado entre el tráfico de la red 192.168.1.0/24 y el de la red 192.168.20.0/24 como lo está por ejemplo el switch Siemens.

La solución sencilla a este problema sería ubicar el PC Ubuntu entre el switch Siemens y el router Cisco. Sin embargo, el PC Ubuntu solo dispone de un puerto ethernet, lo cual imposibilita la capacidad de tener dos conexiones (una hacia el router y otra hacia el switch).

Otro punto a tener en cuenta es que el switch resuelve las rutas de forma automática debido a que encuentra la IP del host en la LAN con la que está configurado. Y entre sus capacidades de configuración la única capacidad de la que dispone el switch para enviar tráfico a Suricata es haciendo *Port mirroring* al puerto dónde está conectada la máquina Ubuntu.

Port Mirroring es una técnica empleada en switches que trata de que todo el tráfico que pasa por un puerto es duplicado en tiempo real al puerto especificado como espejo. [33]

Sin embargo, esta técnica de la que dispone el switch solo es útil en el caso de querer monitorizar la red sin interrumpir en ella. Lo cual quiere decir que se podría utilizar Suricata únicamente en modo detección, IDS.

Claramente se trata de un *hándicap* para el trabajo debido a que el objetivo del mismo es poder usar tanto las herramientas de detección como las de prevención de Suricata. Y la única forma de ello es haciendo que cualquier paquete dirigido a la red a proteger, 192.168.1.0/24, antes de llegar a su destino, deba pasar por Suricata y su correspondiente filtrado.

Para ello, se deben configurar 2 equipos, el router Cisco 1921 y la máquina Ubuntu que alberga Suricata.

4.2 Configuración de los equipos

En el router Cisco 1921 se debe indicar como tal que cualquier paquete dirigido a la red 192.168.1.0/24 tenga como siguiente salto (desde el router) la dirección IP donde está Suricata, 192.168.1.2.

Se programa con los siguientes comandos desde que se entra al router: [34]

- enable
- config t
- ip route 192.168.1.0 255.255.255.0 192.168.1.2

Por otro lado, también ha de configurarse el servidor Ubuntu.

En este caso, se debe utilizar el servidor Ubuntu como Proxy ARP.

ARP es el protocolo de resolución de direcciones y básicamente establece una tabla que relaciona direcciones IP con direcciones MAC. Es importante porque debe haber una asociación entre la dirección de software de un dispositivo (dirección IP) y su dirección de hardware (dirección MAC) para poder enviar los paquetes. Típicamente es el router o switch de una red quién se encarga de establecer dicha tabla ARP. [35] [36]

Si un dispositivo no tiene dicha asociación con el destino en su tabla ARP, envía paquetes al resto de la red preguntando para poder hacer dicha asociación. [35] [36]

En este caso, se configurará el servidor Ubuntu para que actúe de proxy ARP, lo cual implica que responderá a los paquetes de solicitud para resolver dichas asociaciones IP-MAC a otros dispositivos que pretendan establecer una conexión con una IP concreta de la red industrial. En esta respuesta por parte del servidor Ubuntu, establecerá su MAC para resolver la IP solicitada. [35] [36]

Esto quiere decir que, en la red, por ejemplo, en el caso del 192.168.1.21 será el servidor Ubuntu quién tenga la asociación IP-MAC real. El host 192.168.20.10 tendrá en su tabla ARP que debe enviarle al Ubuntu en el caso de querer enviar paquetes al 192.168.1.21.

Para poder hacer uso de estas capacidades en el Ubuntu deberemos introducir una serie de comandos desde el Terminal.

El primero para activar la función de Proxy ARP en el Ubuntu y también la redirección de paquetes (esto será necesario para que los paquetes puedan ser redirigidos a su destino real). [37]

- echo 1 > /proc/sys/net/ipv4/conf/eno1/proxy_arp
- echo 1 > /proc/sys/net/ipv4/ip_forward

Y ahora los comandos para indicar que la máquina Ubuntu responderá a los paquetes ARP de las IPs indicadas: [38]

- ip neigh add proxy 192.168.1.20 dev eno1
- ip neigh add proxy 192.168.1.21 dev eno1
- ip neigh add proxy 192.168.1.22 dev eno1

El siguiente paso será para desactivar las redirecciones automáticas con mejores saltos. Es decir, en el caso de que por ejemplo la IP 192.168.20.10 envíe paquetes a 192.168.1.21 realmente si no está esta opción desactivada lo que sucederá es que la máquina Ubuntu le indicará al Switch Siemens que puede ir de forma directa sin pasar por el Ubuntu. Esto se debe a que cuando el equipo que hace de router encuentra una ruta mejor, envía un paquete *ICMP redirect* [39]

Para desactivarlo se escribe por terminal: [39]

- echo 0 > /proc/sys/net/ipv4/conf/eno1/send_redirects

Hay que tener en cuenta que dónde pone “eno1” hay que poner el nombre de la tarjeta de red del equipo.

Y por último, la modificación de las *ip tables* de la máquina Ubuntu. Es la funcionalidad que tiene Linux para las reglas de filtrado de paquetes, es decir, es básicamente el firewall del sistema. [40]



Se modifican introduciendo los siguientes comandos por el terminal: [41]

- sudo iptables -I INPUT -j NFQUEUE --queue-num 0
- sudo iptables -I OUTPUT -j NFQUEUE --queue-num 0
- sudo iptables -I FORWARD -j NFQUEUE --queue-num 0

Con estos comandos se establece que básicamente cualquier paquete que entre, salga o vaya a ser redirigido, vaya por el NFQUEUE número 0.

NFQUEUE es la cola que utiliza Suricata para el procesado de los paquetes en el modo IPS [41].

Para asegurar la correcta configuración de las iptables se puede ejecutar el siguiente comando: [42]

- sudo iptables -L -v

Por último, hay que modificar un parámetro fundamental en los archivos de configuración de Suricata.

A pesar de tener la instalación de Suricata en el anexo. Es en la configuración para esta red dónde también se modifican los archivos del programa para pasar a Suricata de modo IDS (usando AF-PACKET) a modo IDPS (usando NFQUEUE).

Para ello se modifica el archivo `/etc/default/suricata` comentando la línea que activa el modo AF-PACKET:

```
# LISTENMODE=af-packet
```

Y descomentando la línea que activa el modo NFQUEUE:

```
LISTENMODE=nfqueue
```

Con todas estas configuraciones, ya se tiene la red 192.168.1.0/24 salvaguardada por el software Suricata. Ya se puede proceder a las pruebas de ataque y defensa.

Capítulo 5. Aplicación de Suricata en la red industrial

Cabe resumir brevemente en términos de utilidad para qué sirve en este contexto Suricata y por qué es una buena opción.

Suricata es el software que hará la función de filtrado de todo paquete dirigido a cualquier host de la 192.168.1.0/24. Dicho filtrado depende de las reglas que se hayan establecido en el software y que pueden variar según las necesidades de la red.

No hay ningún paquete que no pase por la cola de filtrado de Suricata, esto otorga mayor control de la clase de tráfico que circula por la red industrial.

Además, tal como se ha explicado en el anterior punto, su implementación ha sido exitosa sin necesidad de tener que reconfigurar el resto de los componentes de la red industrial.

La facilidad de implementación de las reglas de Suricata, que requieren de modificar un archivo, y la capacidad de procesamiento que posee respecto a la competencia, hacen de esta la opción ideal para la implementación de un IDPS en la red industrial.

A continuación, se explica de manera fundamental el funcionamiento sintáctico de las reglas de Suricata.

5.1 Reglas en Suricata [44]

Las reglas en Suricata están fundamentalmente compuestas por 3 partes. Partiendo del siguiente ejemplo:

```
alert http $RED_INTERNA any -> $RED_EXTERNA any (msg:"Mensaje HTTP hacia red exterior"; content:"GET"; sid:1;)
```

Marcada en rojo, es la acción. Marcada en verde, se indica la cabecera. Y marcadas en azul, son las opciones.

5.1.1 La acción

La acción es qué debe suceder cuando la regla se cumpla. Las acciones pueden ser las siguientes:

- *alert*: genera una alerta.
- *pass*: deja pasar el paquete.
- *drop*: genera una alerta y desecha el paquete.
- *reject* y *rejectsrc*: envía un paquete RST/ICMP de error inalcanzable al remitente.
- *rejectdst*: envía un paquete RST/ICMP de error inalcanzable al destinatario.
- *rejectboth*: envía un paquete RST/ICMP de error inalcanzable al remitente y al destinatario.

Se debe tener en cuenta que en modo IPS, las opciones de *reject* también habilitan la opción *drop*.

5.1.2 La cabecera

En esa parte se define el protocolo, la IP origen, puerto origen, IP destino y puerto destino. Todo ello marcando la dirección de la regla.

5.1.2.1 El protocolo.

En esta parte se puede indicar los protocolos más básicos como lo son: TCP, UDP, ICMP o IP. Este último sería equivalente a usar un *any*.

Sin embargo, también es posible hilar más fino con protocolos de la capa 7 como lo son: HTTP, DNS, SSH o WebSocket entre muchos otros.

5.1.2.2 Origen y destino IP

Correspondería al \$RED_INTERNA y \$RED_EXTERNA del ejemplo superior.

En este punto, tal como se puede apreciar, es posible hacer uso de variables. Dichas variables se indican en el suricata.yaml (archivo de configuración).

Se puede indicar la IP de forma literal como podría ser: 192.168.1.20.

También se puede indicar una subred, en este caso indicando su máscara e usando la IP de la subred: 192.168.1.0/24

Es posible hacer uso del operador “!” que hace coincidir todo aquello que no sea como la IP indicada: !192.168.1.20 -> se activaría con cualquier paquete que coincida en las opciones y que no sea de origen (o destino) 192.168.1.20.

Y por último, el operador “[]” que sirve para hacer grupos. Por ejemplo: [192.168.1.0/24, !192.168.1.20] -> la regla se activaría con la subred completa del 192.168.1.0/24 menos con la IP 192.168.1.20.

5.1.2.3 Origen y destino de puertos.

Correspondería a las palabras “any” que hay tras \$RED_INTERNA y \$RED_EXTERNA del ejemplo superior. En la parte izquierda de la flecha sería el origen y en la parte derecha el destino.

En este caso, también se puede indicar el puerto de forma literal, como sería: 80

Pero lo relevante es que hay operadores, el operador de rangos sería “:”, un ejemplo es: [1:1024] -> para que se active la regla si el paquete va dirigido a cualquiera de los puertos del rango.

Está el mismo operador de negación “!”, un ejemplo: !443 -> para que la regla se active si el paquete va dirigido a cualquier puerto que no sea el 443.

Y de nuevo, el operador de grupo “[]”. Un ejemplo: [22:100, ![80,81]] -> en este caso la regla se activaría si el paquete va dirigido a cualquier puerto entre el 22 y el 100 pero que no sea el 80 ni el 81.

5.1.2.4 La dirección de la regla.

Se trata del operador que está entre el puerto origen y la dirección IP destino en este caso, ->.

Este operador puede ser una flecha como ya se observa donde queda claro cual es la dirección pero también puede ser bidireccional, con el operador < >. En este caso, la regla se activaría pudiendo tomar a cualquiera de los dos lados como origen y al otro como destino.

5.1.3 Las opciones

En esta parte se definen las características específicas de la regla.

Las opciones se concatenan unas tras otras con puntos y comas intercaladas. Teniendo la siguiente estructura:

<palabra clave>: <ajuste>;

O también simplemente:

<palabra clave>;



En este punto no se profundizará en las palabras claves que hay disponibles debido a que son una enorme cantidad. Según sea necesario para la seguridad de los ataques en el siguiente apartado, se irá haciendo uso de distintas opciones.

Un ejemplo sencillo es:

```
alert ip any any -> @HOME_NET any(msg:"Tráfico detectado"; sid: 1;)
```

Esta regla se activaría con cualquier tráfico dirigido a la red local. Vemos como hay 2 opciones especificadas.

Una es el mensaje que se desea que aparezca cuando se active la alerta. Y la siguiente opción es el identificativo sid seguido del número con el que se identificará dicha regla.

Capítulo 6. Ataques en la red industrial.

Tras entender la utilidad de Suricata como herramienta de ciberseguridad en el contexto de la red industrial propuesta, se va a proceder a poner a prueba dicho software.

En este capítulo se llevará a cabo una serie de ataques que puede sufrir una red industrial, el resultado que se consigue al ejecutarlos en la red industrial propuesta y la especificación de las reglas necesarias para mitigarlo.

Los ataques provendrán de la máquina virtual ParrotOS con dirección IP 192.168.20.10 hacia la red industrial 192.168.1.0/24. Principalmente hacia el PLC Siemens de IP 192.168.1.21 y en uno de los casos hacia el switch Siemens con IP 192.168.1.20.

6.1 Sondeo de puertos con la herramienta Nmap.

El primer paso que ejecutan la mayor parte de los cibercriminales es el sondeo de puertos. Esta técnica es básica para saber qué dispositivos hay dentro de una red y una vez identificado un objetivo, poder saber al detalle qué puertos están transmitiendo datos y qué puertos están inutilizados. [46]

Un puerto, en términos sencillos, es un punto virtual de una máquina con conexión a una red donde se produce el intercambio de información. Un puerto se ubica en la Capa 4 del modelo OSI, pues solo los protocolos de transporte como TCP y UDP tienen la capacidad de indicar a qué puerto debe ir un paquete. Hay una serie de estándares que especifican unos puertos concretos para determinados tipos de paquetes, como lo es el puerto 25 para el tráfico SMTP (mail) o el 443 para el tráfico HTTP (web). Estos estándares también pueden ser aprovechados por los cibercriminales porque pueden sacar conclusiones de qué servicios está usando la víctima viendo el estado de sus puertos o aprovechar si hay alguna vulnerabilidad asociada a algún protocolo. [45]

En este TFG para el sondeo de puertos se hará uso de la herramienta Nmap, una de las más famosas para esta finalidad. Nmap tiene la capacidad de hacer todos los sondeos de puertos conocidos y tiene también la peculiaridad de poder clasificar los puertos en 6 estados: [47]

- Abierto: son los que aceptan conexiones TCP/UDP. Se trata del estado más vulnerable en términos de ciberseguridad.
- Cerrado: es accesible porque responde a las sondas de Nmap pero no hay ninguna aplicación en él. Pueden servir para saber si un dispositivo está haciendo uso de una dirección IP concreta y detectar el sistema operativo.
- Filtrado: en este estado quiere decir que Nmap no sabe si está abierto o cerrado porque hay un filtrado de paquetes que bloquea las sondas. No se pueden sacar conclusiones cuando se reporta este estado debido a la falta de información.
- No filtrado: el puerto es accesible pero Nmap no determina si está abierto o cerrado. Este estado es resultado solamente del tipo de sondeo ACK. Si se usan otros tipos de sondeo se puede llegar a sacar más conclusiones.
- Abierto|Filtrado: es el estado en el que clasifica Nmap cuando no tiene la certeza de saber si un puerto está abierto o filtrado. Puede concluir en este estado con los tipos de sondeo UDP, protocolo IP, FIN, Null y Xmas.
- Cerrado|Filtrado: análogo al anterior pero en este caso es cuando no tiene la certeza de saber si el puerto está cerrado o filtrado. Este resultado puede ocurrir en un sondeo de tipo IPID pasivo.

A continuación, en los siguientes puntos, se procede a ejecutar los escaneos/sondeos principales y más usados.

6.1.1 Sondeo de descubrimiento de hosts en la red con protocolo TCP. [48]

En este sondeo se hará uso de múltiples opciones agrupadas en una sola gracias al *flag* -A. Concretamente se introducirá desde la máquina atacante el siguiente comando:

- `sudo nmap -A 192.168.1.0/24`

Se debe entender que esta opción concentra múltiples en una. Concretamente, es equivalente a introducir:

- `nmap -O -sV -sC --traceroute 192.168.1.0/24`

Este escaneo al no estar especificando el uso de un protocolo concreto, por defecto Nmap envía ICMP echo request, ICMP timestamp request, TCP SYN al puerto 443 y TCP ACK al puerto 80.

Con la opción -O, se indica que se desea la detección de sistema operativo. En este caso Nmap envía una serie de paquetes TCP/UDP y analiza las respuestas según la base de datos que tiene. De esta forma intenta dar información sobre el fabricante del host, el sistema operativo o el tipo de dispositivo que es.

Con la opción -sV Nmap envía de nuevo una serie de paquetes TCP/UDP con la finalidad de averiguar si el host que escanea se trata de un servidor web, un servidor DNS, etc... También el nombre del host, su número de versión, tipo de equipo además de protocolos de comunicación compatibles como SSH o Telnet.

La opción -sC le indica a Nmap que haga uso de sus scripts predeterminados de nuevo con la finalidad de saber más datos sobre el host al que escanea.

Y por último, la opción --traceroute como bien su nombre indica, sirve para señalarnos las IPs por las que pasa hasta llegar al destino.

El análisis está efectuado sobre la red industrial 192.168.1.0/24.

6.1.1.1 Resultados del escaneo nmap -A 192.168.1.0/24

A continuación, se adjunta una captura del resultado de este escaneo en el caso del 192.168.1.20:

```
Nmap scan report for 192.168.1.20
Host is up (0.0039s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh         OpenSSH 7.2 (protocol 2.0)
| ssh-hostkey:
|_ 1024 38:dd:d7:d1:75:ea:34:95:af:cd:be:6c:ed:04:e7:81 (RSA)
23/tcp    open  telnet      APC PDU/UPS devices or Windows CE telnetd
| fingerprint-strings:
|_ GenericLines:
|_   SIMATIC NET - SCALANCE X200
|_   Address : 00-1B-1B-B4-8C-CC
|_   Device type : SCALANCE X200
|_   PNIO Device name : switchxb18774
|_   Firmware : V 5.2.4.018
|_   Login:
|_   Password:
|_   Login or password incorrect.
|_   Login:
80/tcp    open  http        WindWeb 4.00
|_ http-title: Logon to SCALANCE X Management (192.168.1.20)
84/tcp    open  ctf?
443/tcp   open  ssl/http    WindWeb 4.00
Device type: switch
Running: Avaya embedded
OS details: Avaya 4000- or 5000-series Ethernet Routing switch
Network Distance: 3 hops

TRACEROUTE (using port 53/tcp)
HOP RTT      ADDRESS
- Hops 1-2 are the same as for 192.168.1.2
3  2.20 ms 192.168.1.20
```

Figura 7. Resultado del escaneo Nmap -A desde la máquina atacante dirigido a la red 192.168.1.0/24

En este caso se puede apreciar como si ha detectado tanto los puertos abiertos de comunicación (SSH, TELNET y HTTP) como también que clase de dispositivo es (switch), marca y modelo (Siemens Simatic X200, aunque es realmente el X208).

Esto claramente de por sí puede suponer una vulnerabilidad grave, pues sabiendo todo esto, el atacante ya sabría sin lugar a dudas que está ante una red industrial.

Luego, el resultado del escaneo sobre los PLCs:

```
Nmap scan report for 192.168.1.21
Host is up (0.0019s latency).
All 1000 scanned ports on 192.168.1.21 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
Warning: OSscan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Fortinet FortiGate 1500D firewall (96%), ISS Proventia GX3002 firewall (Linux 2.4
(95%), BlackBerry 7.1 (95%), Check Point ZoneAlarm Z100G firewall (93%), Citrix Access Gateway VPN gatew
(93%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 3 hops

TRACEROUTE (using port 80/tcp)
HOP RTT ADDRESS
- Hops 1-2 are the same as for 192.168.1.2
3 1.36 ms 192.168.1.21

Nmap scan report for 192.168.1.22
Host is up (0.0018s latency).
All 1000 scanned ports on 192.168.1.22 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
Warning: OSscan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Fortinet FortiGate 1500D firewall (96%), ISS Proventia GX3002 firewall (Linux 2.4
(95%), BlackBerry 7.1 (95%), Check Point ZoneAlarm Z100G firewall (93%), Citrix Access Gateway VPN gatew
(93%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 3 hops

TRACEROUTE (using port 80/tcp)
HOP RTT ADDRESS
- Hops 1-2 are the same as for 192.168.1.2
3 1.26 ms 192.168.1.22
```

Figura 8. Resultado del escaneo Nmap -A desde la máquina atacante dirigido a la red 192.168.1.0/24

En este caso al tratarse de dispositivos industriales, el escaneo sobre los puertos típicos se puede observar que no ha ido bien ya que no ha encontrado ningún puerto abierto.

Además, se puede apreciar que el escaneo tanto de servicios como de sistema operativo ha sido erróneo. Arroja un resultado de posibles sistemas operativos dónde no acierta en ningún caso.

Sin embargo, sigue siendo un problema el hecho de que haya encontrado los PLCs porque se puede ejercer un análisis más detallado sobre estos con otro tipo de técnicas.

A continuación la captura del Wireshark desde la máquina Ubuntu. Se ha recortado para poder ir observando distintos momentos del escaneo.

48	0.865957274	192.168.20.10	192.168.1.2	ICMP	60 Echo (ping) request id=0x0f9, seq=0/0, ttl=48 (reply in 50)
49	0.866221626	192.168.20.10	192.168.1.3	ICMP	60 Echo (ping) request id=0x01b3, seq=0/0, ttl=42 (no response found)
50	0.866490803	192.168.1.2	192.168.20.10	ICMP	42 Echo (ping) reply id=0x0f9, seq=0/0, ttl=64 (request in 48)
68	0.96787352	192.168.20.10	192.168.1.20	ICMP	60 Echo (ping) request id=0x7ae4, seq=0/0, ttl=36 (no response found)
69	0.968094814	192.168.20.10	192.168.1.20	ICMP	42 Echo (ping) request id=0x7ae4, seq=0/0, ttl=35 (reply in 74)
70	0.96832709	192.168.20.10	192.168.1.21	ICMP	60 Echo (ping) request id=0xe0bb, seq=0/0, ttl=55 (no response found)
71	0.968793525	192.168.20.10	192.168.1.21	ICMP	42 Echo (ping) request id=0xe0bb, seq=0/0, ttl=54 (reply in 79)
72	0.968862481	192.168.20.10	192.168.1.22	ICMP	60 Echo (ping) request id=0x433a, seq=0/0, ttl=39 (no response found)
73	0.969052465	192.168.20.10	192.168.1.22	ICMP	42 Echo (ping) request id=0x433a, seq=0/0, ttl=38 (reply in 83)
74	0.969168725	192.168.1.20	192.168.20.10	ICMP	60 Echo (ping) reply id=0x7ae4, seq=0/0, ttl=64 (request in 69)
1464	18.732967910	192.168.20.10	192.168.1.2	TCP	60 57664 -> 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1465	18.733264518	192.168.20.10	192.168.1.20	TCP	60 57664 -> 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1466	18.733554776	192.168.20.10	192.168.1.21	TCP	60 57664 -> 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1467	18.733555004	192.168.20.10	192.168.1.23	TCP	60 57664 -> 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1476	18.734370318	192.168.1.21	192.168.20.10	TCP	60 80 -> 57664 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1476	18.734370318	192.168.20.10	192.168.20.10	TCP	60 80 -> 57664 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1477	18.734447488	192.168.1.2	192.168.20.10	TCP	54 53 -> 57664 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1478	18.734448252	192.168.20.10	192.168.1.20	TCP	58 [TCP Retransmission] [TCP Port numbers reused] 57664 -> 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1479	18.734489236	192.168.1.22	192.168.20.10	TCP	54 80 -> 57664 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1480	18.734611080	192.168.20.10	192.168.1.21	TCP	58 [TCP Retransmission] [TCP Port numbers reused] 57664 -> 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1481	18.734681345	192.168.1.21	192.168.20.10	TCP	54 80 -> 57664 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1482	18.735045434	192.168.20.10	192.168.1.22	TCP	60 57664 -> 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1483	18.735045590	192.168.1.21	192.168.20.10	TCP	60 53 -> 57664 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
22842	125.934430163	192.168.20.10	192.168.1.2	SSHv2	338 Client: Diffie-Hellman Key Exchange Init
22843	125.934430887	192.168.20.10	192.168.20.10	TCP	74 [TCP Retransmission] [TCP Port numbers reused] 40300 -> 443 [SYN] Seq=0 Win=5240 Len=0 MSS=1460 SACK_PERM TSval=1136779913 TSecr=0 WS=128
22844	125.934893933	192.168.1.22	192.168.20.10	TCP	66 12 -> 55188 [ACK] Seq=1123 Ack=429 Win=64512 Len=0 TSval=913167191 TSecr=1981634688
22845	125.935142109	192.168.1.20	192.168.20.10	TCP	74 443 -> 40300 [SYN, ACK] Seq=0 Ack=1 Win=16384 Len=0 MSS=1460 WS=1 TSval=541440 TSecr=1136779913
22846	125.935163684	192.168.1.20	192.168.20.10	TCP	74 [TCP Retransmission] 443 -> 40300 [SYN, ACK] Seq=0 Ack=1 Win=16384 Len=0 MSS=1460 WS=1 TSval=541440 TSecr=1136779913
22847	125.938402825	192.168.1.20	192.168.1.20	TCP	66 40300 -> 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1136779918 TSecr=541440
22848	125.939339316	192.168.20.10	192.168.20.10	TCP	58 [TCP Retransmission] [TCP Port numbers reused] 40300 -> 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1136779918 TSecr=541440
22849	125.939321084	192.168.20.10	192.168.1.20	TLSv1.2	583 Client: Hello
22850	125.93941758	192.168.20.10	192.168.1.20	TCP	583 [TCP Retransmission] 40300 -> 443 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=517 TSval=1136779918 TSecr=541440
22852	125.942089626	192.168.1.20	192.168.20.10	TCP	66 443 -> 40300 [ACK] Seq=1 Ack=518 Win=16859 Len=0 TSval=541441 TSecr=1136779918
22853	125.942089624	192.168.1.20	192.168.20.10	TCP	66 [TCP Dup ACK 22852#1] 443 -> 40300 [ACK] Seq=1 Ack=518 Win=16859 Len=0 TSval=541441 TSecr=1136779918
22933	126.941180362	192.168.20.10	192.168.1.20	HTTP	255 551T_DUPLEX_POST /sra/BAU5980-C049-4580-9E23-C84EE04DCD537/ HTTP/1.1

Figura 9. Captura de Wireshark desde el Ubuntu del escaneo Nmap -A dirigido a la red 192.168.1.0/24

Se pueden observar tanto los ICMP Request que envía al inicio del escaneo, como los paquetes TCP al puerto 80. Tras ello en las franjas rojas se ve como se cumple una comunicación TCP entre el atacante y los hosts de la red industrial. En las franjas negras y azul claro hay una prueba de comunicación SSHv2, TLSv1.2 e incluso en la última franja (color verde) un paquete de tipo HTTP.

6.1.1.2 *Cómo proteger a la subred del escaneo nmap -A 192.168.1.0/24 [44]*

Debido a que este escaneo hace uso de múltiples técnicas, se deben incorporar reglas para cada una de ellas con el fin de poder detectar si por ejemplo se está haciendo un escaneo ICMP o un escaneo TCP SYN.

- `drop tcp any any -> 192.168.1.0/24 any (msg:"Escaneo TCP SYN detectado"; flags:S; threshold:type limit, track by_src, count 10, seconds 2; sid:1;)`

En este caso, se parte de querer identificar todos los paquetes TCP dirigidos a la red industrial. Se le indican en las opciones que deseche los paquetes con la bandera SYN a 1. Y luego se establece el umbral de tasa de paquetes con las opciones de threshold, concretamente se establece que rastree por origen y que se active la regla si detecta 10 paquetes en un plazo de 2 segundos.

- `drop tcp any any -> 192.168.1.0/24 any (msg:"Escaneo Deteccion de Versiones"; flow:to_server,established; content:"SSH-"; depth:4; sid:2;)`

En esta regla se hace el uso de Flow. En este caso las opciones de Flow indican que la regla se aplica hacia conexiones establecidas que van hacia el servidor y con el contenido "SSH-". Con la opción Depth se indica que se analicen solo los 4 primeros bytes (suficientes para ver el contenido).

- `drop icmp any any -> 192.168.1.0/24 any (msg:"Escaneo ICMP Ping"; itype:8; threshold:type limit, track by_src, count 5, seconds 10; sid:3;)`

En esta regla se filtra por ICMP y concretamente los paquetes ICMP del tipo 8 que son los correspondientes a los paquetes Ping. De nuevo se establecen las opciones para detectar una tasa de paquetes de este tipo inusual, en el caso de haber 5 paquetes en una ventana de 10 segundos.

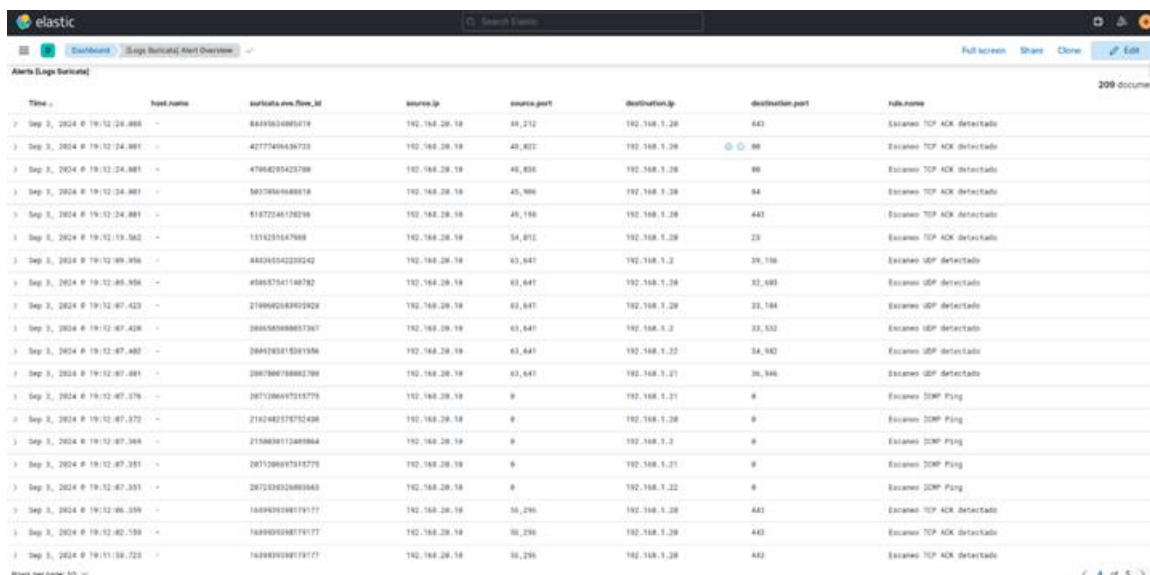
- `drop udp any any -> 192.168.1.0/24 any (msg:"Escaneo UDP detectado"; threshold:type limit, track by_src, count 5, seconds 2; sid:4;)`

Esta regla está formada con la misma base que la anterior. Se activa al detectar un alto tráfico de paquetes UDP, estableciendo el umbral en 5 paquetes cada 2 segundos.

`drop tcp any any -> 192.168.1.0/24 any (msg:"Escaneo TCP ACK detectado"; flags:A; threshold:type limit, track by_src, count 10, seconds 2; sid:5;)`

La regla superior es exactamente igual que la primera regla, propuesta para detectar el escaneo SYN. La diferencia radica en que en este caso es para el escaneo TCP ACK, por ello cambia la bandera a escanear, siendo la bandera A, activada en los paquetes del tipo ACK.

A continuación, se adjunta una captura de Kibana, indicando las alertas, en este caso por las reglas superiores:



Time	host.name	suricata.rule.id	source.ip	source.port	destination.ip	destination.port	rule.name
3 Sep 3, 2024 @ 19:12:24.888	-	848763680578	192.168.20.18	49,212	192.168.1.20	443	Escaneo TCP ACK detectado
3 Sep 3, 2024 @ 19:12:24.891	-	427749636723	192.168.20.18	49,833	192.168.1.20	80	Escaneo TCP ACK detectado
3 Sep 3, 2024 @ 19:12:24.891	-	47608295423768	192.168.20.18	49,833	192.168.1.20	80	Escaneo TCP ACK detectado
3 Sep 3, 2024 @ 19:12:24.891	-	5673966688918	192.168.20.18	49,989	192.168.1.20	84	Escaneo TCP ACK detectado
3 Sep 3, 2024 @ 19:12:24.891	-	51872245128296	192.168.20.18	49,198	192.168.1.20	443	Escaneo TCP ACK detectado
3 Sep 3, 2024 @ 19:12:19.562	-	1182291617888	192.168.20.18	54,812	192.168.1.20	23	Escaneo TCP ACK detectado
3 Sep 3, 2024 @ 19:12:09.956	-	44836452229242	192.168.20.18	63,441	192.168.1.2	39,184	Escaneo UDP detectado
3 Sep 3, 2024 @ 19:12:09.956	-	458837341148782	192.168.20.18	63,441	192.168.1.20	32,980	Escaneo UDP detectado
3 Sep 3, 2024 @ 19:12:07.423	-	2199640249292928	192.168.20.18	63,441	192.168.1.20	33,184	Escaneo UDP detectado
3 Sep 3, 2024 @ 19:12:07.426	-	28865688883267	192.168.20.18	63,441	192.168.1.2	32,332	Escaneo UDP detectado
3 Sep 3, 2024 @ 19:12:07.480	-	2884263815341386	192.168.20.18	63,441	192.168.1.22	34,182	Escaneo UDP detectado
3 Sep 3, 2024 @ 19:12:07.481	-	2887889788882788	192.168.20.18	63,441	192.168.1.21	36,346	Escaneo UDP detectado
3 Sep 3, 2024 @ 19:12:07.476	-	287128687515779	192.168.20.18	0	192.168.1.21	0	Escaneo DMP Ping
3 Sep 3, 2024 @ 19:12:07.472	-	218248275752408	192.168.20.18	0	192.168.1.20	0	Escaneo DMP Ping
3 Sep 3, 2024 @ 19:12:07.388	-	215888911348984	192.168.20.18	0	192.168.1.2	0	Escaneo DMP Ping
3 Sep 3, 2024 @ 19:12:07.351	-	287128687515779	192.168.20.18	0	192.168.1.21	0	Escaneo DMP Ping
3 Sep 3, 2024 @ 19:12:07.351	-	287338328888888	192.168.20.18	0	192.168.1.22	0	Escaneo DMP Ping
3 Sep 3, 2024 @ 19:12:06.339	-	168888888819177	192.168.20.18	58,296	192.168.1.20	443	Escaneo TCP ACK detectado
3 Sep 3, 2024 @ 19:12:02.338	-	168888888819177	192.168.20.18	58,296	192.168.1.20	443	Escaneo TCP ACK detectado
3 Sep 3, 2024 @ 19:11:58.722	-	168888888819177	192.168.20.18	58,296	192.168.1.20	443	Escaneo TCP ACK detectado

Figura 10. Captura de Kibana del escaneo Nmap -A dirigido a la red 192.168.1.0/24

Se puede observar como Suricata se ha activado en multitud de ocasiones debido a los escaneos dirigidos a los hosts de la subred.

Cabe destacar que gracias a esta serie de reglas, los resultados de Nmap han sido notoriamente peores.

```
Nmap scan report for 192.168.1.2
Host is up, received timestamp-reply ttl 63 (0.0017s latency).
Scanned at 2024-09-02 18:47:29 CEST for 55s
All 1000 scanned ports on 192.168.1.2 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
Too many fingerprints match this host to give specific OS details
TCP/IP fingerprint:
SCAN(V=7.94SVN%E=4%D=9/2%OT=%CT=%CU=%PV=Y%G=N%TM=66D5EC58%P=x86_64-pc-linux-gnu)
SEQ()
U1(R=N)
IE(R=N)

TRACEROUTE (using proto 1/icmp)
HOP RTT ADDRESS
1 1.01 ms 192.168.20.1
2 ... 30

Nmap scan report for 192.168.1.20
Host is up, received timestamp-reply ttl 62 (0.0058s latency).
Scanned at 2024-09-02 18:47:29 CEST for 55s
All 1000 scanned ports on 192.168.1.20 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
Too many fingerprints match this host to give specific OS details
TCP/IP fingerprint:
SCAN(V=7.94SVN%E=4%D=9/2%OT=%CT=%CU=%PV=Y%G=N%TM=66D5EC58%P=x86_64-pc-linux-gnu)
SEQ()
U1(R=N)
IE(R=N)
```

Figura 11. Captura del resultado del Nmap -A dirigido a la red 192.168.1.0/24 tras incorporar las reglas de Suricata

No ha sido capaz de sacar datos sobre los PLCs ubicados en las IP 192.168.1.21 y .22. Además, se ve como ha sido muy ambiguo con la información que ha extraído del switch Siemens.

6.1.2 Sondeo de descubrimiento de hosts en la red con protocolo UDP. [48][49]

El protocolo UDP es un protocolo en el que sí que se contempla la pérdida de paquetes ya que no hacen la transmisión de un paquete ACK como en TCP para asegurar que ha sido recibido. Sumado a esto, también se debe tener en cuenta que el sondeo UDP es mucho más lento debido a que el buffer de entrada de paquetes UDP es relativamente limitado, esto hace que no se pueda enviar una gran cantidad de paquetes por segundo.

Salvo para los puertos típicamente utilizados por el protocolo UDP, como lo son el 67 o 161, destinados para los protocolos DHCP y SNMP respectivamente, en los cuales si se envía una carga útil especificada por el protocolo, se envían paquetes vacíos de contenido (a menos que se quiera especificar con las flags de --data).

Si responde con un paquete ICMP tipo 3, código 3 quiere decir que el puerto está cerrado. En caso de que sean otros códigos como el 0, 1, 2, 9, 10 o 13, el puerto se marcará como filtrado. Si el puerto responde con otro paquete UDP, quiere decir que está abierto. Si no responde, se marca como abierto|filtrado.

Hacer un escaneo UDP no es muy común porque requiere de mucho tiempo, debido a la naturaleza del protocolo. Pero esto no quiere decir que no puede dar pie a la búsqueda de vulnerabilidades.

El comando a ejecutar en este caso es:

- `sudo nmap -sU 192.168.1.0/24`

6.1.2.1 Resultados del escaneo `nmap -sU 192.168.1.0/24`

Captura desde la máquina atacante del resultado del escaneo UDP:

```
Nmap scan report for 192.168.1.1
Host is up (0.0013s latency).
Not shown: 999 filtered udp ports (port-unreach)
PORT      STATE      SERVICE
686/udp   open|filtered hcp-wismar

Nmap scan report for 192.168.1.2
Host is up (0.0019s latency).
Not shown: 996 closed udp ports (port-unreach)
PORT      STATE      SERVICE
68/udp    filtered   dhcpc
69/udp    open|filtered tftp
631/udp   open|filtered ip
5353/udp  open|filtered zeroconf

Nmap scan report for 192.168.1.20
Host is up (0.0034s latency).
Not shown: 995 closed udp ports (port-unreach)
PORT      STATE      SERVICE
68/udp    filtered   dhcpc
69/udp    open|filtered tftp
161/udp   open       snmp
49152/udp open|filtered unknown
49153/udp open|filtered unknown

Nmap scan report for 192.168.1.21
Host is up (0.0041s latency).
Not shown: 998 open|filtered udp ports (no-response)
PORT      STATE      SERVICE
68/udp    filtered   dhcpc
161/udp   open       snmp

Nmap scan report for 192.168.1.22
Host is up (0.0050s latency).
Not shown: 998 open|filtered udp ports (no-response)
PORT      STATE      SERVICE
68/udp    filtered   dhcpc
161/udp   open       snmp
```

Figura 12. Resultado del escaneo `Nmap -sU` desde la máquina atacante dirigido a la red `192.168.1.0/24`

De nuevo, se puede observar como el escaneo es exitoso al encontrar todos los hosts posibles en la red industrial.

Sin embargo, también es cierto que la información que brinda no es concluyente para entender qué clase de dispositivos son. Se puede observar que hay servicios como el DHCP o SNMP en común entre los hosts.

Véase la captura Wireshark desde la máquina Ubuntu:

1466	18.878145921	192.168.20.10	192.168.1.2	UDP	82 47377 → 49640 Len=40
1467	18.878146105	192.168.20.10	192.168.1.20	UDP	82 47377 → 49640 Len=40
1468	18.878447604	192.168.20.10	192.168.1.21	UDP	82 47377 → 49640 Len=40
1469	18.878447773	192.168.20.10	192.168.1.22	UDP	82 47377 → 49640 Len=40
1470	18.878487769	192.168.20.10	192.168.1.20	UDP	82 47377 → 49640 Len=40
1471	18.878553563	192.168.1.2	192.168.20.10	ICMP	110 Destination unreachable (Port unreachable)
1472	18.878564992	192.168.20.10	192.168.1.22	UDP	82 47377 → 49640 Len=40
1473	18.878580932	192.168.20.10	192.168.1.21	UDP	82 47377 → 49640 Len=40
1474	18.878770005	192.168.1.20	192.168.20.10	ICMP	126 Destination unreachable (Port unreachable)
1475	18.878871179	192.168.1.20	192.168.20.10	ICMP	126 Destination unreachable (Port unreachable)
9433	61.536722438	192.168.20.10	192.168.1.22	TFTP	61 Read Request, File: r7tftp.txt, Transfer type: octet
9434	61.537382991	192.168.1.21	192.168.20.10	TFTP	85 Unknown (0x3029)
9435	61.537525842	192.168.1.21	192.168.20.10	TFTP	85 Unknown (0x3029)

Figura 13. Captura Wireshark desde el Ubuntu del escaneo Nmap -sU dirigido a la red 192.168.1.0/24

En la imagen se puede observar como la máquina atacante va enviando los paquetes UDP a los distintos hosts de la subred. Y a su vez dichos hosts le van respondiendo a medida que hayan puertos ocupados, indicándole que no pueden mantener una comunicación en este.

6.1.2.2 Cómo proteger a la subred del escaneo nmap -sU 192.168.1.0/24 [44]

En este caso el conjunto de reglas que será necesario en Suricata será menor debido a que realmente es un escaneo más sencillo.

- drop udp any any -> 192.168.1.0/24 any (msg:"Escaneo UDP Detectado"; threshold:type limit, track by_src, count 5, seconds 2; sid:1000010;)

Es la misma regla UDP establecida en el anterior caso. Su propósito es detectar cuando comienza a haber una tasa de paquetes UDP inusual desde la misma fuente.

- drop icmp any any -> any any (msg:"ICMP Port Unreachable - Escaneo UDP Detectado"; itype:3; icode:3; threshold:type limit, track by_dst, count 5, seconds 2; sid:1000012;)

En este caso esta regla atiende a las peculiaridades del escaneo UDP. Como se ha indicado anteriormente, cuando un puerto UDP se encuentra ocupado, el dispositivo sí lo indica a través de un ICMP Port Unreachable. Esta regla sirve para detectar si comienza a haber una tasa impropia de este tipo de paquetes. Para ello se filtra por paquete ICMP de tipo 3 y código 3, correspondiente al Port Unreachable. Posteriormente se establece el umbral de una tasa anormal en 5 paquetes en una ventana de 2 segundos.

- drop udp any any -> 192.168.1.0/24 53 (msg:"Escaneo UDP hacia puerto DNS"; threshold:type limit, track by_src, count 5, seconds 2; sid:1000013;)

En esta regla se pretende detectar un alto tráfico dirigido al puerto 53 del dispositivo, usado para el protocolo DNS. Para ello, como se puede observar, en el destino indicamos únicamente este puerto y en las opciones de la regla indicamos el umbral en 5 paquetes en el plazo de 2 segundos.

- drop udp any any -> 192.168.1.0/24 161 (msg:"Escaneo UDP hacia puerto SNMP"; threshold:type limit, track by_src, count 5, seconds 2; sid:1000014;)

Esta regla es análoga a la anterior pero con la finalidad de detectar si hay una alta tasa de paquetes dirigido al puerto SNMP.

A continuación se puede observar el resultado desde Kibana tras la activación de las reglas y el escaneo lanzado de nuevo.

Alert	Time	Source	Destination	Count	Severity	Message
Escaneo UDP Detectado	2024-09-24 19:24:56.489	13148802249617	192.168.20.10	45,815	68	Escaneo UDP Detectado
Escaneo UDP Detectado	2024-09-24 19:24:56.493	812226787262387	192.168.20.10	45,823	68	Escaneo UDP Detectado
Escaneo UDP Detectado	2024-09-24 19:24:56.436	163128244999888	192.168.20.10	43,831	68	Escaneo UDP Detectado
30MP Port Unreachable - Escaneo UDP Detectado	2024-09-24 19:24:56.897	219688632785385	127.0.0.1	0	0	30MP Port Unreachable - Escaneo UDP Detectado
30MP Port Unreachable - Escaneo UDP Detectado	2024-09-24 19:24:56.897	18831768402480	127.0.0.1	0	0	30MP Port Unreachable - Escaneo UDP Detectado
30MP Port Unreachable - Escaneo UDP Detectado	2024-09-24 19:24:56.897	128173972621779	127.0.0.1	0	0	30MP Port Unreachable - Escaneo UDP Detectado
30MP Port Unreachable - Escaneo UDP Detectado	2024-09-24 19:24:56.897	188318861284035	127.0.0.1	0	0	30MP Port Unreachable - Escaneo UDP Detectado
30MP Port Unreachable - Escaneo UDP Detectado	2024-09-24 19:24:56.897	187377047647699	127.0.0.1	0	0	30MP Port Unreachable - Escaneo UDP Detectado
Escaneo UDP Detectado	2024-09-24 19:23:31.485	282238733992379	192.168.20.10	45,829	68	Escaneo UDP Detectado
Escaneo UDP Detectado	2024-09-24 19:23:31.377	188626347972428	192.168.20.10	45,827	68	Escaneo UDP Detectado
30MP Port Unreachable - Escaneo UDP Detectado	2024-09-24 19:23:31.148	384775453888886	127.0.0.1	0	0	30MP Port Unreachable - Escaneo UDP Detectado
30MP Port Unreachable - Escaneo UDP Detectado	2024-09-24 19:23:31.148	284772343388886	127.0.0.1	0	0	30MP Port Unreachable - Escaneo UDP Detectado
30MP Port Unreachable - Escaneo UDP Detectado	2024-09-24 19:23:31.148	2842884861641386	127.0.0.1	0	0	30MP Port Unreachable - Escaneo UDP Detectado
30MP Port Unreachable - Escaneo UDP Detectado	2024-09-24 19:23:31.148	116199798268483	127.0.0.1	0	0	30MP Port Unreachable - Escaneo UDP Detectado
30MP Port Unreachable - Escaneo UDP Detectado	2024-09-24 19:23:31.148	2842884861641386	127.0.0.1	0	0	30MP Port Unreachable - Escaneo UDP Detectado
Escaneo UDP Detectado	2024-09-24 19:23:16.359	178038244999597	192.168.20.10	45,825	68	Escaneo UDP Detectado
Escaneo UDP Detectado	2024-09-24 19:22:46.322	184788578288012	192.168.20.10	45,823	68	Escaneo UDP Detectado
Escaneo UDP hacia puerto DNS	2024-09-24 19:22:46.888	187714013178688	192.168.20.10	45,816	83	Escaneo UDP hacia puerto DNS
Escaneo UDP hacia puerto DNS	2024-09-24 19:22:46.787	4136782333949	192.168.20.10	45,814	83	Escaneo UDP hacia puerto DNS

Figura 14. Captura de Kibana del escaneo Nmap -sU dirigido a la red 192.168.1.0/24

De nuevo, las reglas se han activado con éxito.

En el caso de la máquina atacante, ha tenido los mismos resultados pese a las reglas aplicadas. Sin embargo, puede considerarse beneficioso conocer que efectivamente ha habido un escaneo en la red.

6.1.3 Sondeo de puertos sobre un host con técnicas de evasión IDS. [48]

En esta ocasión, se hace uso de un comando considerablemente más complejo con la intención de poner a prueba las capacidades de Suricata.

➤ `sudo nmap -sS -D RND:10 -f -T2 -p 1-1024 192.168.1.21`

Primero, la bandera -sS, que significa Stealth Scan. Esta es la técnica de escaneo más popular. Básicamente se basa en abrir conexiones TCP y nunca cerrarlas. Debemos saber que una conexión TCP, en términos simples, se establece mediante el *three-way handshake*. Este método se trata de que el primer mensaje que envía el cliente al servidor es un paquete SYN, el servidor devuelve un SYN-ACK para hacer ver que está disponible y ha recibido correctamente el paquete y por último para establecer la conexión el cliente le envía un paquete ACK.

En este tipo de sondeo, se envía primeramente el paquete SYN y se espera a la respuesta. Si devuelve un SYN/ACK significa que el puerto está abierto. Por otro lado, si devuelve un RST significa que el puerto no está disponible. Si en cambio no responde o da un error de inalcanzable (tipo 3, códigos 0, 1, 2, 3, 9, 10 o 13) quiere decir que el puerto está en estado “filtrado”. Esta técnica es la más popular porque al no cerrar la conexión, es difícil de identificar.

La bandera -D RND:10 indica a Nmap hacer uso de direcciones IP distintas a la original, señuelos. Concretamente en este caso, de 10 tipos de direcciones IP totalmente aleatorias. De esta forma parecerá que el escaneo se realiza desde 10 IPs distintas, y para las reglas de Suricata es más difícil de detectar.

La bandera -f hace que los paquetes viajen fragmentados. De esta forma detectar los patrones es más difícil porque Suricata tendría que ponerse a reconstruirlos y leerlos, todo esto en un tiempo mínimo que no suponga un cuello de botella, lo cual es inviable.

La bandera -T2 indica una velocidad de envío de paquetes ligeramente lenta, en una escala del 1 al 5. Es otra técnica que ayuda a no ser detectado fácilmente.

Tras esta bandera se indica en -p 1-1024 que se pretenden escanear los primeros 1024 puertos de la dirección IP 192.168.1.21.

6.1.3.1 Resultados del escaneo de puertos sobre un host con técnicas de evasión

Véase el resultado desde la máquina atacante de este escaneo:

```
[root@parrot:~/home/valentino]
└─# nmap -sS -D RND:10 -f -T2 -p 1-1024 192.168.1.21
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-02 17:08 CEST
Stats: 0:01:26 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 17.58% done; ETC: 17:15 (0:05:42 remaining)
Stats: 0:04:19 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 59.77% done; ETC: 17:15 (0:02:46 remaining)
Stats: 0:05:50 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 82.03% done; ETC: 17:15 (0:01:14 remaining)
Stats: 0:06:39 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 93.85% done; ETC: 17:15 (0:00:25 remaining)
Nmap scan report for 192.168.1.21
Host is up (0.0048s latency).
Not shown: 1023 closed tcp ports (reset)
PORT      STATE SERVICE
102/tcp   open  iso-tsap
```

Figura 15. Resultado del escaneo Nmap -sU desde la máquina atacante dirigido a la red 192.168.1.0/24

Con este escaneo, es posible sacar información muy valiosa pues con una rápida búsqueda en internet, se averigua que el puerto 102 con servicio iso-tsap es utilizado generalmente para la comunicación de los PLCs Siemens. [57]

Y a continuación véase las capturas de Wireshark desde el Ubuntu:

865	15.324935579	98.211.194.254	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=0, ID=1de9) [Reassembled in #867]
866	15.325226279	98.211.194.254	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=8, ID=1de9) [Reassembled in #867]
867	15.325489643	98.211.194.254	192.168.1.21	TCP	60	48952 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
868	15.325752756	174.128.7.222	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=0, ID=1de9) [Reassembled in #870]
869	15.326015509	174.128.7.222	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=8, ID=1de9) [Reassembled in #870]
870	15.326015534	174.128.7.222	192.168.1.21	TCP	60	48952 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
871	15.326276679	155.143.156.5	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=0, ID=1de9) [Reassembled in #873]
872	15.326276701	155.143.156.5	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=8, ID=1de9) [Reassembled in #873]
873	15.326537794	155.143.156.5	192.168.1.21	TCP	60	48952 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
874	15.326800045	87.126.69.186	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=0, ID=1de9) [Reassembled in #876]
875	15.326800068	87.126.69.186	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=8, ID=1de9) [Reassembled in #876]
876	15.327062994	87.126.69.186	192.168.1.21	TCP	60	48952 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
877	15.327063016	94.58.187.41	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=0, ID=1de9) [Reassembled in #879]
878	15.327320982	94.58.187.41	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=8, ID=1de9) [Reassembled in #879]
879	15.327321004	94.58.187.41	192.168.1.21	TCP	60	48952 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
880	15.327581415	192.168.20.10	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=0, ID=1de9) [Reassembled in #882]
881	15.327581438	192.168.20.10	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=8, ID=1de9) [Reassembled in #882]
882	15.327581457	192.168.20.10	192.168.1.21	TCP	60	48952 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
883	15.327843578	154.248.53.83	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=0, ID=1de9) [Reassembled in #885]
884	15.327843599	154.248.53.83	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=8, ID=1de9) [Reassembled in #885]
885	15.327843617	154.248.53.83	192.168.1.21	TCP	60	48952 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
886	15.328103753	25.226.198.9	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=0, ID=1de9) [Reassembled in #888]
887	15.328103776	25.226.198.9	192.168.1.21	IPv4	60	Fragmented IP protocol (proto=TCP 6, off=8, ID=1de9) [Reassembled in #888]
888	15.328103795	25.226.198.9	192.168.1.21	TCP	60	48952 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460

Figura 16. Captura de Wireshark del escaneo de puertos Nmap -sS -D RND:10 -f -T2 -p 1-1024 192.168.1.21

Es evidente que los paquetes dirigidos al PLC .21 parece que provengan de múltiples IP además de haber una gran cantidad de paquetes fragmentados, los cuales dificultan poder leer su contenido.

6.1.3.2 Cómo proteger a la subred de escaneos ejecutados con técnicas de evasión [44]

En este caso si se debe hacer uso de reglas en Suricata que sean precisas en su función.

- drop tcp any any -> 192.168.1.21 any (msg:"Escaneo con Decoy detectado"; flags:S; flow:stateless; threshold:type both, track by_dst, count 10, seconds 5; sid:1;)

Esta regla es para cualquier paquete dirigido al PLC .21 enfocándose los paquetes TCP de tipo SYN. Con la opción Flow:stateless lo que se consigue es escanear los paquetes que no tengan una sesión establecida, con esta opción se centra en las múltiples IPs ya que no van a establecer una sesión con cada IP. El threshold está configurado de tal forma que el umbral de contar 10 en 5 segundos se aplique en ambas direcciones.

- drop tcp any any -> 192.168.1.0/24 any (msg:"Escaneo SYN detectado"; flags:S; threshold:type limit, track by_src, count 10, seconds 2; sid:2;)

De nuevo se puede hacer uso de la regla de escaneo TCP SYN usada anteriormente para toda la red.

- drop tcp any any -> 192.168.1.21 any (msg:"Escaneo de puertos múltiple"; flags:S; threshold:type both, track by_src, count 20, seconds 10; sid:3;)

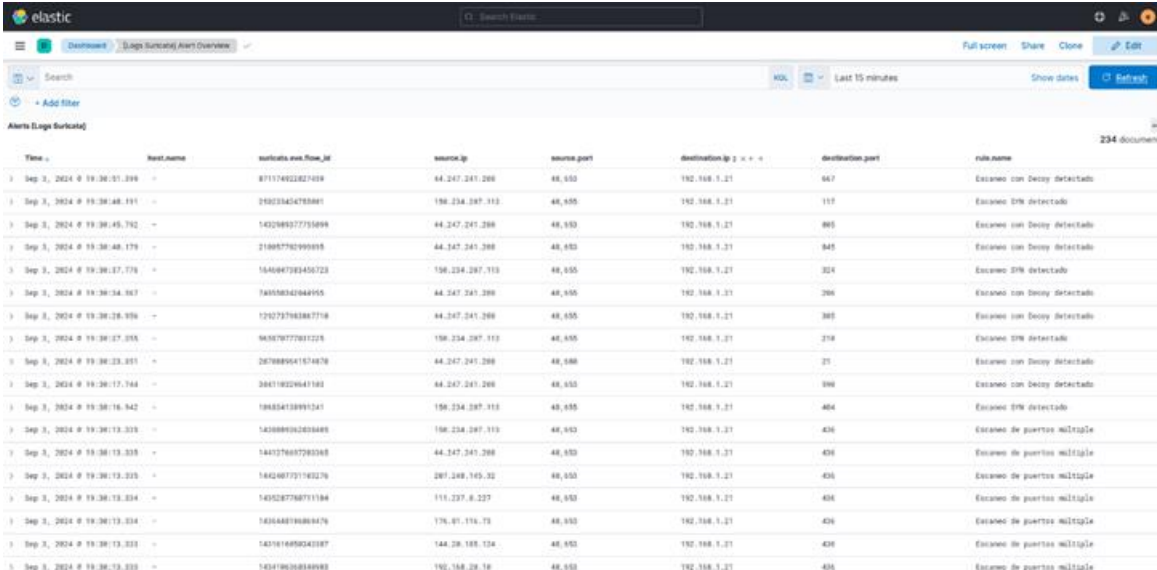
Esta regla es muy similar a las anteriores, la diferencia radica en que se varía el umbral (20 paquetes en 10 segundos) con tal de dar sensibilidad a Suricata y se rastrea por la IP origen para ver cuántos intentos de establecer conexiones TCP hace.

Luego también como idea, es posible implementar un dropeo de paquetes fragmentado. Pero esto es muy delicado ya que la fragmentación de paquetes como tal no es una actividad inusual, lo que sucede es que si se tiene conocimiento del tipo de conexiones que tendrá la red, puedes concluir si en este caso sí correspondería a una actividad inusual. Sería de la siguiente forma:

- drop ip any any -> 192.168.1.21 any (msg:"Fragmento IP sospechoso"; fragbits:M; dsize:<64; sid:1000003;)

Esta regla detecta cualquier paquete IP fragmentado gracias a la opción de fragbits. Además también se ha especificado que solo los paquetes menores a 64 bytes con la opción dsize.

El resultado desde Kibana es el siguiente:



Time	host.name	suricata.rule_flow_id	source.ip	source.port	destination.ip	destination.port	rule.name
1 Sep 3, 2024 @ 19:30:01.299	-	871174922827439	44.247.241.200	44,553	192.168.1.21	847	Escaneo con Deny detectado
1 Sep 3, 2024 @ 19:30:48.191	-	29323434788861	190.234.247.113	44,555	192.168.1.21	117	Escaneo SYN detectado
1 Sep 3, 2024 @ 19:30:49.792	-	1432889527735099	44.247.241.200	44,553	192.168.1.21	865	Escaneo con Deny detectado
1 Sep 3, 2024 @ 19:30:48.179	-	218957792999919	44.247.241.200	44,553	192.168.1.21	845	Escaneo con Deny detectado
1 Sep 3, 2024 @ 19:30:47.776	-	164047393450723	190.234.247.113	44,555	192.168.1.21	324	Escaneo SYN detectado
1 Sep 3, 2024 @ 19:30:34.947	-	74058342044915	44.247.241.200	44,555	192.168.1.21	306	Escaneo con Deny detectado
1 Sep 3, 2024 @ 19:30:28.936	-	120273763867718	44.247.241.200	44,555	192.168.1.21	385	Escaneo con Deny detectado
1 Sep 3, 2024 @ 19:30:27.055	-	943870777931225	190.234.247.113	44,555	192.168.1.21	218	Escaneo SYN detectado
1 Sep 3, 2024 @ 19:30:23.051	-	2678889541574476	44.247.241.200	44,588	192.168.1.21	21	Escaneo con Deny detectado
1 Sep 3, 2024 @ 19:30:17.744	-	38419229447181	44.247.241.200	44,553	192.168.1.21	398	Escaneo con Deny detectado
1 Sep 3, 2024 @ 19:30:16.942	-	188454138991241	190.234.247.113	44,555	192.168.1.21	404	Escaneo SYN detectado
1 Sep 3, 2024 @ 19:30:13.329	-	143089162803485	190.234.247.113	44,543	192.168.1.21	426	Escaneo de puertos múltiple
1 Sep 3, 2024 @ 19:30:13.333	-	144376467282348	44.247.241.200	44,553	192.168.1.21	426	Escaneo de puertos múltiple
1 Sep 3, 2024 @ 19:30:13.325	-	144240773182276	207.249.145.32	44,553	192.168.1.21	426	Escaneo de puertos múltiple
1 Sep 3, 2024 @ 19:30:13.324	-	1432827768711184	111.237.8.227	44,553	192.168.1.21	426	Escaneo de puertos múltiple
1 Sep 3, 2024 @ 19:30:13.324	-	143444819886476	176.97.174.73	44,553	192.168.1.21	426	Escaneo de puertos múltiple
1 Sep 3, 2024 @ 19:30:13.323	-	143161469342387	144.28.185.124	44,553	192.168.1.21	426	Escaneo de puertos múltiple
1 Sep 3, 2024 @ 19:30:13.323	-	143419034844983	192.168.28.18	44,553	192.168.1.21	426	Escaneo de puertos múltiple

Figura 17. Captura de Kibana del escaneo de puertos Nmap -sS -D RND:10 -f -T2 -p 1-1024 192.168.1.21

Como se puede ver, tras el sondeo, las reglas se han activado correctamente. Y se puede observar como se activa desde los distintos tipos de dirección origen.

Véase el resultado desde la máquina atacante.


```
SYN Stealth Scan Timing: About 99.90% done; ETC: 20:19 (0:00:02 remaining)
Completed SYN Stealth Scan at 20:19, 2158.91s elapsed (1024 total ports)
Nmap scan report for 192.168.1.21
Host is up, received echo-reply ttl 28 (0.0031s latency).
Scanned at 2024-09-02 19:43:40 CEST for 2159s
Not shown: 1023 filtered tcp ports (no-response)
PORT      STATE      SERVICE REASON
22/tcp    closed    ssh      reset ttl 28

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 2172.44 seconds
Raw packets sent: 24772 (1.090MB) | Rcvd: 333 (18.604KB)
```

Figura 18. Captura de Wireshark del escaneo de puertos Nmap -sS -D RND:10 -f -T2 -p 1-1024 192.168.1.21

De nuevo se puede considerar la medida de prevención todo un éxito, pues Nmap no ha sido capaz de descubrir que el puerto 102 está abierto. Además, cabe destacar que ha tardado 5 veces más en efectuar el escaneo, de 424 segundos a 2172 segundos. Esto último es importante pues dificulta el trabajo del atacante al hacerle gastar más tiempo.

6.2 Ataque DoS

Probablemente el tipo de ataque más conocido. El ataque de denegación de servicio tiene como objetivo hacer una caída del sistema, es decir, mediante peticiones masivas con ciertas características, se hace colapsar ya sea la aplicación, la máquina o incluso la red. [50]

Sin embargo, hacer colapsar un sistema es relativamente extraño debido a su dificultad y que están preparados para soportar este tipo de ataques. Generalmente se puede dar como efectivo el ataque en el momento que consigue ralentizar la capacidad de procesamiento de las peticiones reales. [50]

Este tipo de ataque es muy utilizado en el sector de las organizaciones y empresas. Porque estos proveen un servicio y la finalidad es sencillamente bloquear su capacidad para proveerlo. De hecho, Incibe lo cataloga como uno de los cuatro ataques más frecuentes en el sector industrial en 2023 [50][59]

Un ejemplo de los perjuicios que ha llegado a causar un ataque de esta variante es en 2016 hacia la empres Dyn, proveedor DNS, que tras recibir un ataque DoS se estimaron las pérdidas en 110 millones de dólares. [60]

Este ataque ya ha demostrado tener la capacidad de provocar el fallo en PLCs Siemens Simatic S7-1500, presentes en la red industrial propuesta, con una versión entre la 2.9.2 y la 2.9.4, entre otros PLCs de la lista. Esto lo lleva a cabo enviando paquetes TCP al puerto 102 preparados con una carga útil que provoca la caída del PLC. La única forma de conseguir restablecer el PLC es mediante un reinicio. Por cuestiones de seguridad, es conocido el exploit pero no es público la forma de poder ejecutarlo, no obstante es relevante tenerlo en cuenta para poder hacer una simulación similar en el trabajo. [58]

Para llevar a cabo este ataque, usaremos la herramienta hPing3. Esta herramienta es conocida por su sencillez de uso (desde terminal) y sus altas capacidades. [51]

Concretamente se desarrollará un ataque sobre el puerto 102 de uno de los PLCs de la siguiente forma:

➤ `sudo hping3 --flood -S -p 102 192.168.1.21`

En este caso con la bandera --flood se le indica a hPing3 es que envíe paquetes a la máxima velocidad posible.

Con -S se establece que se envíen paquetes del tipo TCP SYN. Estos paquetes para DoS son efectivos debido a que piden abrir conexiones.

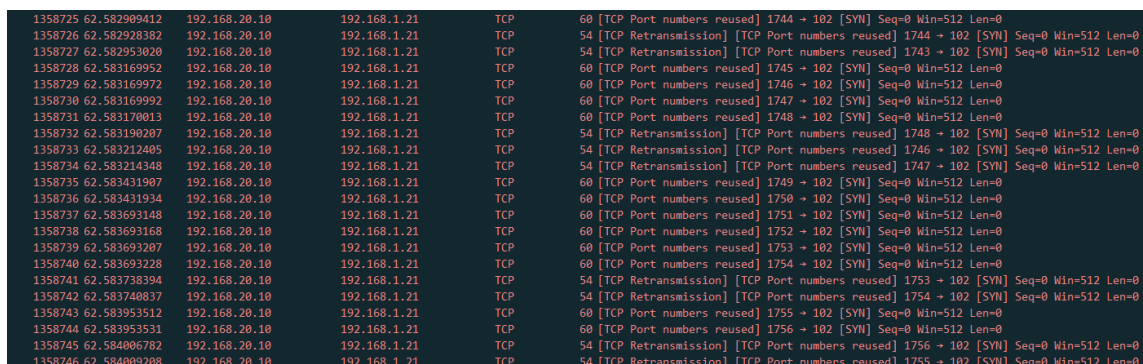
Con -p 102 se le indica que el ataque vaya dirigido al puerto 102.

Y por último la dirección IP de la víctima, 192.168.1.21.

6.2.1 Resultados del ataque DoS.

Por parte de la terminal atacante no hay nada interesante a señalar, simplemente la propia ejecución del comando.

Sin embargo, si podemos analizar las capacidades que ha tenido el ataque desde la captura de Wireshark hecha en el servidor Ubuntu.

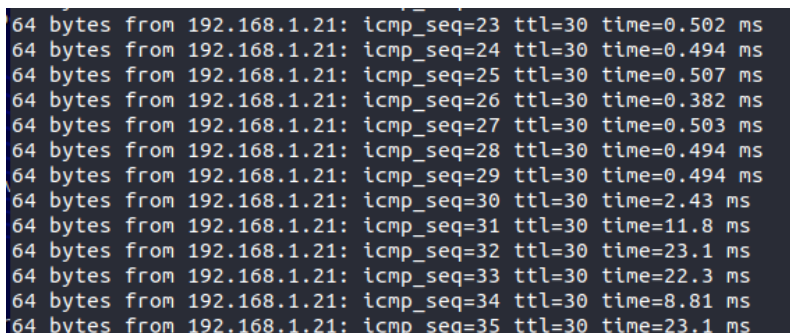


No.	Time	Source	Destination	Protocol	Length	Info
1358725	62.582999412	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1744 → 102 [SYN] Seq=0 Win=512 Len=0
1358726	62.582928382	192.168.20.10	192.168.1.21	TCP	54	[TCP Retransmission] [TCP Port numbers reused] 1744 → 102 [SYN] Seq=0 Win=512 Len=0
1358727	62.582953020	192.168.20.10	192.168.1.21	TCP	54	[TCP Retransmission] [TCP Port numbers reused] 1743 → 102 [SYN] Seq=0 Win=512 Len=0
1358728	62.583169952	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1745 → 102 [SYN] Seq=0 Win=512 Len=0
1358729	62.583169972	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1746 → 102 [SYN] Seq=0 Win=512 Len=0
1358730	62.583169992	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1747 → 102 [SYN] Seq=0 Win=512 Len=0
1358731	62.583170013	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1748 → 102 [SYN] Seq=0 Win=512 Len=0
1358732	62.583190207	192.168.20.10	192.168.1.21	TCP	54	[TCP Retransmission] [TCP Port numbers reused] 1748 → 102 [SYN] Seq=0 Win=512 Len=0
1358733	62.583212405	192.168.20.10	192.168.1.21	TCP	54	[TCP Retransmission] [TCP Port numbers reused] 1746 → 102 [SYN] Seq=0 Win=512 Len=0
1358734	62.583214348	192.168.20.10	192.168.1.21	TCP	54	[TCP Retransmission] [TCP Port numbers reused] 1747 → 102 [SYN] Seq=0 Win=512 Len=0
1358735	62.583431907	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1749 → 102 [SYN] Seq=0 Win=512 Len=0
1358736	62.583431934	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1750 → 102 [SYN] Seq=0 Win=512 Len=0
1358737	62.583693148	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1751 → 102 [SYN] Seq=0 Win=512 Len=0
1358738	62.583693168	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1752 → 102 [SYN] Seq=0 Win=512 Len=0
1358739	62.583693207	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1753 → 102 [SYN] Seq=0 Win=512 Len=0
1358740	62.583693228	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1754 → 102 [SYN] Seq=0 Win=512 Len=0
1358741	62.583738394	192.168.20.10	192.168.1.21	TCP	54	[TCP Retransmission] [TCP Port numbers reused] 1753 → 102 [SYN] Seq=0 Win=512 Len=0
1358742	62.583740837	192.168.20.10	192.168.1.21	TCP	54	[TCP Retransmission] [TCP Port numbers reused] 1754 → 102 [SYN] Seq=0 Win=512 Len=0
1358743	62.583953512	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1755 → 102 [SYN] Seq=0 Win=512 Len=0
1358744	62.583953531	192.168.20.10	192.168.1.21	TCP	60	[TCP Port numbers reused] 1756 → 102 [SYN] Seq=0 Win=512 Len=0
1358745	62.584006782	192.168.20.10	192.168.1.21	TCP	54	[TCP Retransmission] [TCP Port numbers reused] 1756 → 102 [SYN] Seq=0 Win=512 Len=0
1358746	62.584009208	192.168.20.10	192.168.1.21	TCP	54	[TCP Retransmission] [TCP Port numbers reused] 1755 → 102 [SYN] Seq=0 Win=512 Len=0

Figura 19. Captura de Wireshark del ataque hping3 --flood -S -p 102 192.168.1.21

Se puede observar que en el segundo 62 del ataque, hay ya 1.358.740 paquetes procesados. Sin duda es una cantidad grande pero no lo suficiente para bloquear el sistema.

No obstante, sí ha sido una tasa lo suficientemente rápida para una notoria ralentización en la capacidad de respuesta del PLC Siemens. Como se puede ver a continuación con una prueba sencilla en la que se le envían Pings y tras varios envíos se pone en marcha el ataque DoS.



```

64 bytes from 192.168.1.21: icmp_seq=23 ttl=30 time=0.502 ms
64 bytes from 192.168.1.21: icmp_seq=24 ttl=30 time=0.494 ms
64 bytes from 192.168.1.21: icmp_seq=25 ttl=30 time=0.507 ms
64 bytes from 192.168.1.21: icmp_seq=26 ttl=30 time=0.382 ms
64 bytes from 192.168.1.21: icmp_seq=27 ttl=30 time=0.503 ms
64 bytes from 192.168.1.21: icmp_seq=28 ttl=30 time=0.494 ms
64 bytes from 192.168.1.21: icmp_seq=29 ttl=30 time=0.494 ms
64 bytes from 192.168.1.21: icmp_seq=30 ttl=30 time=2.43 ms
64 bytes from 192.168.1.21: icmp_seq=31 ttl=30 time=11.8 ms
64 bytes from 192.168.1.21: icmp_seq=32 ttl=30 time=23.1 ms
64 bytes from 192.168.1.21: icmp_seq=33 ttl=30 time=22.3 ms
64 bytes from 192.168.1.21: icmp_seq=34 ttl=30 time=8.81 ms
64 bytes from 192.168.1.21: icmp_seq=35 ttl=30 time=23.1 ms

```

Figura 20. Captura del resultado de los pings mientras sucede el ataque DoS

Se puede apreciar claramente como se pasa de una media de 0.5ms de respuesta a una respuesta mucho más irregular llegando a ser 46 veces más alta como en el caso de la respuesta de 23.1ms.

Sin duda alguna, en un caso peor como un DDoS, donde hay más máquinas atacantes implicadas de forma distribuida, se podrían obtener resultados aún peores para la víctima.

6.2.2 Cómo proteger a la subred de un ataque DoS. [44]

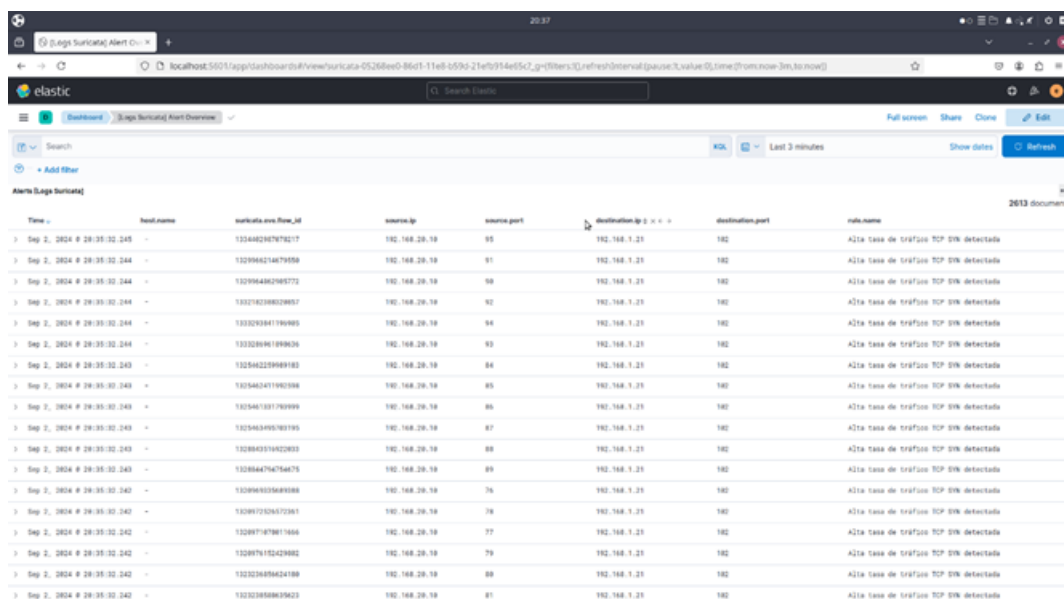
La regla para prevenir un ataque DoS es la siguiente:

- drop tcp any any -> 192.168.1.21 102 (msg:"Alta tasa de tráfico TCP SYN detectada"; flags:S; threshold:type limit, track by_dst, count 200, seconds 1; sid:1;)

Esta regla se encarga de detectar si está habiendo un tráfico de paquetes TCP anormalmente alto dirigido estratégicamente al puerto 102, a sabiendas de que es un puerto de comunicación del PLC. Se ha establecido que una tasa alta son 200 paquetes por segundo. Lógicamente se puede variar la sensibilidad.

La razón por la que solo hay una regla es porque realmente el ataque es relativamente sencillo al enviar el mismo tipo de paquete pero de forma masiva.

El resultado, como se puede ver en Kibana, ha sido exitoso:



Time	host.name	suricata.ana_flow_id	source.ip	source.port	destination.ip	destination.port	rule.name
Sep 2, 2024 # 20:35:32.245	-	1326423979217	192.168.20.10	95	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.244	-	13299621467958	192.168.20.10	91	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.244	-	132994482985772	192.168.20.10	88	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.244	-	13274238829857	192.168.20.10	92	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.244	-	13329284179085	192.168.20.10	94	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.244	-	13328996189926	192.168.20.10	93	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.243	-	132546225988183	192.168.20.10	84	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.243	-	132546247190288	192.168.20.10	85	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.243	-	132546133176999	192.168.20.10	86	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.243	-	1325463492163195	192.168.20.10	87	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.243	-	132884331822833	192.168.20.10	88	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.243	-	13288447876875	192.168.20.10	89	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.242	-	132984935689188	192.168.20.10	76	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.242	-	132887326872361	192.168.20.10	78	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.242	-	132887187861866	192.168.20.10	77	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.242	-	132887815242982	192.168.20.10	79	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.242	-	132328888424188	192.168.20.10	80	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada
Sep 2, 2024 # 20:35:32.242	-	132328888429823	192.168.20.10	81	192.168.1.23	102	Alerta tasa de tráfico TCP SYN detectada

Figura 21. Captura de Kibana del ataque DoS

Pese a que la regla se activaba, cabe destacar que desde Suricata no es posible ejecutar un bloqueo directo del atacante, simplemente se desecha el paquete una vez se detecta que cumple la regla. Esto quiere decir que se ha reducido la capacidad del ataque DoS pero no se ha bloqueado.

6.3 Ataque del tipo ICMP Redirect (Man In The Middle).

Este tipo de ataque tiene la finalidad de establecer al atacante como la puerta de enlace predeterminada sobre la víctima. [52]

Situando al atacante en dicha posición, es capaz de ejecutar ataques del tipo Man In The Middle. Se trata de un ataque dónde el cibercriminal al situarse en el medio de una conexión entre dos dispositivos es capaz de interceptar el tráfico y analizarlo. Este ataque es realmente peligroso ya que con las técnicas de descifrado necesarias es posible averiguar credenciales o analizar el tráfico con tal de poder recrearlo en el momento necesario. [61]

Esto lo intenta mediante el envío del paquete ICMP Redirect. Este paquete se envía típicamente desde un router que ha encontrado una ruta más óptima para llegar al destino desde un host. Se le envía a dicho host y así este establece una nueva ruta pasando por el atacante. [53]

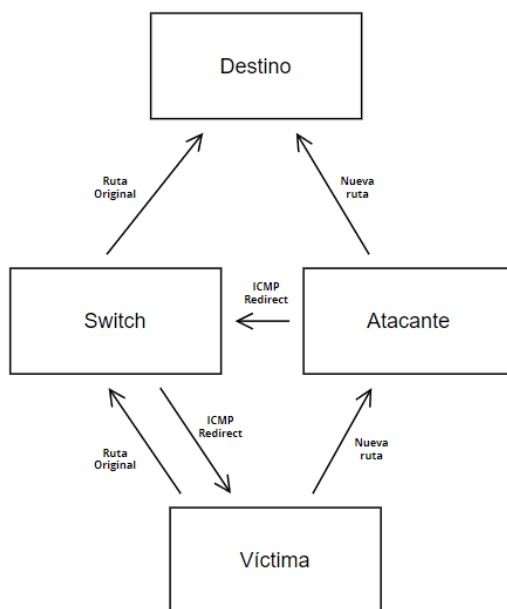


Figura 22. Diagrama de funcionamiento del ataque ICMP Redirect [53]

En este caso se hace de nuevo uso de la herramienta hPing3. En este caso, el comando será: [51]

- `hping3 --icmp --icmptype 5 --icmpcode 1 -I ens33 -a 192.168.1.1 -d 46 -E redirect_packet.bin 192.168.1.21`

Claramente es un comando con múltiples opciones. Analizándolas una a una:

Las opciones `--icmp`, `--icmptype` e `--icmpcode` sirven para poder enviar paquetes del tipo ICMP, concretamente en este caso del Tipo 5, que es el tipo Redirect. Y con el código 1 quiere decir que el redireccionamiento es para la red completa.

La opción `-I` es para indicar a continuación la interfaz de red por dónde debe salir el paquete. Depende del host desde el que lanzas el comando.

La bandera `-a` sirve para suplantar una dirección IP, en este caso se suplanta la dirección IP del router Cisco, 192.168.1.1.

Con `-d 46` se especifica la longitud de la carga útil del paquete. Con tal de que parezca un paquete normal.

La bandera `-E` con el valor `redirect_packet.bin` es básicamente un archivo de datos que se envía, se puede enviar con valores nulos.

Por último, la dirección IP de la víctima.

6.3.1 Resultado del ataque ICMP Redirect

Desde la máquina Ubuntu, se puede observar claramente en la captura Wireshark como se envían los paquetes ICMP Redirect.

93	1.784156723	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
94	1.784277235	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
145	2.785132217	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
146	2.785182937	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
199	3.786293893	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
200	3.786357346	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
251	4.787533990	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
252	4.787672373	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
303	5.788655522	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
304	5.788758520	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
357	6.790018144	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
358	6.790157973	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
409	7.791926669	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
410	7.792046563	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
462	8.792744797	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
463	8.792835579	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
514	9.793454298	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
515	9.793595866	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
566	10.794434708	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
567	10.794571963	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)
618	11.795519831	192.168.1.1	192.168.1.21	ICMP	116 Redirect	(Redirect for host)

Figura 23. Captura de Wireshark del ataque ICMP Redirect

Obsérvese como hay una gran cantidad de paquetes del tipo ICMP Redirect y cómo se indica el Redirect for host.

6.3.2 *Cómo proteger a la subred de un ataque ICMP Redirect [44]*

En este caso se hará uso de 2 reglas claramente diferenciadas pero que detectan este ataque:

- `drop icmp any any -> 192.168.1.21 any (msg:"ICMP Redirect Spoofing Detectado - TTL anómalo"; itype:5; ttl:<=62; sid:1000040; rev:1;)`

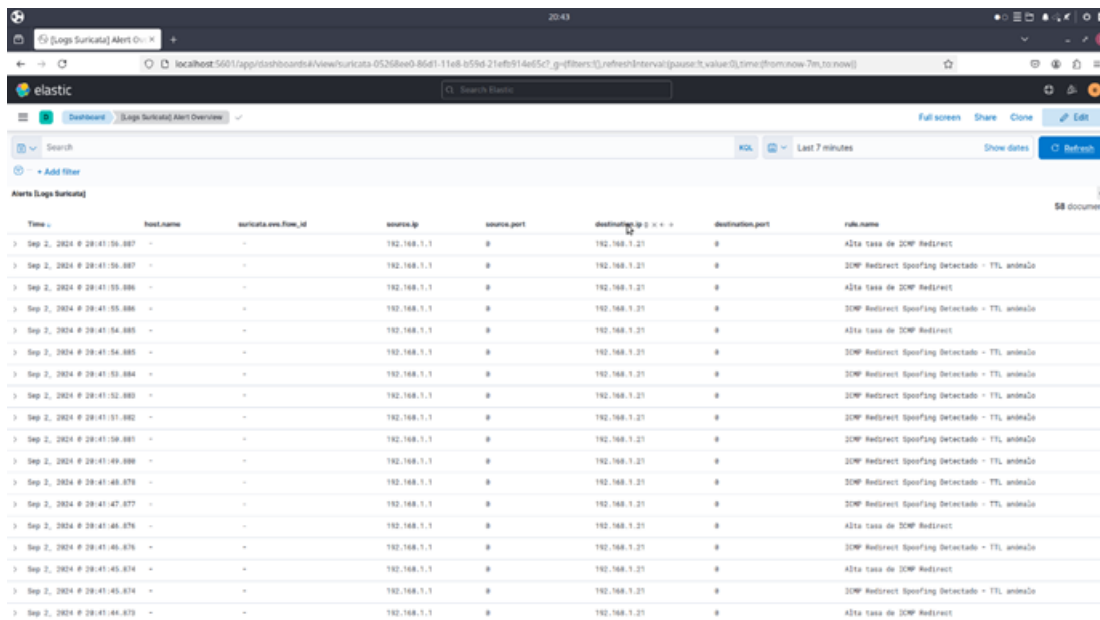
Esta regla requiere de tener conocimiento de cómo está estructurada la red pues podemos comprender que si llega un paquete con un TTL que está debajo de lo que debería contando desde el router hacia el Ubuntu, es una suplantación.

Por ello se especifica en la regla que cualquier ICMP del tipo 5 (Redirect) con un TTL de 62 o menor debe de ser desde otra máquina.

- `drop icmp any any -> 192.168.1.0/24 any (msg:"Alta tasa de ICMP Redirect"; itype:5; threshold:type limit, track by_dst, count 3, seconds 10; sid:1000032; rev:1;)`

Luego por otro lado también es posible detectar si se está bajo un ataque ICMP Redirect en el caso de que haya una tasa de paquetes de este tipo relativamente alta, como pueden ser 3 cada 10 segundos.

A continuación se ve en Kibana el resultado de aplicar estas reglas:



Time	host.name	suricata.event_id	source.ip	source.port	destination.ip	destination.port	rule.name
2024-08-28T09:41:50.887	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:50.887	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo
2024-08-28T09:41:50.888	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:50.888	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo
2024-08-28T09:41:50.889	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:50.889	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo
2024-08-28T09:41:50.890	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:50.890	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo
2024-08-28T09:41:50.891	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:50.891	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo
2024-08-28T09:41:49.889	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:49.889	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo
2024-08-28T09:41:48.879	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:48.879	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo
2024-08-28T09:41:47.877	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:47.877	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo
2024-08-28T09:41:46.876	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:46.876	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo
2024-08-28T09:41:46.876	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:46.876	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo
2024-08-28T09:41:46.879	-	-	192.168.1.1	0	192.168.1.21	0	Alta tasa de ICMP Redirect
2024-08-28T09:41:46.879	-	-	192.168.1.1	0	192.168.1.21	0	ICMP Redirect Spoofing Detectado - TTL anómalo

Figura 24. Captura de Kibana tras el ataque ICMP Redirect

Se puede concluir que la detección y prevención del ataque ha sido un éxito debido a que tras un análisis desde la máquina atacante, se corrobora que no está pasando el tráfico del PLC por la máquina.

6.4 Ataque de fuerza bruta con diccionario sobre la conexión SSH / Telnet

Con este ataque la intención es averiguar la contraseña de un usuario haciendo uso de un listado de posibles contraseñas. Cobra sentido en este contexto al efectuarlo sobre dispositivos que gestionen la red. [54][55]

El ataque híbrido de fuerza bruta y diccionario es uno de los ataques principales destinados al robo de credenciales. [62]

Este ataque también está situado como uno de los 4 tipos de ataque más repetidos en el sector industrial en 2023. [59]

En este caso, que el Switch Siemens da gestión a los equipos, sería un problema que alguien de forma remota pudiera conectarse a él y bajar los puertos dónde hay conectados PLCs.

Para poder ejecutar el ataque, en esta ocasión se hará uso de la herramienta Hydra. Esta es una herramienta conocida para ejercer ataques de fuerza bruta. [56]

El comando a ejecutar será el siguiente: [56]

- hydra -l admin -P contraseñas.txt 192.168.1.20 telnet

Básicamente se intentará entrar al switch Siemens que se vio que tenía el puerto Telnet abierto.

Con el comando -l se indica con qué nombre de usuario se hace el intento.

Con el comando -P habilita cargar un archivo de contraseñas. Para esta prueba de laboratorio, se han cargado 100 contraseñas.

6.4.1 Resultados del ataque de fuerza bruta con diccionario.

Véase el resultado del ataque con 100 contraseñas almacenadas, ninguna de ellas correcta.


```
[*]-[root@parrot]-[/home/valentino/Desktop]
#hydra -l admin -P /usr/share/wordlists/contrasenyas.txt 192.168.1.20 telnet
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations,
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-08-31 20:34:41
[WARNING] telnet is by its nature unreliable to analyze, if possible better choose FTP, SSH, etc. if available
[DATA] max 16 tasks per 1 server, overall 16 tasks, 100 login tries (1:1/p:100), ~7 tries per task
[DATA] attacking telnet://192.168.1.20:23/
[STATUS] 100.00 tries/min, 100 tries in 00:01h, 1 to do in 00:01h, 13 active
1 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-08-31 20:35:52
```

Figura 25. Resultado del ataque de fuerza bruta desde la máquina atacante

Como se puede observar en el resultado, se ha hecho un total de 100 intentos a una tasa de 100 intentos por minuto.

De los ataques realizados este es el único que depende directamente de que el administrador de redes haya establecido una contraseña lo suficientemente robusta para que este tipo de ataque no sea efectivo.

6.4.2 *Cómo proteger al switch industrial de un ataque de fuerza bruta por diccionario. [44]*

Las reglas para paliar este ataque son sencillas:

- drop tcp any any -> 192.168.1.20 23 (msg:"Ataque de fuerza bruta detectado - Alta tasa de conexiones TELNET"; threshold:type both, track by_src, count 5, seconds 10; sid:1; rev:2;)

Se ha establecido una regla conociendo que el punto débil es el switch y concretamente su puerto 23, Telnet. Por tanto, se configura la regla para que si se detectan un alto número de paquetes Telnet, se active. En este caso se han puesto 5 paquetes en 10 segundos.

- drop tcp 192.168.1.20 23 -> any any (msg:"Ataque de fuerza bruta detectado - Intentos de autenticación fallidos"; flow:from_server,established; content:"incorrect"; nocase; threshold:type limit, track by_src, count 3, seconds 20; sid:2;)

En este caso se establece una alerta de que han habido intentos fallidos a la hora de conectarse via Telnet con el switch Siemens. Concretamente, se puede observar en las opciones como el Flow es desde el servidor con conexiones establecidas y los paquetes deben contener “incorrect”. Además la alerta se genera si se cuentan 3 paquetes de este tipo en un plazo de 20 segundos.

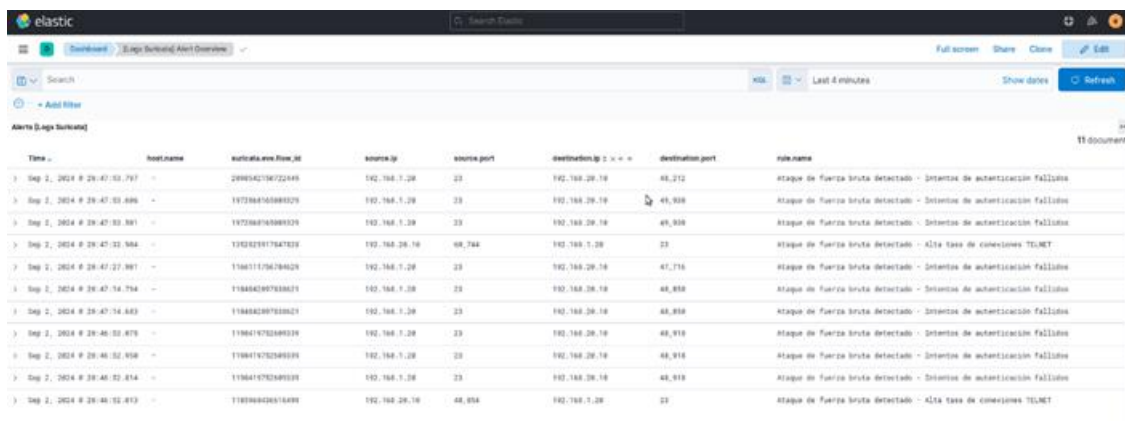
En este caso, en el archivo de contraseñas se ha introducido la correcta para probar el sistema.

Véase el resultado del ataque con las reglas de Suricata aplicadas:

```
[*]-[root@parrot]-[/home/valentino]
#hydra -l admin -P /usr/share/wordlists/contrasenyas.txt 192.168.1.20 telnet
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for ill
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-09-02 20:46:53
[WARNING] telnet is by its nature unreliable to analyze, if possible better choose FTP, SSH, etc. if available
[DATA] max 16 tasks per 1 server, overall 16 tasks, 98 login tries (1:1/p:98), ~7 tries per task
[DATA] attacking telnet://192.168.1.20:23/
[23][telnet] host: 192.168.1.20 login: admin password: admin
[STATUS] 98.00 tries/min, 98 tries in 00:01h, 1 to do in 00:01h, 6 active
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-09-02 20:48:04
```

Figura 26. Resultado del ataque de fuerza bruta desde la máquina atacante tras la activación de las reglas de Suricata

No ha sido posible evitar el ataque, sin embargo, en Kibana se observa se han intentado desechar los paquetes.



Time	host.name	suricata.ena.flow_id	source.ip	source.port	destination.ip	destination.port	rule.name
2024-08-29T13:47:53.797Z	192.168.1.20	289842150722449	192.168.1.20	23	192.168.28.18	44,212	Ataque de fuerza bruta detectado - Intentos de autenticación fallidos
2024-08-29T13:47:53.886Z	192.168.1.20	197366165889329	192.168.1.20	23	192.168.28.18	44,908	Ataque de fuerza bruta detectado - Intentos de autenticación fallidos
2024-08-29T13:47:53.981Z	192.168.1.20	197366165889329	192.168.1.20	23	192.168.28.18	44,908	Ataque de fuerza bruta detectado - Intentos de autenticación fallidos
2024-08-29T13:47:53.984Z	192.168.28.18	132829517847828	192.168.28.18	44,744	192.168.1.20	23	Ataque de fuerza bruta detectado - Alta tasa de conexiones TELNET
2024-08-29T13:47:57.981Z	192.168.1.20	1186311756784628	192.168.1.20	23	192.168.28.18	47,716	Ataque de fuerza bruta detectado - Intentos de autenticación fallidos
2024-08-29T13:47:58.794Z	192.168.1.20	1184848397638628	192.168.1.20	23	192.168.28.18	48,858	Ataque de fuerza bruta detectado - Intentos de autenticación fallidos
2024-08-29T13:47:58.883Z	192.168.1.20	1184848397638628	192.168.1.20	23	192.168.28.18	48,858	Ataque de fuerza bruta detectado - Intentos de autenticación fallidos
2024-08-29T13:48:03.478Z	192.168.1.20	1186416782889328	192.168.1.20	23	192.168.28.18	48,918	Ataque de fuerza bruta detectado - Intentos de autenticación fallidos
2024-08-29T13:48:03.938Z	192.168.1.20	1186416782889328	192.168.1.20	23	192.168.28.18	48,918	Ataque de fuerza bruta detectado - Intentos de autenticación fallidos
2024-08-29T13:48:03.814Z	192.168.1.20	1186416782889328	192.168.1.20	23	192.168.28.18	48,918	Ataque de fuerza bruta detectado - Intentos de autenticación fallidos
2024-08-29T13:48:03.813Z	192.168.28.18	11898862816488	192.168.28.18	44,854	192.168.1.20	23	Ataque de fuerza bruta detectado - Alta tasa de conexiones TELNET

Figura 27. Captura de Kibana tras el ataque de fuerza bruta

A pesar de que no haya funcionado como se esperaba en cuanto a resultado final, sí se han activado las alertas del ataque y el intento de paliarlo.

Podría parecer un intento fallido, pero para paliar este tipo de ataques desde Suricata solo se podría con unas reglas muy concretas que se activarán también cuando se efectúa de una forma muy concreta el ataque. Es por ello que este es uno de los casos donde es relevante combinar la herramienta de Suricata con otras herramientas de ciberseguridad.

Capítulo 7. Conclusiones de la implementación de un IDPS en un entorno de red industrial

Se puede considerar más que justificada la necesidad de un sistema de esta categoría en una red industrial.

Se ha constatado como en los escaneos de Nmap se han obtenido peores resultados y el ataque de tipo ICMP Redirect ha sido fácilmente mitigado.

Sin embargo, el ataque de tipo DoS pese a haber sido paliado, no ha sido totalmente contrarrestado. Y en el caso del ataque de fuerza bruta con una contraseña correcta en el diccionario también.

Se ha podido constatar la importancia de entender la red que se protege y a raíz de ello poder calcular qué tipo de paquetes pueden ser potencialmente maliciosos. Cabe recalcar que, sin dichas observaciones sobre la red, la aplicación de Suricata en un entorno de red industrial tiene claramente menos capacidades más allá de poder bloquear ataques muy genéricos.

Entre los casos analizados, cabe señalar el ataque de ICMP Redirect. En el cual se calculó el TTL real desde el router y se comparaba con los paquetes que llegaban suplantando dicho router. Se ha comprobado lo efectiva que es la regla aplicando esta lógica, llegando a reducir por completo el ataque del ICMP Redirect.

También resaltar el entendimiento del funcionamiento de los dispositivos vulnerables y sus características. Un ejemplo de ello es formar reglas específicamente en el puerto 102 de los PLCs teniendo en cuenta que este puede llegar a ser descubierto en un escaneo Nmap y sea vulnerable a ataques de tipo DoS.

Otro ejemplo es conocer en detalle el tráfico de paquetes en la red. En este caso se sabía (tras captura de Wireshark) que el tráfico entre un PC y un PLC a través del TIA Portal no conlleva paquetes fragmentados, por ello se podría sospechar o establecer una alerta si se detectan paquetes fragmentados porque tal como se ha podido observar, podrían ser parte de una técnica de evasión IDS.

Es por los motivos mencionados, que antes de establecer reglas en el sistema IDPS, primero se debe entender el contexto de la red a proteger.

Por otro lado, cabe destacar que las capacidades que ha demostrado Suricata con una configuración sobre la red existente tan mínima como simplemente la aplicación de rutas estáticas desde el router Cisco 1921 han sido realmente satisfactorias.

Abre la veda a la posibilidad de que, en una red industrial completa ya formada, si se requiere la implementación de un IDPS, se pueda implementar de forma realmente sencilla en términos técnicos. No hay que apagar todos los dispositivos con tal de cambiar la topología ni similar, simplemente con tener a los PLCs en paralelo o debajo (topológicamente hablando) es suficiente.

Además de su fácil implementación, cabe destacar las capacidades que tiene Suricata para poder filtrar los paquetes. Esto es relevante en un ejemplo hipotético como el siguiente:

En el caso de tener un PLC y ser consciente de que recientemente ha surgido una nueva vulnerabilidad y se han obtenido los datos para comprender como funciona dicha vulnerabilidad. Suponiendo que no se pudieran actualizar el PLC por cuestiones de producción o capital humano, por ejemplo, es posible configurar Suricata para que establezca una serie de reglas con la intención de mitigar un ataque utilizando dicha vulnerabilidad.

Verdaderamente Suricata ha demostrado ser una opción claramente válida para instalarse en una red industrial.



7.1 Futuras líneas de acción

Suricata hace función de filtrado de paquetes. Pero tal como se ha visto, en ocasiones los paquetes no cumplen un umbral. O por el lado contrario se puede incluso establecer un umbral tan sensible que pudiera perjudicar el tráfico normal de la red.

Por ello debe considerarse la implementación conjunta de otras medidas de seguridad.

En el contexto de este TFG, sin lugar a duda, se pueden combinar las reglas de Suricata con la funcionalidad de las IPTables del Ubuntu. De esta forma se puede hacer funcionar un servicio personalizado que al detectar una cantidad de alertas de Suricata haga activar una regla de IPTable para, por ejemplo, bloquear esa IP de forma temporal.

Con una medida como la anterior, el resultado del ataque de fuerza bruta sería fallido en caso de que la contraseña correcta no estuviera en las primeras posiciones. Debido a que la configuración del servicio personalizado al detectar las alertas de intentos fallidos de Telnet podría bloquear la comunicación mediante este canal con el switch Siemens.

Por otro lado, también se mitigaría por completo el ataque DoS. De forma análoga al anterior, en este caso se podría configurar para que, al generarse cierta cantidad de alertas, bloquee directamente los paquetes que provengan de la dirección maliciosa.

Concluyendo, la combinación de Suricata con herramientas de ciberseguridad tales como Firewalls, daría lugar a mejores resultados en materia de seguridad. Y, desde esta perspectiva, los siguientes pasos en un trabajo como éste, sería la implementación sugerida.

Capítulo 8. Bibliografía

- [1] Check Point Research, “Check Point Research Reports a 38% Increase in 2022 Global Cyberattacks” <https://blog.checkpoint.com/2023/01/05/38-increase-in-2022-global-cyberattacks/> [Online].
- [2] SecureList by Kaspersky, “Threat landscape for industrial automation systems. Statistics for H1 2023” <https://securelist.com/threat-landscape-for-industrial-automation-systems-statistics-for-h1-2023/110605/> [Online].
- [3] El confidencial, “El primer timo hecho por una persona falsa generada por IA en una videoconferencia” https://www.elconfidencial.com/tecnologia/novaceno/2023-05-25/estafa-china-videollamada-inteligencia-artificial_3636801/ [Online]
- [4] OWL Cyber Defense, “Air-Gapped Networks and Data Diodes” <https://owlcyberdefense.com/blog/air-gapped-networks-and-data-diodes/> [Online]
- [5] IBM, “What is an intrusion detection system (IDS)?” <https://www.ibm.com/topics/intrusion-detection-system> [Online]
- [6] IBM, “What is an intrusion prevention system (IPS)?” <https://www.ibm.com/topics/intrusion-prevention-system> [Online]
- [7] Incibe, “¿Qué son y para qué sirven los SIEM, IDS e IPS?” <https://www.incibe.es/empresas/blog/son-y-sirven-los-siem-ids-e-ips> [Online]
- [8] OSSEC, “Documentation” <https://www.ossec.net/docs/> [Online]
- [9] Zeek, “Documentation” <https://docs.zeek.org/en/v6.2.0/about.html> [Online]
- [10] Cisco, “Deploy Snort IPS on Cisco Integrated Services Routers 4000 Series” <https://www.cisco.com/c/en/us/support/docs/security/ios-intrusion-prevention-system-ips/220565-deploy-snort-ips-on-cisco-integrated-ser.html> [Online]
- [11] Snort, “Snort page” <https://www.snort.org/> [Online]
- [12] Suricata, “Suricata’s page” <https://suricata.io/> [Online]
- [13] Check Point, “Sistema de detección de intrusos (IDS)” <https://www.checkpoint.com/es/cyber-hub/network-security/what-is-an-intrusion-detection-system-ids/> [Online]
- [14] Incibe, “Diseño y Configuración de IPS, IDS y SIEM en Sistemas de Control Industrial” https://www.incibe.es/sites/default/files/contenidos/guias/doc/certsi_diseno_configuracion_ips_ids_siem_en_sci.pdf [Online]
- [15] Security Art Work, “Evasión en IDS” <https://www.securityartwork.es/2010/06/21/evasion-en-ids-i/> [Online]
- [16] Yang L, Shami A, “IDS-ML: An open source code for Intrusion Detection System development using Machine Learning” <https://www.sciencedirect.com/science/article/pii/S266596382200130> [Online]
- [17] Waleed A., Fareed A., Masood A., “Which open-source IDS? Snort, Suricata or Zeek” <https://www.sciencedirect.com/science/article/pii/S1389128622002420> [Online]
- [18] GeeksForGeeks, “Difference between HIDS and NIDS” <https://www.geeksforgeeks.org/difference-between-hids-and-nids/> [Online]
- [19] Redes y Energías, “¿Qué es un Sistema SCADA?” <https://redesyenergias.com/ingenieria-electrica/que-es-un-sistema-scada/> [Online]

- [20] Aula 21, “Qué es un HMI: para qué sirve la Interfaz Humano-Máquina” <https://www.cursosaula21.com/que-es-un-hmi/> [Online]
- [21] Itztli, “Diferencia entre PLC y RTU” <https://www.itztli.es/diferencia-entre-plc-y-rtu/> [Online]
- [22] Snort, “Advisories” <https://www.snort.org/advisories> [Online]
- [23] Snort, “Documents” <https://snort.org/documents/registered-vs-subscriber> [Online]
- [24] Suricata, “Suricata update” <https://docs.suricata.io/en/latest/rule-management/suricata-update.html> [Online]
- [25] Snort, “Tweaks Scripts” https://docs.snort.org/start/tweaks_scripts [Online]
- [26] Suricata, “Rule Lua Scripting” <https://docs.suricata.io/en/suricata-6.0.0/rules/rule-lua-scripting.html> [Online]
- [27] DigitalOcean, “How to build a SIEM with Suricata and Elastic Stack on Ubuntu” <https://www.digitalocean.com/community/tutorials/how-to-build-a-siem-with-suricata-and-elastic-stack-on-ubuntu-20-04> [Online]
- [28] IBM, “QRadar SIEM” <https://www.ibm.com/es-es/products/qradar-siem> [Online]
- [29] Splunk, “Splunk Enterprise Security” https://www.splunk.com/en_us/products/enterprise-security.html [Online]
- [30] IBM, “QRadar Pricing” <https://www.ibm.com/es-es/products/qradar-siem/pricing> [Online]
- [31] eSecurity Planet, “Splunk Enterprise Security Review: SIEM Product Features & Pricing” <https://www.esecurityplanet.com/products/splunk-enterprise-security-es/> [Online]
- [32] Cody Wilson, “Evaluating the Efficacy of Network Forensic Tools: A Comparative Analysis of Snort, Suricata, and Zeek in Addressing Cyber Vulnerabilities” <https://sansorg.egnyte.com/dl/11u6Zjhdgy> [Online]
- [33] Huawei, “¿Qué es el port mirroring (puerto espejo)?” <https://forum.huawei.com/enterprise/es/%C2%BFqu%C3%A9-es-el-port-mirroring-puerto-espejo/thread/667241351445626880-667212883219591168> [Online]
- [34] Cisco, “IP Route Commands” https://www.cisco.com/c/en/us/td/docs/routers/nfvis/switch_command/b-nfvis-switch-command-reference/ip_route_commands.pdf [Online]
- [35] Fortinet, “Qué es el protocolo de resolución de direcciones (ARP)?” <https://www.fortinet.com/lat/resources/cyberglossary/what-is-arp> [Online]
- [36] Cisco, “Comprender el protocolo de resolución de direcciones de proxy (ARP)” https://www.cisco.com/c/es_mx/support/docs/ip/dynamic-address-allocation-resolution/13718-5.html [Online]
- [37] Medium, “How to enable proxy arp on Linux” <https://medium.com/@mytalk123/how-to-enable-proxy-arp-on-linux-70dd8b22a07e> [Online]
- [38] Linux, “IP Neighbor” <http://linux-ip.net/html/tools-ip-neighbor.html> [Online]
- [39] Datadog, “Disable Kernel Parameter for Sending ICMP Redirects on all IPv4 Interfaces by Default” https://docs.datadoghq.com/security/default_rules/xccdf-org-ssgproject-content-rule-sysctl-net-ipv4-conf-default-send-redirects/ [Online]
- [40] BeHackerPro, “¿Qué es Iptables?” <https://behacker.pro/que-es-iptables-y-como-funciona/> [Online]

- [41] HackTricks, “Suricata & Iptables cheatsheet” <https://book.hacktricks.xyz/v/es/generic-methodologies-and-resources/basic-forensic-methodology/pcap-inspection/suricata-and-iptables-cheatsheet> [Online]
- [42] DigitalOcean, “How to list and delete Iptables firewall” <https://www.digitalocean.com/community/tutorials/how-to-list-and-delete-iptables-firewall-rules-es> [Online]
- [43] DigitalOcean, “How to configure Suricata as an Intrusion Prevention System (IPS) on Ubuntu 20.04” <https://www.digitalocean.com/community/tutorials/how-to-configure-suricata-as-an-intrusion-prevention-system-ips-on-ubuntu-20-04> [Online]
- [44] Suricata, “Documentation” <https://docs.suricata.io/en/latest/rules/index.html> [Online]
- [45] Cloudflare, “¿Qué es un puerto de ordenador? | Puertos en la red” <https://www.cloudflare.com/es-es/learning/network-layer/what-is-a-computer-port> [Online]
- [46] Fortinet, “¿Qué es un escaneo de puertos? Cómo prevenir ataques ¿Cómo prevenir los ataques de escaneo de puertos?” <https://www.fortinet.com/lat/resources/cyberglossary/what-is-port-scan> [Online]
- [47] Nmap, “Introducción al análisis de puertos” <https://nmap.org/man/es/man-port-scanning-basics.html> [Online]
- [48] Nmap, “Documentación” <https://nmap.org/book/man.html> [Online]
- [49] Qualys, “How does UDP port scanning and service detection work?” <https://success.qualys.com/support/s/article/000006121#:~:text=For%20UDP%20scanning%2C%20the%20service,the%20port%20is%20considered%20closed.> [Online]
- [50] Incibe, “¿Qué son los ataques DoS y DDoS?” <https://www.incibe.es/ciudadania/blog/que-son-los-ataques-dos-y-ddos> [Online]
- [51] Kali, “hPing3 Documentation” <https://www.kali.org/tools/hping3/> [Online]
- [52] Cicese, “Estos son todos los ataques a las redes que existen y cómo evitarlos” <https://seguridad.cicese.mx/alerta/1428/Estos-son-todos-los-ataques-a-las-redes-que-existen-y-c%C3%B3mo-evitarlos> [Online]
- [53] Medium, “ICMP Redirect Attack” <https://medium.com/@sherishrat/icmp-redirect-attack-18755cd0897> [Online]
- [54] Antimalwares, “Ataques de fuerza bruta y diccionario” <https://antimalwares.es/ataques-de-fuerza-bruta-y-diccionario> [Online]
- [55] Kaspersky, “¿Qué es un ataque de diccionario?” <https://latam.kaspersky.com/resource-center/definitions/what-is-a-dictionary-attack?srsltid=AfmBOoraNNyPhZRYLpP4oLUPXRqqtyGfIY9pXXAZYyrstAMYQnVbPzMk> [Online]
- [56] Kolibers, “Hydra – Herramienta de fuerza bruta” <https://www.kolibers.com/blog/hydra-herramienta-de-fuerza-bruta.html> [Online]
- [57] Speedguide, “Port 102 Details” <https://www.speedguide.net/port.php?port=102> [Online]
- [58] InfoSecMatter, “Siemens (CVE-2021-37185) - Nessus” <https://www.infosecmatter.com/nessus-plugin-library/?id=500615> [Online]
- [59] Incibe, “Las tendencias de ataque en el sector industrial durante 2023” <https://www.incibe.es/incibe-cert/blog/las-tendencias-de-ataque-en-el-sector-industrial-durante-2023> [Online]



[60] MetaCompliance, “Los 10 mayores ataques DDoS y cómo su organización puede aprender de ellos” <https://www.metacompliance.com/es/blog/cyber-security-awareness/10-biggest-ddos-attacks-and-how-your-organisation-can-learn-from-them> [Online]

[61] Incibe, “El ataque del “Man in the middle” en la empresa, riesgos y formas de evitarlo” <https://www.incibe.es/empresas/blog/el-ataque-del-man-middle-empresa-riesgos-y-formas-evitarlo> [Online]

[62] Techopedia, “50 Estadísticas Clave de Ciberseguridad para Septiembre de 2024” <https://www.techopedia.com/es/estadisticas-ciberseguridad> [Online]

Capítulo 9. Anexo de instalación de los softwares.

9.1 Instalación y configuración de Suricata en Ubuntu 20.04.

Para la descarga, instalación y configuración de Suricata se ha hecho servir la documentación oficial.

- sudo apt-get install software-properties-common
- sudo add-apt-repository ppa:oisf/suricata-stable
- sudo apt-get update

Con estos tres primeros comandos introducidos en orden lo que se consigue es agregar el repositorio de Suricata y actualizar la lista de paquetes disponible.

A continuación, se procede con la instalación de Suricata:

- sudo apt-get install suricata

Para asegurar que ha sido correctamente instalado se puede hacer uso del siguiente comando.

- whereis suricata

Lo cual nos dará una respuesta como la de la siguiente figura.

```
vcoralc@cibereguridad:~$ whereis suricata
suricata: /usr/bin/suricata /usr/lib/suricata /etc/suricata /usr/share/suricata
```

Figura 28. Resultado del comando whereis suricata

Una vez se tiene instalado Suricata se debe proceder a modificar determinados parámetros para que funcione correctamente en la máquina / red.

Primeramente, ha de conocerse en que interfaz de red del host va a estar conectada la red. Esto se puede averiguar con el comando:

- ip addr

En el caso de la máquina utilizada en este trabajo es:

```
vcoralc@cibereguridad:~$ ip addr
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: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether a4:bb:6d:5c:8e:88 brd ff:ff:ff:ff:ff:ff
    inet 158.42.188.65/24 brd 158.42.188.255 scope global dynamic eno1
        valid_lft 172383sec preferred_lft 172383sec
    inet6 2001:720:101c:188:a6bb:6dff:fe5c:8e88/64 scope global dynamic mngtmpaddr noprefixroute
        valid_lft 2591804sec preferred_lft 604604sec
    inet6 fe80:a6bb:6dff:fe5c:8e88/64 scope link
        valid_lft forever preferred_lft forever
vcoralc@cibereguridad:~$
```

Figura 29. Resultado del comando ip addr

Como se puede ver en la imagen, la primera interfaz es la de *loopback*. Y la segunda, en este caso eno1, es la que es necesario saber el nombre para proceder a la configuración.

A continuación se debe editar el archivo suricata.yaml ubicado en el caso de una máquina Ubuntu en /etc/suricata :

- nano /etc/suricata/suricata.yaml

En este caso se usa nano para editar el archivo, pero puede ser cualquier editor a preferencia del usuario.

Una vez dentro del archivo, hay que editar las líneas correspondientes al *af-packet* y la *interface* que hace uso. Tal como se muestra a continuación.

```
1: vcoralc@cibereguridad: ~
GNU nano 4.8 /etc/suricata/suricata.yaml Modified

##
## Step 3: Configure common capture settings
##
## See "Advanced Capture Options" below for more options, including Netmap
## and PF_RING.
##

# Linux high speed capture support
af-packet:
- interface: eno1
  # Number of receive threads. "auto" uses the number of cores
  #threads: auto
  # Default clusterid. AF_PACKET will load balance packets based on flow.
  cluster-id: 99
  # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or pe
  # This is only supported for Linux kernel > 3.1
  # possible value are:
  # * cluster_flow: all packets of a given flow are sent to the same socket
```

Figura 30. Captura del archivo *suricata.yaml* en la sección de interfaz

Por defecto el valor de *interface* es *eth1*, hay que modificarlo para poner el nombre de la interfaz de red a examinar.

Como último paso, queda la instalación de las reglas y la capacidad de poder hacer uso de las reglas personalizadas.

Para descargar el paquete de reglas actualizadas por la comunidad, se hace uso del siguiente comando.

- `sudo suricata-update`

```
1: vcoralc@cibereguridad: /etc/suricata
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/dnp3-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/dns-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/files.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/http2-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/http-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/ipsec-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/kerberos-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/modbus-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/mqtt-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/nfs-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/ntp-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/qic-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/rfb-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/smb-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/smtp-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/ssh-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/stream-events.rules
4/5/2024 -- 17:11:13 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/tls-events.rules
4/5/2024 -- 17:11:14 - <Info> -- Ignoring file 199696be1bd9c387e199527d3f6eaafb/rules/emerging-deleted.rules
4/5/2024 -- 17:11:14 - <Info> -- Loaded 49158 rules.
4/5/2024 -- 17:11:14 - <Info> -- Disabled 14 rules.
4/5/2024 -- 17:11:14 - <Info> -- Enabled 0 rules.
4/5/2024 -- 17:11:14 - <Info> -- Modified 0 rules.
4/5/2024 -- 17:11:14 - <Info> -- Dropped 0 rules.
4/5/2024 -- 17:11:15 - <Info> -- Enabled 136 rules for flowbit dependencies.
4/5/2024 -- 17:11:15 - <Info> -- Creating directory /var/lib/suricata/rules.
4/5/2024 -- 17:11:15 - <Info> -- Backing up current rules.
4/5/2024 -- 17:11:15 - <Info> -- Writing rules to /var/lib/suricata/rules/suricata.rules: total: 49158; enabled: 37370;
added: 49158; removed 0; modified: 0
4/5/2024 -- 17:11:15 - <Info> -- Writing /var/lib/suricata/rules/classification.config
4/5/2024 -- 17:11:15 - <Info> -- Testing with suricata -T.
5/4/5/2024 -- 17:11:32 - <Info> -- Done.
vcoralc@cibereguridad:/etc/suricata$
```

Figura 31. Resultado del comando *suricata-update*

En el caso de este trabajo se puede observar que se han añadido 49158 reglas. Todas ellas ubicadas en el archivo *suricata.rules* en */var/lib/suricata/rules*.

Para poder hacer uso de un archivo de reglas personalizadas lo que se debe hacer es crear un archivo con cualquier nombre, pero extensión *.rules*.

Y el siguiente paso es editar el */etc/suricata/suricata.yml* para indicarle que archivo *.rules* se desea usar en el *rule-files*, también en caso de haber creado un archivo *.rules* en otro directorio habría que indicárselo en el *default-rule-path*.

```
Tilix: vcoralc@cibereguridad: /etc/suricata
1: vcoralc@cibereguridad: /etc/suricata
GNU nano 4.8 suricata.yaml
hashmode: hashStuplesorted

##
## Configure Suricata to load Suricata-Update Managed rules.
##

default-rule-path: /var/lib/suricata/rules

rule-files:
- suricata.rules

##
## Auxiliary configuration files.
##

classification-file: /etc/suricata/classification.config
reference-config-file: /etc/suricata/reference.config

Get Help Write Out Where Is Cut Text Justify Cur Pos Undo Mark Text
Exit Read File Replace Paste Text To Spell Go To Line Redo Copy Text
```

Figura 32. Captura del archivo suricata.yaml en la sección de reglas

Se pueden indicar varios archivos .rules a utilizar si se desea.

Una vez está todo instalado y correctamente configurado. Se puede iniciar el servicio de Suricata. El comando es:

- sudo systemctl start suricata

Sin embargo, el servicio por configuración predeterminada se inicia solo.

En caso de que se haya actualizado cualquier archivo de configuración de suricata hay que reiniciar el servicio.

- sudo systemctl restart suricata

Para comprobar su estado:

- service suricata status

```
vcoralc@cibereguridad:~$ service suricata status
● suricata.service - LSB: Next Generation IDS/IPS
   Loaded: loaded (/etc/init.d/suricata; generated)
   Active: active (running) since Wed 2024-05-15 11:14:58 CEST; 1h 18min ago
     Docs: man:systemd-sysv-generator(8)
    Tasks: 10 (limit: 9217)
   Memory: 469.1M
    CGroup: /system.slice/suricata.service
            └─1280 /usr/bin/suricata -c /etc/suricata/suricata.yaml --pidfile /var/run/suricata.pid --af-packet -D -vvv

Warning: some journal files were not opened due to insufficient permissions.
```

Figura 33. Resultado del comando service suricata status

En este caso como se puede apreciar Suricata está en marcha.

Y para parar el servicio se introduce:

- sudo systemctl stop suricata

9.2 Instalación y configuración de Elasticsearch y Kibana

Comenzando con la descarga e instalación de Elasticsearch

Primeramente, se descarga la clave pública GPG de Elasticsearch la cual sirve para verificar la autenticidad e integridad de los paquetes que se van a instalar y se añade al almacén de claves APT:

- `curl -fsSL https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -`

A continuación se le indica a APT dónde va a poder hallar el programa:

- `echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list`

Y ya actualizamos la lista APT y se instala Elasticsearch.

- `sudo apt update`
- `sudo apt install elasticsearch`

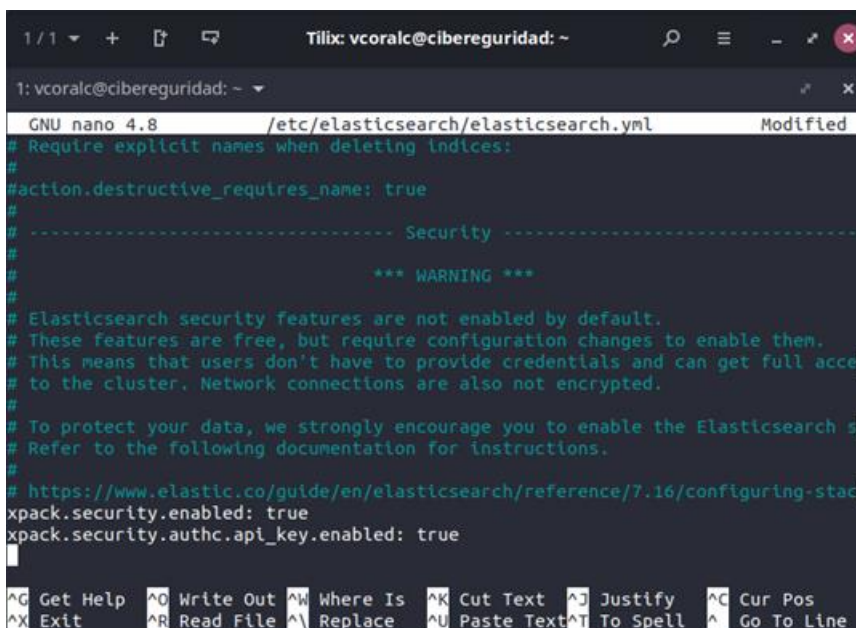
De forma análoga, se procede a la instalación de Kibana

Al ya poseer las claves y los repositorios indicados a APT. Se deben introducir los siguientes comandos.

- `sudo apt update`
- `sudo apt install kibana`

Una vez descargados ambos softwares. El siguiente paso es preparar los archivos de configuración del software Elasticsearch

Para utilizar posteriormente los Fleet Agents que son agentes que sirven para la gestión centralizada de los datos se deberá modificar el archivo `/etc/elasticsearch/elasticsearch.yml` añadiéndole las últimas dos líneas que se muestran en la figura a continuación.



```

1 / 1 + [ ] [ ] Tilix: vcoralc@cibereguridad: ~
1: vcoralc@cibereguridad: ~
GNU nano 4.8 /etc/elasticsearch/elasticsearch.yml Modified
# Require explicit names when deleting indices:
#
#action.destructive_requires_name: true
#
# ----- Security -----
#
#                *** WARNING ***
#
# Elasticsearch security features are not enabled by default.
# These features are free, but require configuration changes to enable them.
# This means that users don't have to provide credentials and can get full access
# to the cluster. Network connections are also not encrypted.
#
# To protect your data, we strongly encourage you to enable the Elasticsearch security
# features. Refer to the following documentation for instructions.
#
# https://www.elastic.co/guide/en/elasticsearch/reference/7.16/configuring-stack.html
xpack.security.enabled: true
xpack.security.authc.api_key.enabled: true

```

Figura 34. Captura del archivo `elasticsearch.yml`

A continuación, se procede a establecer las contraseñas para los servicios de Elastic y securizar todos los programas.

Para ello desde consola se debe ejecutar el siguiente comando:

- `sudo /usr/share/elasticsearch/bin/elasticsearch-setup-passwords interactive`

```
Tilix: vcoralc@cibereguridad: ~
1: vcoralc@cibereguridad: ~
May 21 11:32:15 cibereguridad systemd[1]: Started Elasticsearch.
vcoralc@cibereguridad:~$ sudo /usr/share/elasticsearch/bin/elasticsearch-setup-passwords interactive
Initiating the setup of passwords for reserved users elastic,apm_system,kibana,kibana_system,logstash_system,
beats_system,remote_monitoring_user.
You will be prompted to enter passwords as the process progresses.
Please confirm that you would like to continue [y/N]y

Enter password for [elastic]:
Reenter password for [elastic]:
Enter password for [apm_system]:
Reenter password for [apm_system]:
Enter password for [kibana_system]:
Reenter password for [kibana_system]:
Enter password for [logstash_system]:
Reenter password for [logstash_system]:
Enter password for [beats_system]:
Reenter password for [beats_system]:
Enter password for [remote_monitoring_user]:
Reenter password for [remote_monitoring_user]:
Changed password for user [apm_system]
Changed password for user [kibana_system]
Changed password for user [kibana]
Changed password for user [logstash_system]
Changed password for user [beats_system]
Changed password for user [remote_monitoring_user]
Changed password for user [elastic]
vcoralc@cibereguridad:~$
```

Figura 35. Resultado del comando `elasticsearch-setup-passwords interactive`

Tal como se aprecia en la figura, de forma sencilla y rápida se configuran todas las contraseñas para los servicios de Elastic.

A continuación, se procede a la preparación de los archivos de configuración de Kibana.

Al establecer contraseñas para los servicios, se debe indicar en los archivos de Kibana con que usuario y contraseña se accede a los datos de Elastic.

Para ello se modifica el archivo `/etc/kibana/kibana.yml`. De forma predeterminada el usuario será “elastic” y la contraseña será la indicada en el comando del paso anterior, siempre entre comillas.

```
Tilix: vcoralc@cibereguridad: /var/log
1: vcoralc@cibereguridad: /var/log
GNU nano 4.8 /etc/kibana/kibana.yml
# Kibana uses an index in Elasticsearch to store saved searches, visualizations and
# dashboards. Kibana creates a new index if the index doesn't already exist.
#kibana.index: ".kibana"

# The default application to load.
#kibana.defaultAppId: "home"

# If your Elasticsearch is protected with basic authentication, these settings provide
# the username and password that the Kibana server uses to perform maintenance on the Kibana
# index at startup. Your Kibana users still need to authenticate with Elasticsearch, which
# is proxied through the Kibana server.
elasticsearch.username: "elastic"
elasticsearch.password: " "

# Kibana can also authenticate to Elasticsearch via "service account tokens".
# If you use this token instead of a username/password.
# elasticsearch.serviceAccountToken: "my_token"

# Enables SSL and paths to the PEM-format SSL certificate and SSL key files, respectively.
# These settings enable SSL for outgoing requests from the Kibana server to the browser.
#server.ssl.enabled: false
#server.ssl.certificate: /path/to/your/server.crt
#server.ssl.key: /path/to/your/server.key

Optional settings that provide the paths to the PEM-format SSL certificate and key files.

^G Get Help      ^O Write Out    ^W Where Is    ^K Cut Text      ^J Justify     ^C Cur Pos     ^L Undo
^X Exit          ^R Read File    ^M Replace     ^N Paste Text  ^T To Spell   ^_ Go To Line   ^U Redo
```

Figura 36. Captura del archivo `kibana.yml`

Por último, para poner en funcionamiento ambos programas, se deben ejecutar los siguientes comandos:

- `sudo systemctl start elasticsearch`
- `sudo systemctl start kibana`

En caso de haber hecho alguna modificación sobre los archivos de alguno de los programas (como los .yml) se deberá reiniciar el servicio de la siguiente forma:

- sudo systemctl restart elasticsearch

Y en caso de querer parar el servicio:

- sudo systemctl stop elasticsearch

9.3 Configuración del Stack Elastic para Suricata mediante los Elastic Agents y Fleet Servers.

Al iniciar Kibana desde el navegador se podrá acceder indicando el usuario y contraseña configurado en el punto anterior del anexo.

Una vez dentro, en la sección Integrations habrá habilitada una barra de búsqueda para encontrar el programa a implementar. En esta se busca Suricata y saldrá como primer resultado.

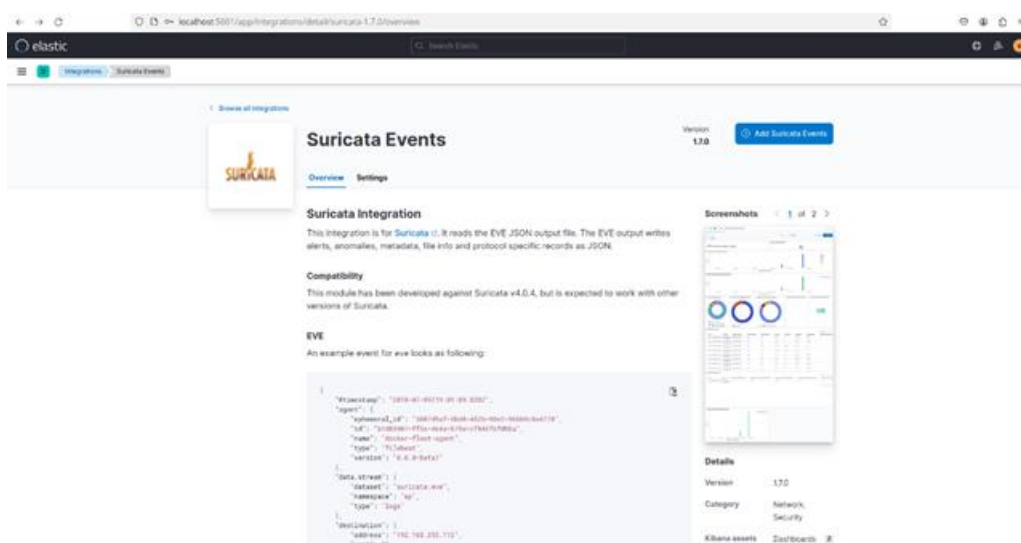


Figura 37. Captura de la integración Suricata dentro de la interfaz web de Kibana

Se añade la integración y procedemos a la instalación de la integración.

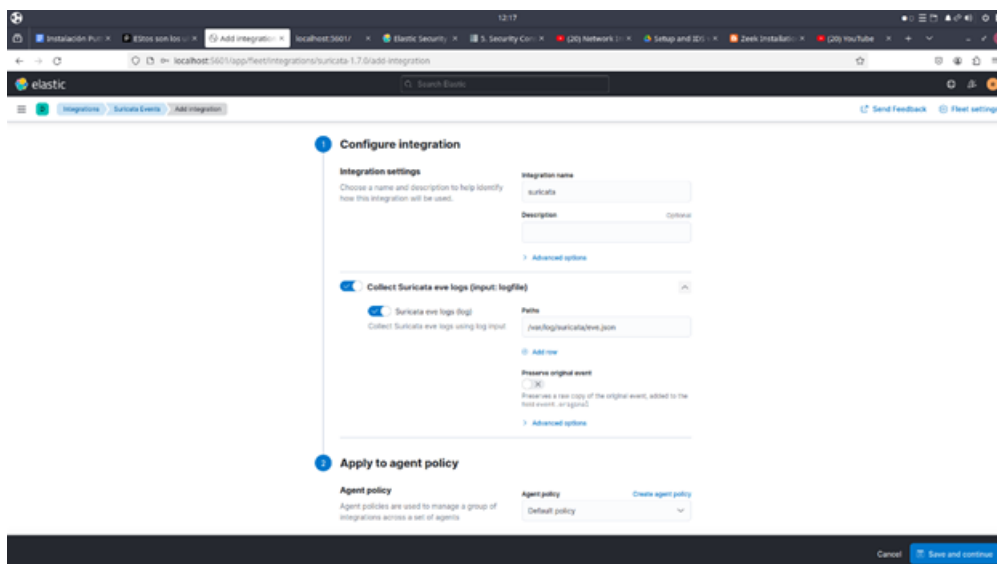


Figura 38. Captura de la configuración de Suricata como Fleet Agent desde Kibana

En este caso, especial atención y asegurar que se van a extraer los datos desde el path correcto. Si no se ha modificado el suricata.yaml, el path predeterminado es el que se muestra en la figura.

El Agent Policy se configura tras guardar la integración ya que avisa de que para poder tratar con los datos hace falta un Fleet Server:

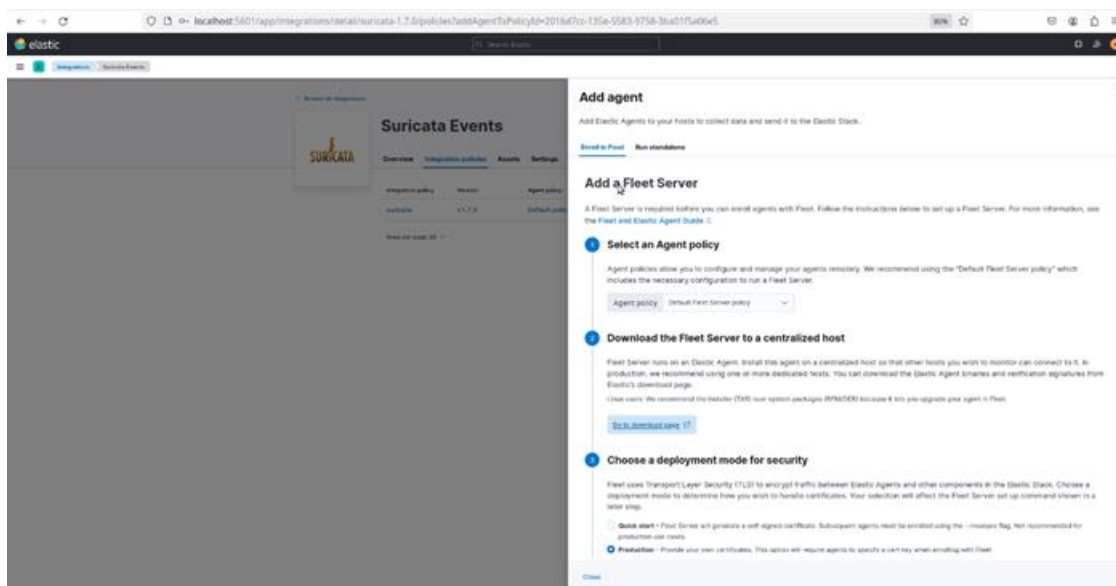


Figura 39. Captura de los pasos a seguir para instalar el Fleet Server de Suricata

En el paso 1 se selecciona el Default Fleet Server Policy.

En el paso 2 te redirige a una web donde descargar el Fleet Server.

Para el paso 3 se deja la opción de Production.

En el paso 4 se establece el host en la dirección recomendada: 127.0.0.1:8220

En el paso 5 simplemente se genera el token automático el cual se corresponde con el código que también se genera automáticamente en el paso 6.

En el último paso copiamos el código que se proporciona y desde la ruta donde se encuentra el fleet server descargado lo ejecutamos desde terminal. Para hacerlo automático se puede no

introducir los comandos referentes a authorities, es-ca, cert, cert-key ya que para ellos hay que indicar donde se encuentran en el caso de que se hayan generado manualmente.

Y finalmente ya debería estar funcionando.

Para la comprobación se puede acceder a Observability>Logs>Stream y ver si efectivamente está cogiendo datos de Suricata.

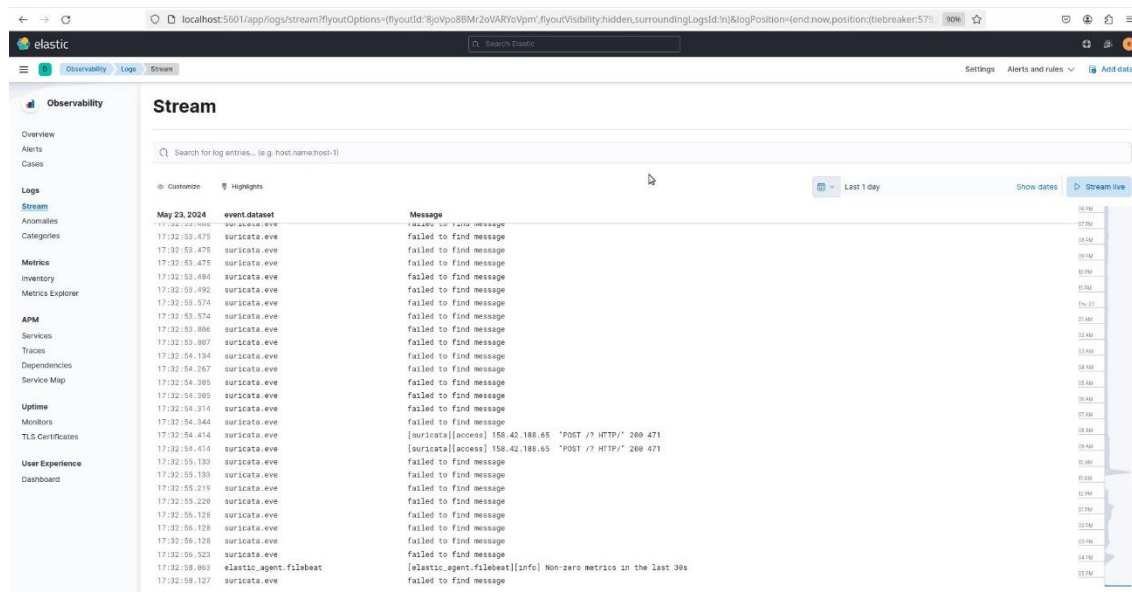


Figura 40. Captura del funcionamiento del Suricata Fleet Server desde Kibana

Y efectivamente, como se puede ver, se están cogiendo datos de la fuente suricata.eve.