Universidad Zaragoza
1542

Trabajo Fin de Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Implementación de un sistema SIEM en un sistema SCADA

Jaime del Amo Pamplona

Director Ricardo J. Rodríguez Fernández
Codirector: José Manuel Portilla Saiz

Escuela de Ingeniería y Arquitectura
Universidad de Zaragoza
Diciembre de 2022
Curso 2022/2023

Resumen

Cada vez es más habitual encontrar la delincuencia en Internet. Los ciberdelincuentes buscan atacar a las grandes organizaciones tanto desde dentro como desde fuera de su red interna. Las organizaciones crean grupos especializados en la ciberseguridad dedicados a prevenir y contrarrestar los ataques. Sin embargo, las fuentes de información son muchas y la información a analizar es abrumadora. Para solventar este problema, se utilizan herramientas como los sistemas SIEM (*Security Information and Event Management*), que son capaces de analizar y correlar información relativa a la seguridad de la organización y alertar de eventos sospechosos. En este trabajo se pretende securizar un entorno de infraestructuras críticas implantando un sistema SIEM en el sistema SCADA que lo controla. Un sistema SCADA es una herramienta informática implantada en una máquina central cuya función consiste en supervisar un entorno industrial analizando datos y subsanando fallos. El sistema SIEM implantado integra la solución de *Elastic*, *Elastic Security*, junto con el resto de tecnología desarrollada por *Elastic*. Para completarlo, se estudian herramientas de terceros que añaden funcionalidades adicionales al sistema SIEM.

Abstract

It is increasingly common to find crime on Internet. Cybercriminals seek to attack large organizations both from within and from outside their internal network. Organizations create specialized cybersecurity groups dedicated to preventing and countering attacks. However, the sources of information are many and the information to be analyzed is overwhelming. To solve this problem, tools such as SIEM (*Security Information and Event Management*) systems are used, which are capable of analyzing and correlating information related to the organization's security and alerting of suspicious events. This work aims to secure an critical infrastructures environment by implementing a SIEM system in the SCADA system that controls it. A SCADA system is a computer tool implemented in a central machine whose function is to supervise an industrial environment by analyzing data and correcting faults. The implemented SIEM system integrates *Elastic*'s solution, *Elastic Security*, together with the rest of the technology developed by *Elastic*. To complete it, third-party tools that add additional functionalities to the SIEM system are studied.

Índice general

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

La sociedad actual se encuentra en un estado de continua revolución informática. La ciberseguridad es actualmente una de las mayores preocupaciones de cualquier empresa debido a que la imparable progresión cibernética crea nuevas formas de vulnerar la seguridad de las mismas [32]. Los medios de comunicación anuncian cada semana nuevos ciberataques, nuevas brechas de seguridad en productos cotidianos o nuevos avances en la guerra cibernética [32]. Las técnicas usadas por los ciberdelincuentes son cada vez más sofisticadas, convirtiéndolas en un peligro a tener en cuenta.

De la misma forma que se han desarrollado técnicas para vulnerar la seguridad, también se han desarrollado técnicas que previenen ataques de distinta índole. La mayoría de ellas nacen de la necesidad de paliar el daño causado por vulnerabilidades *0-day*. Una *vulnerabilidad 0-day* se define como una vulnerabilidad que acaba de ser descubierta y para la que aún no se dispone de una contramedida que la neutralice.

Muchas de las amenazas residen en el tráfico de red que se recibe a diario. Por tanto, se ha desarrollado software y *hardware* específico como *firewalls* o sistemas IDS (*Intrusion Detection System*), que son sistemas capaces de detectar y alertar de accesos no autorizados a un equipo o a una red a través de la monitorización del tráfico de red [29]. La mayoría de los sistemas de seguridad orientados a la monitorización del tráfico de red implementan técnicas para bloquear el tráfico malicioso.

Otras amenazas consisten en atacar directamente los datos alojados en los equipos locales a través de virus, gusanos, troyanos y demás software malicioso (*malware*) [28]. Por ejemplo, se ha diseñado software como los antivirus capaces de neutralizar amenazas de este tipo o técnicas como MTD (*Moving Target Defense*). Las técnicas MTD consisten en cambiar periódicamente un atributo de una red. El atributo que se cambia suele ser uno que el atacante intenta conocer mediante técnicas de reconocimiento para posteriormente explotarlo [31].

Todas las amenazas mencionadas se incrementan en un entorno de producción donde el número de dispositivos aumenta exponencialmente. La abrumadora cantidad

de información a analizar requiere el desarrollo de sistemas que correlacionen dicha información y administren la seguridad de una organización. A este respecto, las organizaciones crean sistemas especializados en resolver problemas relacionados con la seguridad de las mismas a través de los SIEM (*Security Information and Event Management*) [2]. No obstante, la integración de los sistemas SIEM está lejos de ser una solución que evite la totalidad de los ataques. Debido a la naturaleza cambiante del mundo de la ciberseguridad, hay posibilidades de que un ataque sea exitoso con el tiempo y los recursos necesarios.

Para entender los pasos que siguen los ciberatacantes hasta conseguir su objetivo se ha propuesto un modelo aceptado internacionalmente, la *Intrusion Kill Chain* [21], en el que la fase de reconocimiento es la más prematura de todas. En esta fase el atacante intenta descubrir si la máquina objetivo tiene vulnerabilidades que explotar. Es usual que se produzcan actividades sospechosas como los escaneos de puertos, comprobar que un equipo está activo o la detección del sistema operativo del equipo objetivo. Detectar este tipo de actividades puede ayudar a prevenir un posible ataque en una fase muy temprana del mismo.

Este trabajo está orientado a securizar un entorno de infraestructuras críticas controlado por un sistema SCADA [24]. Los sistemas SCADA son herramientas de automatización y control industrial dedicadas a supervisar, recopilar datos, analizar datos y subsanar posibles errores de forma remota.

El trabajo parte de un estado en el que se ha implantado un sistema SIEM con el software *Wazuh*, una solución de seguridad de código abierto y gratuita capacitada para la detección de amenazas, monitorización de integridad y respuesta ante incidentes. A través de un servidor central, se despliegan múltiples agentes para securizar equipos remotos. Sin embargo, los agentes desplegados no recopilan información relativa a la actividad en la red y por lo tanto el SIEM no es capaz de detectar técnicas maliciosas propias de la fase de reconocimiento de un ataque. A raíz de la poca información que ofrece la solución *Wazuh* en lo relativo a la red, se propone paliar este problema y desplegar un sistema SIEM con tecnología que permita inspeccionar el tráfico de red.

1.1. Objetivos

Los objetivos a cumplir en este trabajo son los siguientes: (i) *Diseño de un sistema SIEM mediante la solución Elastic Security*, que se implantará en un SCADA permitiendo securizar un entorno de producción. Se ha escogido *Elastic Security* porque es una solución que permite recopilar múltiples tipos de eventos de seguridad apropiados para satisfacer el resto de objetivos propuestos; (ii) *Detección de ataques*

a nivel de red, que permitirán detectar posibles ataques en fases muy tempranas; (iii) *Analizar las vulnerabilidades a través de un escáner de vulnerabilidades*, que permitirá detectar aplicaciones instaladas con versiones desactualizadas evitando vulnerabilidades que explotar; y (iv) *Dotar al SIEM de integridad de ficheros*, que permitirá alertar de eventos sospechosos relativos a los ficheros del equipo.

1.2. Estructura

En el capítulo 2 se ofrece una breve explicación de los conceptos necesarios para una mejor comprensión del trabajo, describiendo la arquitectura general de un sistema SIEM, los sistemas SCADA, las principales técnicas propias de la fase de reconocimiento y una pequeña introducción a UML. En el capítulo 3 se describen los requisitos funcionales y los requisitos no funcionales del sistema a diseñar, se contextualiza cada requisito y se estudian las mejores herramientas para cada caso. En el capítulo 4 se describe el despliegue de las herramientas seleccionadas en el capítulo anterior adjuntando en algunos casos diagramas UML y se exponen los escenarios del trabajo. En el capítulo 5 se simulan distintos ataques en un entorno controlado para estudiar y analizar la respuesta del sistema SIEM. Por último, en el capítulo 6 se exponen las conclusiones extraídas del trabajo y el trabajo futuro.

Capítulo 2

Conocimientos previos

Este capítulo pretende introducir pequeñas nociones teóricas para facilitar la comprensión del desarrollo realizado en este trabajo. Primero, se introduce la arquitectura general de un sistema SIEM. Después, se proporciona una visión general de los sistemas SCADA. A continuación, se definen las técnicas más usadas en la fase de reconocimiento de un ataque. Por último, se introduce de forma breve UML y se definen los tipos de diagramas UML utilizados en este trabajo.

2.1. Sistemas SIEM

Un SIEM (*Security Information and Event Management*) es una solución de seguridad basada en software que tiene la capacidad de monitorizar la red y los dispositivos conectados a ella [2]. Los SIEM son capaces de detectar, responder y neutralizar ciberamenazas de prácticamente cualquier tipo.

Los sistemas SIEM nacen de la combinación de dos soluciones distintas.

1. **SIM (*Security Information Management*)**. Agentes que se comunican con un servidor centralizado y envían información acerca de eventos relacionados con la ciberseguridad. Las fuentes de datos suelen ser antivirus, *firewalls*, sistemas IDS o servidores proxy.

2. **SEM (*Security Event Management*)**. Sistema que analiza la información recibida por los sistemas SIM en tiempo real y establece correlaciones que permiten detectar actividades sospechosas.

2.2. Sistemas SCADA

Los sistemas SCADA (*Supervisory Control and Data Acquisition*) son sistemas muy utilizados en procesos industriales [24]. Como su nombre indica, se centran en supervisar

y adquirir los datos de diferentes fuentes. Los sistemas SCADA son paquetes de software colocados en el *hardware* que se quiere monitorizar. Normalmente, la conexión se realiza a través de PLCs (*Programmable Logic Controllers*) u otros módulos de *hardware* comercial [3].

El SCADA en el que se prevee implementar el sistema SIEM tiene como sistema operativo Windows. Por ello, la configuración e instalación de todos los módulos del sistema SIEM se realizan en un entorno Windows.

2.3. Técnicas de reconocimiento

Antes de iniciar un ataque, el ciberdelincuente necesita recopilar información del objetivo. Para conseguirla, se usan técnicas que permiten encontrar máquinas activas, puertos abiertos, versiones de aplicaciones, el tipo de sistema operativo y vulnerabilidades de red, entre otras. Las técnicas más comunes son las siguientes:

- **Encontrar máquinas activas.** En esta técnica se utiliza el protocolo de red ICMP. El atacante envía múltiples paquetes *ICMP request*, y si el objetivo está activo y recibe los paquetes, se genera una respuesta *ICMP response* para cada paquete recibido. Si el atacante recibe estos paquetes, deduce que el objetivo está activo.

- **Escaneo de puertos.** A la hora de buscar vulnerabilidades es importante conocer qué puertos están abiertos. Los escaneos de puertos dan mucha información relevante, como por ejemplo los servicios que se están ejecutando en la máquina objetivo. En este trabajo se tratan tres tipos de escaneos de puertos (TCP SYN, TCP Connect y UDP) [23].

  TCP SYN es de los escaneos más populares debido a su rapidez y a que es un ataque silencioso. Se trata de un ataque que inicia conexiones pero no termina de completarlas. Otro aspecto a destacar es que permite diferenciar los estados de un puerto (abierto, cerrado o filtrado).

  TCP Connect se trata de un escaneo parecido a TCP SYN. Es más ruidoso ya que en este caso se realiza una conexión con el objetivo y también permite diferenciar los estados de un puerto.

  UDP es la técnica de escaneo de puertos menos eficiente. Consiste en enviar cabeceras UDP vacías a los puertos objetivos. En caso de estar cerrado, el puerto objetivo envía un error ICMP (normalmente *Port Unreachable*). Hay muchos sistemas operativos como Linux que limitan la cantidad de mensajes ICMP *Port*

Unreachable que se pueden enviar. Otra desventaja clara es que es bastante más lento que las técnicas TCP. En la Tabla 2.1 se recogen los detalles de cada uno de los escaneos descritos.

Nombre	Funcionamiento	Observaciones
TCP SYN Stealth	Envía un SYN a los puertos objetivos. Es rápida, fiable y relativamente sigilosa.	- Closed: Recibe RST. - Open: Recibe SYN/ACK. - Filtered: ICMP unreachable o expira el timeout.
TCP Connect	Envía un SYN, luego un RST para cerrar la conexión. Se utilizan llamadas del sistema operativo por lo que es menos eficiente que SYN Stealth.	- Closed: Recibe RST. - Open: Recibe SYN/ACK. - Filtered: ICMP unreachable o expira el timeout.
UDP Scan	Envía un paquete UDP vacío. Es bastante más lento que un escaneo TCP.	- Closed: Recibe ICMP Port Unreachable. - Filtered: Recibe otros ICMP unreachable. - Open: Hay una respuesta. - Open/Filtered: Expira el timeout.

Tabla 2.1: Resumen de técnicas de reconocimiento mediante escaneo de puertos

2.4. UML

El Lenguaje Unificado de Modelado (UML, en inglés *Unified Modeling Language*) [25] es un estándar adoptado a nivel internacional por un elevado número de empresas y organismos para crear esquemas, diagramas y documentación relativa a los desarrollos de software. En este trabajo se ha utilizado el diagrama de actividad, que representa los flujos de trabajo de forma gráfica. Existe un único nodo de inicio y el flujo puede terminar en múltiples sitios del diagrama.

Capítulo 3

Análisis

En este capítulo se pretende definir las funcionalidades que implementa el SIEM. Primero, se describen los requisitos funcionales y no funcionales junto con un diccionario de palabras clave. A continuación, se proponen soluciones que implementan los requisitos descritos.

3.1. Requisitos funcionales y no funcionales

El primer problema a abordar en el desarrollo del sistema consiste en establecer de forma clara qué funcionalidades va a incluir el SIEM. Para ello, se han considerado los requisitos funcionales (RF) y los requistos no funcionales (RNF) que se muestran en la Tabla 3.1.

Requisitos	Descripción
RF1	El sistema debe ser capaz de detectar software malicioso.
RF2	El sistema debe ser capaz de detectar técnicas propias de la fase de reconocimiento de un ataque.
RF3	El sistema debe ser capaz de detectar intentos de inicio de sesión por parte de los usuarios.
RF4	El sistema debe detectar eventos de integridad de ficheros.
RF5	El sistema debe ser capaz de capturar el tráfico de red.
RF6	El sistema debe ser capaz de detectar vulnerabilidades en las aplicaciones instaladas.
RF7	El sistema debe ser capaz de recibir, identificar e interpretar datos de múltiples fuentes.
RF8	El sistema debe facilitar la creación de reglas definidas con las que analizar la información entrante y generar alertas que notifiquen potenciales riesgos para la seguridad.
RF9	El sistema debe ser capaz de desplegar múltiples agentes en múltiples equipos de forma simultánea.
RNF1	La comunicación entre el servidor y los agentes debe estar autenticada mediante certificados.

Tabla 3.1: Descripción requisitos funcionales y no funcionales

En el siguiente diccionario se recogen algunos conceptos que requieren de una explicación adicional.

- **Software malicioso.** Programa ejecutable con intenciones maliciosas ejecutado en la máquina objetivo por el atacante.

- **Inicio de sesión.** Acción realizada por un usuario para acceder a un equipo introduciendo las credenciales correspondientes.
- **Integridad de ficheros.** Capacidad para detectar cambios en los ficheros de un equipo.
- **Tráfico de red.** Paquetes entrantes y salientes detectados por una interfaz de red.
- **Vulnerabilidades.** Debilidades que ponen en riesgo a un equipo o a una red al permitir a un atacante explotarla con fines maliciosos.
- **Fuentes de datos.** Información procesada por los agentes desplegados y enviada al servidor central.
- **Reglas.** Permiten al sistema alertar si las condiciones que las definen se cumplen.
- **Certificados.** Documentos electrónicos que permiten una conexión segura entre un cliente y un servidor generados por una entidad de confianza.

3.2. SIM y SEM

En este apartado se definen las herramientas utilizadas para implementar las dos partes bien diferenciadas que conforman un SIEM (véase la Sección 2.1).

3.2.1. SIM

La parte SIM está formada por los agentes recopiladores de datos. La información recopilada es enviada posteriormente a un servidor central. En este proyecto se decide utilizar como agentes recopiladores el agente desarrollado por Elastic, *Elastic agent* [11] y *Beats* [5]. En la Tabla 3.2 se recogen los aspectos más destacados de cada agente.

Agentes	Características
Elastic agent	- Personalizables. - Definidos por integraciones.
Beats	- Dedicados a un solo tipo de dato. - Amplio abanico.

Tabla 3.2: Principales características de los agentes

Una de las características del *Elastic agent* es que se puede personalizar. Cada uno de los agentes desplegados está orientado a recopilar uno o varios tipos de datos. Los tipos de datos que recopila vienen definidos por las integraciones que forman al

propio agente, de tal manera que un mismo agente puede recopilar varios tipos de datos diferentes.

Por otra parte, los *Beats* son agentes de Elastic diseñados para recopilar un solo tipo de datos. La familia de los *Beats* está formada por muchos agentes. De esta manera se establece una red de agentes permitiendo al SIEM obtener información de múltiples fuentes.

3.2.2. SEM

La parte SEM es la pieza central del SIEM. Dado que se va a trabajar con Elastic se decide establecer la *Elastic Stack* como organismo central. La *Elastic Stack* se define como la unión de los software *Elasticsearch* [16], *Kibana* [17] y *Logstash* [12]. En la Tabla 3.3 se recogen las características de cada uno.

Agentes	**Características**
Elasticsearch	- Servidor central. - Indexa la información en índices.
Kibana	- Herramienta gráfica. - Despliega agentes. - Implementa reglas.
Logstash	- Transforma datos

Tabla 3.3: Características principales de *Elastic stack*

La información recopilada por los agentes se envía a *Elasticsearch* donde se indexa en forma de índices. La información puede ir directamente al servidor o ser procesada primero por *Logstash*. Por otro lado, *Kibana* es la interfaz gráfica del conjunto, y su principal función consiste en mostrar los datos indexados en *Elasticsearch* de una manera fácil de entender. No obstante, a través de herramientas implementadas como *Fleet* se despliegan agentes y se implementan reglas. En la Figura 3.1 se describe cómo actúa la *Elastic stack*.

Una de las integraciones disponibles es la integración *System*. Esta integración permite que dicho agente recopile logs de una ruta concreta. De tal manera que, aunque la *Elastic Stack* dispone del software *Logstash* para recopilar y procesar datos, se proponen los *Elastic agent* como vehículo principal para transportar los logs hasta *Elasticsearch*.

Disponer de *Kibana* permite al SIEM desplegar múltiples agentes a través de *Fleet*, así como implementar reglas personalizadas con las que generar alertas en caso de detectar un evento sospechoso.

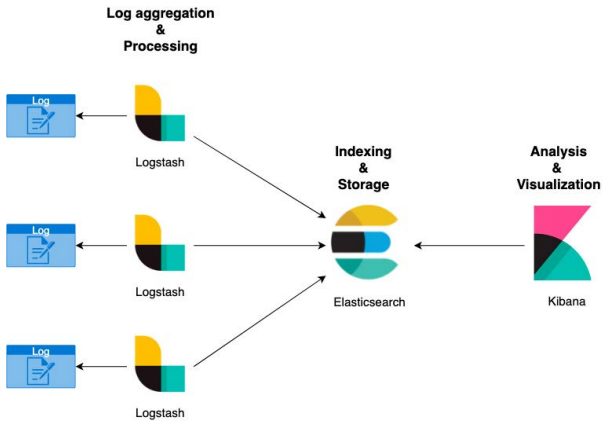

Figura 3.1: Diagrama *Elastic stack* (Fuente [27])

3.3. *Elastic security*, detector de *malware* y recolector de eventos de seguridad

La integración que va a permitir detectar software malicioso y recolectar información relativa a eventos de seguridad es la integración de *Elastic Security* [7]. Incorpora dos modos de trabajo en el SIEM:

Prevención. Al detectar el malware lo inutiliza, declarando los ficheros sospechosos en cuarentena y genera una alerta para avisar del evento.

Detección. Al detectar el fichero simplemente genera la alerta correspondiente, pero no declara en cuarentena ningún fichero.

Elastic Security posee una base de datos cuyo contenido consiste en los identificadores de las aplicaciones que se deben bloquear. A través de la base de datos decide si una aplicación debe ser bloqueada o no [6]. En la Tabla 3.4 se resumen las ventajas y las desventajas de *Elastic security* como detector de software malicioso.

Agentes	Ventajas	Desventajas
Elastics Security	- Permite trabajar en modo prevención y modo detección. - Incorpora una *blocklist* predefinida. - Permite la recolección de un amplio tipo de eventos. - Genera alertas relativas al *malware* detectado mediante reglas predifinidas.	- *Blocklist* personalizada requiere de pago. - Servicios adicionales requieren licencia.

Tabla 3.4: Ventajas y desventajas de *Elastic security*

Aunque *Elastic* es un servicio gratuito, la mayoría de servicios ofrecidos por la integración *Elastic security* son de pago. No obstante, los servicios incluidos en el plan gratuito son suficientes para los objetivos propuestos en este trabajo.

Entre otros eventos recopilables por *Elastic security* en Windows, se encuentran los intentos de inicio de sesión y el tráfico de red. Sin embargo, *Beats* dispone de un agente dedicado a la captura del tráfico de red más eficiente que esta integración.

3.4. Detección de técnicas propias de la fase de reconocimiento

En la Sección 2.3 se describen algunas de las técnicas más usuales en la fase de reconocimiento de un ataque. La mayoría de ellas intentan escanear al objetivo en busca de vulnerabilidades que explotar a través de la red. Por este motivo se analizan soluciones IDS/IPS que puedan ser de interés para su integración en el SIEM. Las soluciones IPS (sistemas de prevención de intrusos) son soluciones capaces no solo de alertar, sino de actuar ante un posible riesgo en la seguridad (por ejemplo, descartando paquetes o conexiones).

Snort. *Snort* [18] es un sistema IDS/IPS de código abierto que puede funcionar como un rastreador de paquetes para monitorizar el sistema en tiempo real. Su funcionamiento consiste en un sistema de reglas fácil de crear y de implementar. Incorpora preprocesadores útiles para detectar ataques como denegaciones de servicio, desbordamientos de búfer, ataques vía web y escaneo de puertos sigiloso.

Suricata. *Suricata* [18] es un motor de red de alto rendimiento IDS, IPS y seguridad de red. Se trata de una aplicación de código abierto y multiplataforma por lo que es compatible con muchos sistemas operativos diferentes. Está basado en un conjunto de reglas desarrolladas externamente para supervisar el tráfico de red y generar alertas al usuario administrador cuando se producen eventos sospechosos.

En la Tabla 3.5 se resumen los aspectos más importantes de cada software con el fin de compararlos y escoger la solución más adecuada. Aunque ambos software tienen lo necesario para encajar en el proyecto, se decide escoger *Snort*. El principal motivo es el previo conocimiento del sofware y de su sistema de reglas. *Snort* también cuenta con un preprocesador dedicado a detectar los escaneos de puertos.

3.5. Integridad de ficheros y procesos

Uno de los objetivos del atacante es que la víctima no sea consciente del ataque. Para ello, el malware suele ocultarse, ya sea en una ruta poco usual o cambiando el nombre de la aplicación maliciosa por uno que a priori no es sospechoso. En cualquier

Software	Ventajas	Desventajas
Snort	- Gratis. - Código abierto. - Sistema de reglas conocido. - Amplio abanico de opciones en las reglas. - Preprocesadores.	- Configuración poco intuitiva.
Suricata	- Gratis. - Código abierto. - Compatibilidad elementos seguridad.	- Sistema de reglas no conocido

Tabla 3.5: Snort vs Suricata

caso, es sensato implementar una funcionalidad que controle la modificación de ficheros y procesos. Las soluciones estudiadas son las siguientes:

Auditbeat. *Auditbeat* es un agente perteneciente a la familia de los *Beats* especializado en auditar las actividades de los usuarios y procesos del host en el que está desplegado [4].

File Integrity Monitoring. *File Integrity Monitoring (FIM)* es una integración que ofrece un control de los ficheros del equipo securizado a tiempo real [10].

Software	Ventajas	Desventajas
Auditbeat	- Audita la modificación de ficheros. - Información sobre procesos.	- Programa adicional. - Configuración más tediosa.
FIM	- Implementación rápida y sencilla. - No consume prácticamente recursos.	- No ofrece información sobre procesos.

Tabla 3.6: Auditbeat vs File Integrity Monitoring

La Tabla 3.6 muestra un estudio comparativo entre ambas soluciones. Finalmente, se decide utilizar Auditbeat, puesto que se trata de una solución más completa aunque implique una configuración más compleja y un programa adicional más.

3.6. Captura de tráfico de red

Aunque se incorpora Snort como solución IDS, que es capaz de capturar el tráfico de red, se decide también desplegar un agente dedicado a la captura de paquetes. La motivación principal es porque los agentes introducen un archivo JSON en *Elasticsearch* por cada paquete inspeccionado, permitiendo posteriormente analizar los campos del paquete en profundidad. Se han estudiado dos soluciones posibles:

Packetbeat. *Packetbeat* es un agente perteneciente a la familia de los *Beats* especializado en analizar el tráfico a tiempo real. El principal motivo por el que se considera es para implementar reglas más robustas [14].

Network Packet Capture. *Network Packet Capture (NPC)* es una integración que permite capturar el tráfico de red en una interfaz de red [13].

Software	Ventajas	Desventajas
Packetbeat	- Soporta más protocolos. - Está más depurado.	- Programa adicional.
NPC	- Implantación rápida y sencilla. - No consume prácticamente recursos.	- Soporta menos protocolos. - Puede comprometer el correcto funcionamiento del sistema

Tabla 3.7: Packetbeat vs Network Packet Capture

La Tabla 3.7 muestra un estudio comparativo entre ambas soluciones. En este caso, se decide escoger Packetbeat debido a que la integración de NPC generaba errores al implantarla en el sistema.

3.7. Escáner de vulnerabilidades

Muchas de las posibles formas de vulnerar la seguridad es a través de aplicaciones instaladas con una versión desactualizada. Las actualizaciones arreglan vulnerabilidades encontradas en versiones anteriores. Por este motivo es muy recomendable disponer de un escáner de vulnerabilidades. En este trabajo se estudian tres soluciones con las que desplegar el escáner de vulnerabilidades.

VulnWhisperer. *VulnWhisperer* es un proyecto de código abierto cuya principal función consiste en hacer de intermediario entre un escáner de vulnerabilidades y *Logstash*. Los escáners compatibles con VulnWhisperer son *Nessus*, *Qualys Web Application*, *Qualys Vulnerability Management* y *OpenVAS* [19].

Nessus. Nessus es un escáner de vulnerabilidades de red capaz de detectar los equipos que forman un red. Cuenta con una aplicación que se encarga de realizar el escaneo de vulnerabilidades. También dispone de una interfaz gráfica a través de la que se puede ver la información. Nessus es el único escáner compatible con VulnWhisperer que tiene una versión gratuita [30].

Nmap. *Nmap* es un software famoso por ejecutar escaneos de puertos de diversa índole. Sin embargo, mediante el NSE (*Nmap Scripting Engine*) se pueden desarrollar scripts que automatizan una gran variedad de tareas, permitiendo a *Nmap* tener

funcionalidades más allá que la de escanear puertos. Vulscan es un módulo que precisamente convierte a *Nmap* en un escáner de vulnerabilidades de red, al igual que Nessus [26].

Wazuh. *Wazuh* es un software que propone una solución SIEM. Dispone de un servidor central en el que se almacena la información recopilada por los *Wazuh agent*. Entre otras cosas, los agentes recopilan información relativa a módulos y paquetes para compararla en el servidor central con bases de vulnerabilidades conocidas actualizadas.

En la Tabla 3.8 se recogen las ventajas y desventajas de cada una de las soluciones propuestas. Finalmente, se decide utilizar el detector de vulnerabilidades del software implementado en el sistema SCADA (*Wazuh*) por motivo económico. En el caso de asumir gastos económicos, la opción más correcta es Nessus ya que ofrece todo lo que ofrece Wazuh junto con un análisis de vulnerabilidades de red.

Cabe destacar que para implementar *Wazuh* se decide usar los servicios disponibles en la nube, dado que al estar instalado en el sistema SCADA se considera poco eficiente instalarlo de forma local.

Software	**Ventajas**	**Desventajas**
Wazuh	- Despliega agentes. - Gratis. - Detecta vulnerabilidades en aplicaciones instaladas. - Trabaja con bases de vulnerabilidades actualizadas.	- No detecta vulnerabilidades de red - La información del SIEM queda repartida entre *Elasticsearch* y el servidor de *Wazuh*.
Nessus	- Información recopilada se puede indexar en Elasticsearch a través de *VulnWhisperer*. - Detecta vulnerabilidades de red. - Escanea y detecta máquinas en redes locales.	- Versión gratuita muy limitada. - No despliega agentes en la versión gratis. - No detecta vulnerabilidades en aplicaciones instaladas.
Nmap	- Trabaja con bases de vulnerabilidades conocidas	- No detecta vulnerabilidades en aplicaciones instaladas.

Tabla 3.8: *Wazuh* vs Nessus vs *Nmap*

Capítulo 4

Integración del sistema SIEM

En este apartado se explica en detalle cómo se han desplegado las soluciones escogidas en el capítulo 3. Para facilitar la comprensión del despliegue de agentes se muestran diagramas de actividad que describen el proceso. Adicionalmente, se propone un escenario alternativo en el que desplegar el sistema SIEM.

4.1. Descripción del sistema

El escenario completo se puede observar en la Figura 4.1 y se divide en dos partes fundamentales:

Sistema SIM. Este sistema es el encargado de recoger los datos del equipo securizado. Los datos recopilados son los eventos de seguridad, información de rendimiento del equipo, los logs de alertas generados por *Snort*, los paquetes capturados por *Packetbeat* y la información relativa a la integridad de ficheros mediante *Auditbeat*. Todos los datos recopilados son enviados al servidor central *Elasticsearch* mediante el *Elastic agent*. Las vulnerabilidades del equipo securizado son recogidas y enviadas por el *Wazuh agent* a la nube de *Wazuh*.

Sistema SEM. Por otro lado, este sistema es el que recibe toda la información de los agentes y se encarga de correlacionar toda la información y generar alertas en función de las reglas predefinidas o personalizadas a través de *Kibana*.

4.2. Integración del sistema SIM

En esta sección se despliega la tecnología que define el sistema SIM. Cada apartado explica el despliegue de las herramientas necesarias para satisfacer los requisitos de este sistema.

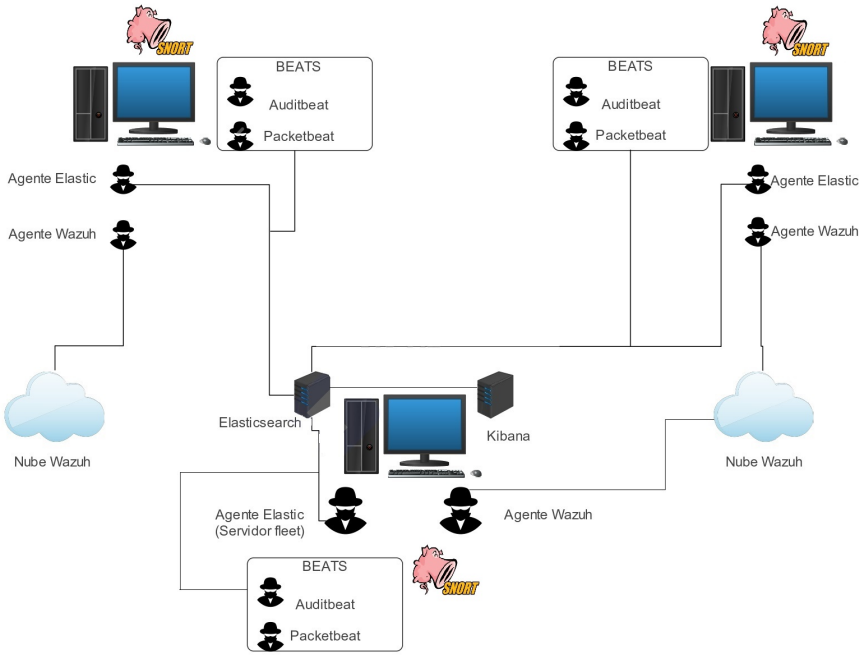

Figura 4.1: Escenario completo

4.2.1. Detección de *malware*

Para la implantación del detector de software malicioso es necesaria la incorporación de la integración *Elastic Security* en el agente del equipo que se pretende securizar.

Entre otros eventos que recopila esta integración se encuentran los relativos a los procesos. Cuando un proceso es ejecutado en el equipo, la integración lo detecta. Para determinar si se trata de *malware*, coteja el identificador o la firma del proceso detectado con su *blocklist*, y en caso de haber coincidencias, se etiqueta como *malware*. Tal y como se explica anteriormente, el sistema puede detectar *malware* en dos modos de trabajo: *prevención* y *detección*.

Por motivos obvios el modo de trabajo predefinido de la integración es el modo prevención. En el caso de que el proceso detectado sea considerado como benigno, la información recopilada es enviada igualmente a *Elasticsearch*. De esta manera queda registrada la información relativa tanto a procesos etiquetados como malware como a procesos benignos.

4.2.2. Detección de técnicas de reconocimiento mediante escaneos de red

Para detectar las amenazas a través del tráfico de red es imprescindible que *Snort* esté configurado de forma correcta. Una vez implementadas las reglas de forma correcta, es preciso que *Snort* esté activo para inspeccionar los paquetes del tráfico de red entrante

y saliente. *Snort* busca en los campos del paquete indicados en las opciones de las reglas *Snort* o en los preprocesadores activos en el fichero de configuración. En el momento en que detecta una coincidencia genera un log informando de la actividad sospechosa detectada. *Snort* puede funcionar en varios modos de trabajo, en este caso interesa que los logs generados por las reglas o preprocesadores se almacenen en un fichero de alertas. Para ello se especifica que mientras *Snort* esté activo, este continúe inspeccionando paquetes y guardando logs en el caso de detectar una coincidencia.

Por otro lado, se encuentra el agente *Elastic agent*. En la recolección de logs del fichero de alertas de *Snort* interviene la integración *System*. Al detectar contenido en el fichero, recopila la información y la envía a *Elasticsearch* para su posterior indexación. De esta manera, *Snort* detecta paquetes de red propios de técnicas de reconocimiento. Al mismo tiempo, la información queda indexada en *Elasticsearch* para un posterior análisis.

Cabe destacar que con la implementación de las reglas Snort pertinentes no solo se avisa de la presencia de posibles sospechosos, sino que puede rechazarlos evitando que lleguen al equipo securizado.

4.2.3. Detección inicio de sesión

Entre los eventos que puede recopilar un agente con la integración *Elastic Security* se encuentran los eventos generados por los inicios de sesión. Cuando un usuario ingresa sus credenciales en el sistema, se generan eventos que son detectados por la integración. La información recopilada (ya sea de un inicio de sesión exitoso o de uno fallido) se envía a *Elasticsearch* para su posterior indexado.

4.2.4. Integridad de ficheros

Para dotar al sistema de integridad de ficheros es necesario configurar e iniciar de forma correcta el agente *Auditbeat*. Entre otros aspectos es importantes configurar las rutas en las que *Auditbeat* detecta los cambios en ficheros. Las rutas vigiladas por *Auditbeat* son rutas en las que un evento generado por la interacción con un fichero se considera un evento sospechoso. Si *Auditbeat* detecta un evento generado por la interacción con un fichero, lo puede etiquetar con distintas categorías (en concreto, creación, modificación o borrado).

Una vez se cataloga el tipo de acción, se recopila información importante relativa al evento como la ruta, el identificador del fichero, el usuario que ha causado el evento o el nombre del fichero. La información recopilada se envía al *Elasticsearch* para su posterior indexado.

4.2.5. Captura del tráfico de red

Para la captura de paquetes es necesario que el agente *Packetbeat* esté activo. Antes de ejecutar este servicio, es preciso configurar el agente con aspectos como la interfaz en la que debe capturar y los protocolos de red que debe capturar.

Packetbeat captura todos los paquetes, ya sean entrantes o salientes. Por cada paquete capturado por el agente se genera una tabla JSON con la información recopilada de ese paquete. La tabla JSON es enviada a *Elasticsearch* en donde se indexa. De esta manera, todos los paquetes capturados quedan registrados para su posterior análisis.

4.2.6. Detector de vulnerabilidades

El detector de vulnerabilidades utilizado es el que tiene implementado *Wazuh*. El usuario accede a los servicios de *Wazuh* a través de la nube. El servidor *Wazuh* pide a los agentes que recopilen información y la envíen de forma periódica. Los agentes recopilan la información de los módulos y paquetes instalados en el equipo securizado. Por otro lado, el administrador crea una base de datos global a partir de repositorios CVE públicos y los usa para cotejar la información de cada base de datos [33].

Una vez generada la base de datos, el servidor busca paquetes o módulos vulnerables en la información recopilada por cada agente. El paquete o módulo se declara vulnerable cuando su versión está dentro del rango definido por una CVE. En el caso de que uno de los módulos recopilados se considere vulnerable, se genera una alerta y se añade la vulnerabilidad a la lista que se muestra al usuario.

Una vez realizado el primer escaneo de vulnerabilidades, se pueden encontrar dos tipos de escaneos. Mediante el *escaneo parcial* solo los nuevos paquetes o módulos son escaneados y como resultado cualquier cambio en la base de datos de vulnerabilidades será alertado. A través del *escaneo completo* se vuelven a escanear todos los paquetes y la base de datos global se vuelve a generar con los repositorios CVEs. Las alertas se generan cuando un paquete o módulo se considera vulnerable.

4.3. Despliegue del sistema SIM

En esta sección se describen los procesos de despliegue de agentes y de definición de reglas a través del sistema SEM.

4.3.1. Despliegue de agentes

Uno de los requisitos del SIEM es poder desplegar múltiples agentes en diferentes equipos con el fin de recopilar información. Los agentes que se depliegan en este sistema se clasifican en tres categorías: *Elastic agent*, *Beats* y *Wazuh agent*.

Elastic agent

En la Figura 4.2 se muestra el diagrama de actividad propio del despliegue de los *Elastic agent* [15]. El despliegue de los *Elastic agent* se realiza a través de *Fleet*, una herramienta incorporada en *Kibana*. Para desplegar un agente es necesario asignarle una política, que se define como un conjunto de integraciones.

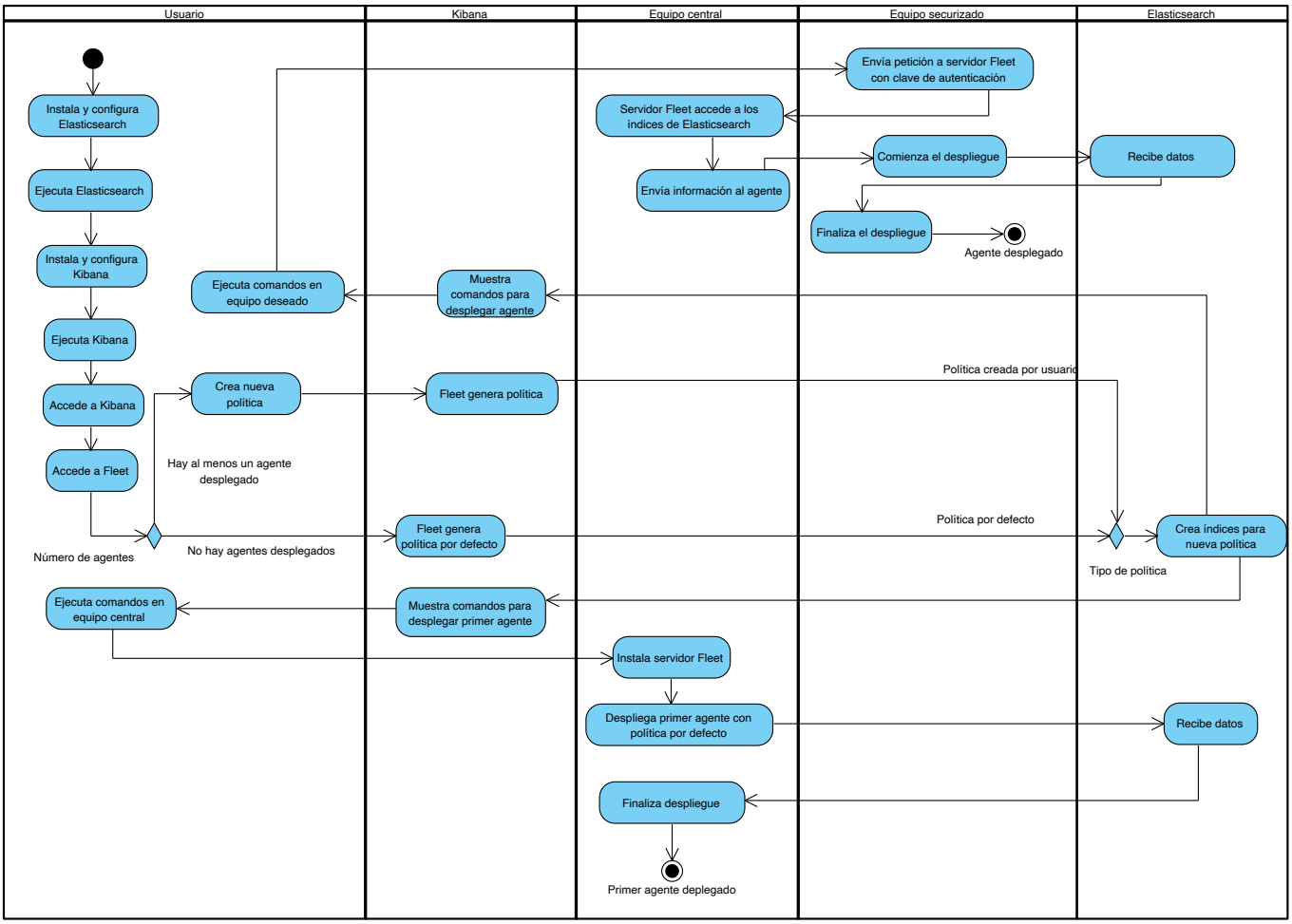

Figura 4.2: Diagrama de actividad del despliegue de *Elastic agent*

En el despliegue del primer agente es el propio *Fleet* el que genera una política por defecto. Esto se debe a que al primer agente desplegado se le añade el servidor *Fleet* para procesar peticiones del resto de agentes. Al generar la política, se crean índices en *Elasticsearch* que albergan información de la propia política. *Fleet* proporciona los

comandos necesarios para el despliegue del agente y para la instalación del servidor *Fleet* que se deben ejecutar en el equipo central. Una vez instalado el servidor de forma correcta, se verifica el despliegue del agente con el primer envío de datos a *Elasticsearch*.

Para desplegar nuevos agentes una vez se ha desplegado el primero, el proceso es diferente. En este caso la política es elegida por el usuario. Al crear la nueva política se generan nuevos índices en *Elasticsearch* a través de *Fleet*. Los comandos se ejecutan en el equipo en el que se quiere desplegar el agente. El agente envía una petición al servidor *Fleet* junto con una clave de autenticación. El servidor recoge la información de la política con la que se quiere desplegar el nuevo agente de los índices correspondientes de *Elasticsearch* y se la envía al agente. El agente utiliza la información recibida para finalizar el despliegue y comenzar a enviar datos a *Elasticsearch*.

Cabe destacar que para la incorporación de nuevas integraciones a políticas ya existentes el proceso es el mismo, con la diferencia de que no hay comandos que ejecutar.

Beats

En la Figura 4.3 se observa el diagrama de actividad propio del despliegue de los *Beats*. El proceso de despliegue de los *Beats* es más sencillo que el proceso de despliegue de los *Elastic agent*. En este caso, el agente se descarga y se configura indicando aspectos como la localización de *Elasticsearch* y *Kibana*. Una vez configurado, se ejecuta en el equipo que se quiere securizar. El *Beat* se comunica con *Elasticsearch* para crear un conjunto de índices predefinidos y con *Kibana* para crear un conjunto predefinido de plantillas. Una vez se han creado tanto los índices como las plantillas, el *Beat* se instala como un servicio en el equipo y empieza a recopilar información que posteriormente se envía a *Elasticsearch*.

Wazuh agents

En la Figura 4.4 se muestra el diagrama de actividad propio del despliegue de los *Wazuh agent* [34]. El último de los agentes que se despliegan en el SIEM son los *Wazuh agent*. El servidor central de *Wazuh* se implementa en la nube tal y como se indica en la Sección 3.7. Para ello, es necesario desplegar un entorno en el que recibir datos de los agentes. La interfaz gráfica de *Wazuh* muestra el menú a través del cual se despliega el agente, donde hay que proporcionar información acerca del sistema operativo en el que se va a desplegar y la dirección del servidor de *Wazuh*. La interfaz gráfica muestra los comandos a ejecutar en el equipo deseado. Al ejecutarlos, el proceso de configuración y despliegue lo gestiona la nube de *Wazuh*. El agente se registra en el servidor y comienza a recopilar información para enviarla a la nube, donde se procesa y se analiza.

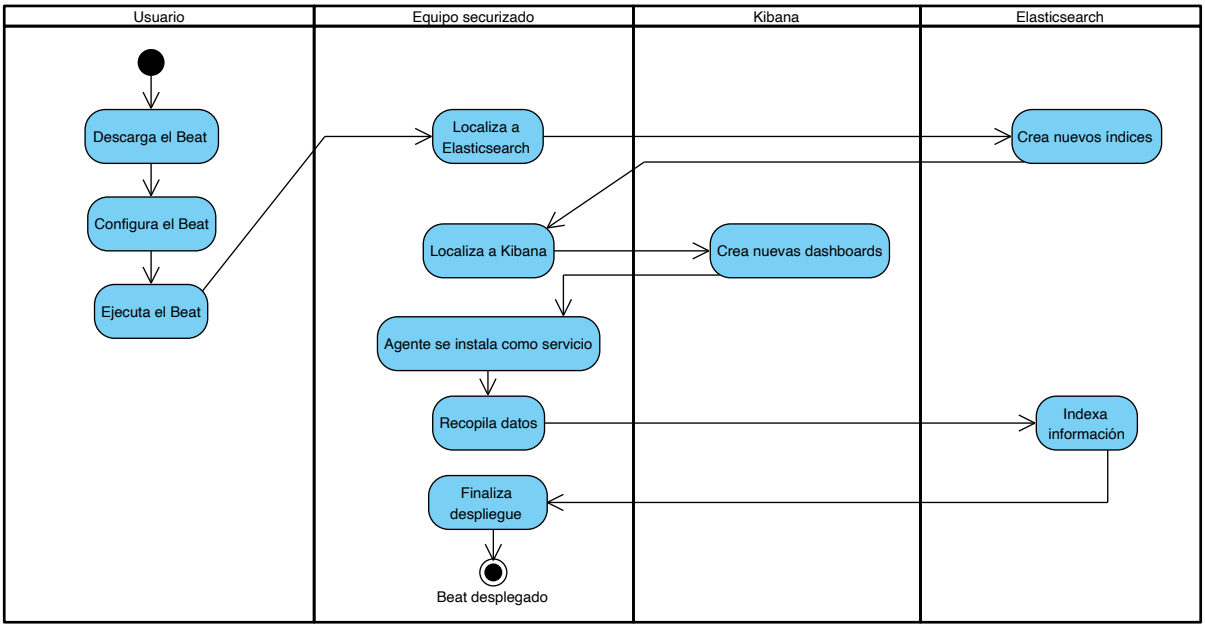

Figura 4.3: Diagrama de actividad del despliegue de *Beats*

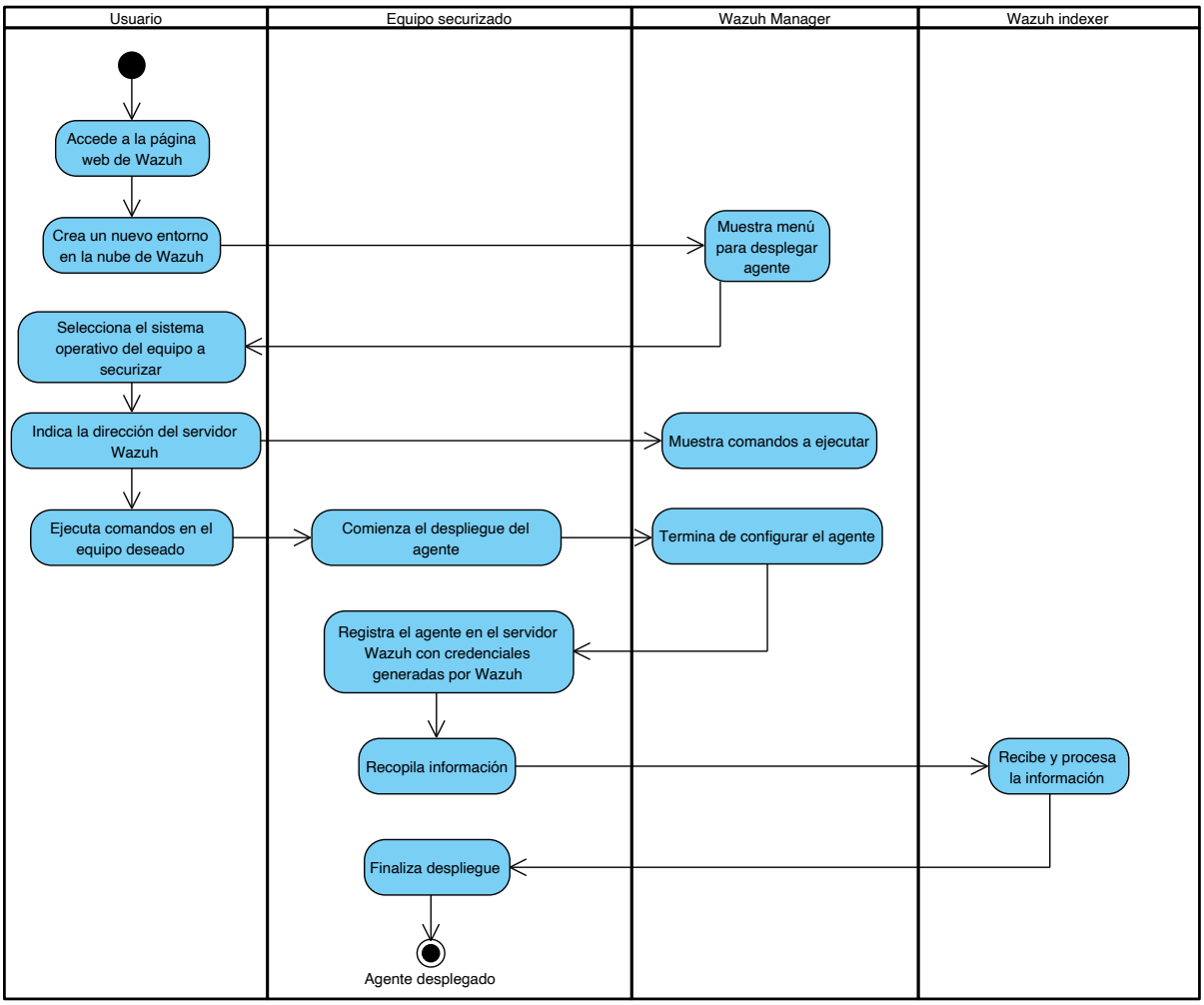

Figura 4.4: Diagrama de actividad del despliegue de *Wazuh agent*

4.3.2. Implementación de las reglas y creación de alarmas

La parte SEM de un sistema SIEM también incluye la creación de reglas que generen alarmas al detectar un evento sospechoso a partir de la información recibida por los agentes.

La integración *Elastic Security* añade la funcionalidad de implementar reglas a través de *Kibana*. Para crear una nueva regla es preciso indicar qué tipo de regla se quiere implementar, la condición que se debe cumplir para generar la alerta y cada cuánto tiempo se consultarán los índices implicados en la condición. La regla queda guardada en los índices correspondientes de *Elasticsearch* y mientras se encuentre activa se consulta la información enviada por los agentes. Si la condición que define la regla se cumple, se genera una nueva alerta añadiendo la información del evento que la ha causado [8].

4.4. Securización de las conexiones

Un aspecto a tener en cuenta es que las conexiones entre las partes del SIEM deben estar autenticadas. *Elasticsearch* cuenta con un módulo de seguridad, *x-pack-security*, con el que se puede generar un certificado firmado por *Elasticsearch* como entidad de confianza. Para securizar las conexiones hay que distinguir entre los tipos de agentes desplegados [9]:

1. *Elastic agent*. Para securizar la conexión entre los *Elastic agent* y *Elasticsearch* se debe incluir el certificado generado por el módulo de seguridad en la configuración de *Fleet*. De esta manera, todos los *Elastic agent* desplegados se comunican de manera segura.

2. *Beats*. Para securizar la comunicación de un *Beat* con *Elasticsearch* se debe indicar el certificado generado a través del módulo de seguridad en los ficheros de configuración de cada *Beat*.

3. *Wazuh agent*. Debido a que la infraestructura de *Wazuh* ya se encuentra implementada en el sistema SCADA, la securización de los *Wazuh agent* ya está implementada. Cabe destacar que en este trabajo la nube de *Wazuh* se encarga de securizar las conexiones con sus agentes.

4.5. Escenario alternativo

El despliegue del sistema se puede realizar en un escenario alternativo al propuesto. Esta opción se caracteriza por no ser escalable, ya que el sistema SEM se aloja en el equipo central sin ser alcanzable por el resto de máquinas del escenario, por lo que el entorno del SIEM se reduce a un solo equipo. En la Figura 4.5 se observa el escenario alternativo.

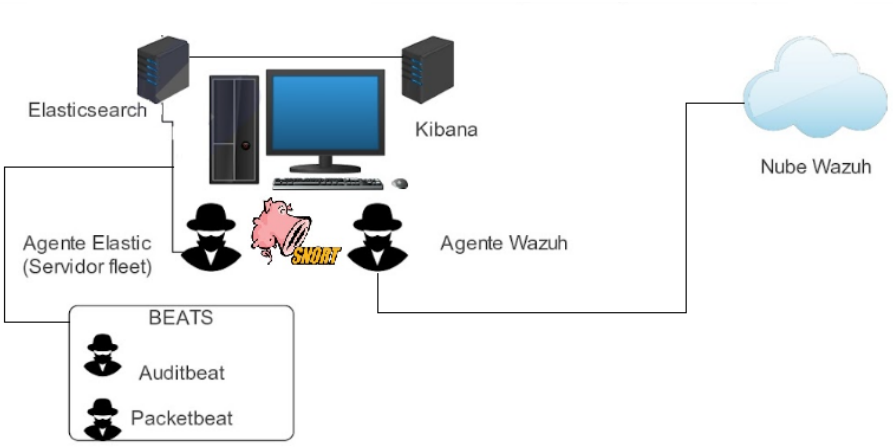

Figura 4.5: Diagrama de red del escenario alternativo

Capítulo 5

Pruebas y evaluación

Este capítulo está dedicado a estudiar la reacción del SIEM desplegado al someterlo a ataques de distinta naturaleza. A lo largo del capítulo se describen los ataques realizados y los resultados obtenidos. Todas las pruebas se realizan en el entorno alternativo descrito en la Sección 4.5.

5.1. Detección de *malware*

La integridad *Elastic security* permite al SIEM detectar software malicioso en el equipo securizado. Para analizar la reacción del SIEM se ha utilizado el software *Mimikatz* [1]. Se trata de una aplicación de código abierto que los atacantes usan para robar credenciales y escalar privilegios. Debido a la fama del software, la mayoría de antivirus y sistemas de seguridad lo detectan, por lo que se ha elegido para estas pruebas.

En la Figura 5.1 se muestran los resultados obtenidos al ejecutar la muestra de *malware* en modo prevención. Como se puede observar, el sistema detecta la ejecución del fichero *mimikatz.exe* catalogándolo como *malware* y declarándolo en cuarentena. La ruta en la que se deposita el fichero se observa en la información adjunta a la alerta.

La información obtenida en modo detección es la misma que la generada en modo prevención. La única diferencia es que no genera información relativa a la cuarentena. En la Figura 5.2 se muestran las alertas generadas por el SIEM en ambos modos de trabajo.

5.2. Técnicas de reconocimiento

El IDS *Snort* permite crear reglas que detecten paquetes relativos a las técnicas de reconocimiento. Las reglas implementadas en Snort siguen el esquema mostrado en la Figura 5.3. Las reglas implementadas en *Snort* son las siguientes:

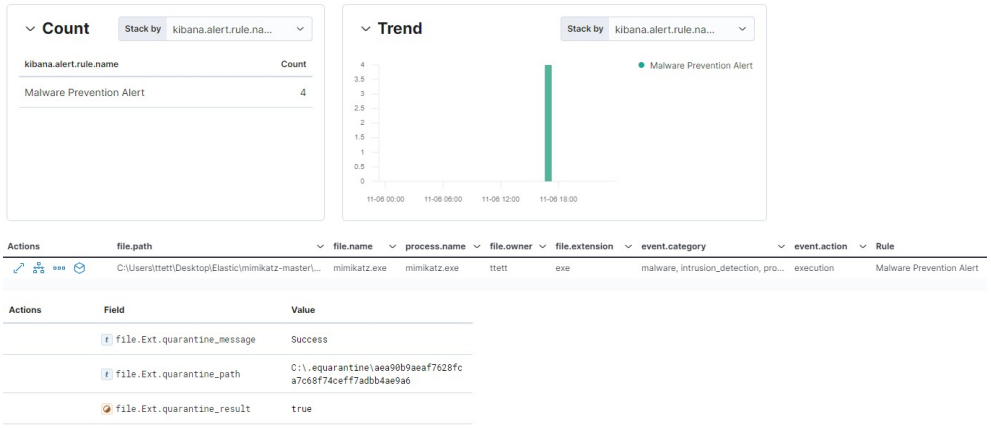

Figura 5.1: Resultados ejecución *mimikatz.exe* en modo prevención

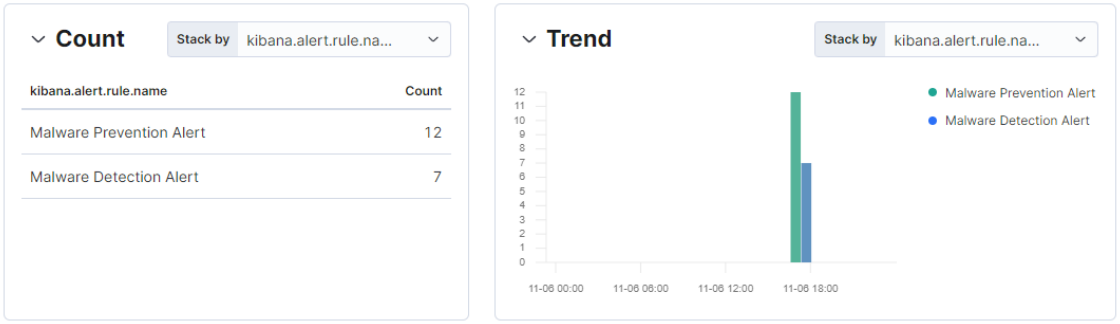

Figura 5.2: Alertas generadas por el SIEM en ambos modos de trabajo

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any
(msg: "portscan";flow:not_established;flags:S;sid:35000)

alert icmp $HOME_NET any -> $EXTERNAL_NET any
(msg: "UDP portscan"; icode:3; itype:3;sid:1000000)

alert icmp $EXTERNAL_NET any -> $HOME_NET any
(msg: "Ping attempt"; itype: 8; sid:1000002)
```

Para esta prueba se utilizan máquinas adicionales que adoptan el rol de atacantes y una máquina objetivo cuya dirección IP es la dirección 192.168.1.36. Se realizan ataques de escaneo de puertos sobre los protocolos de transporte TCP/UDP y envíos

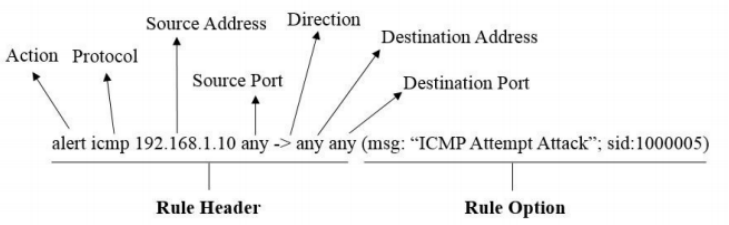

Figura 5.3: Ejemplo regla Snort [20]

de paquetes *ICMP request*. El escaneo de puertos sobre TCP se realiza fragmentando los paquetes en partes de 8 bytes, ya que es una técnica comúnmente usada por los atacantes para complicar el análisis del ataque.

Los resultados obtenidos de los escaneos de puertos se recogen en las Figuras 5.4 y 5.5. Como se puede observar, el sistema ha detectado el escaneo al puerto 53 a través del protocolo UDP. Se detecta que la dirección IP responsable del escaneo es la dirección 192.168.1.42. Por otra parte, el escaneo realizado a través del protocolo TCP se ha realizado a más de un puerto. Debido a que el sistema ha recibido muchos paquetes en diferentes puertos, se detecta un ataque de denegación de servicio, que se caracteriza por enviar un elevado número de paquetes para evitar el correcto funcionamiento de un servicio instalado en la máquina objetivo. En este caso, la dirección IP responsable del ataque es la dirección 192.168.1.42.

En la Figura 5.6 se muestra el resultado obtenido del envío de paquetes *ICMP Request*. Como se puede observar, el sistema alerta de la detección de cuatro paquetes *ICMP Request* (paquetes comúnmente utilizados para determinar el estado de una máquina). En este caso, la dirección IP que intenta reconocer un equipo del entorno securizado es la dirección 192.168.1.34. En el número de paquetes de red se puede observar que el equipo rastreado ha contestado a los paquetes enviados por el atacante.

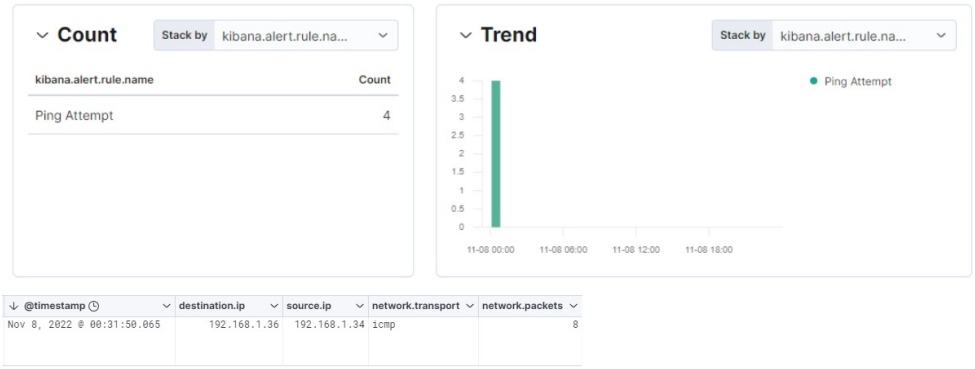

Figura 5.6: Alerta generada por la detección de paquetes *ICMP Request*

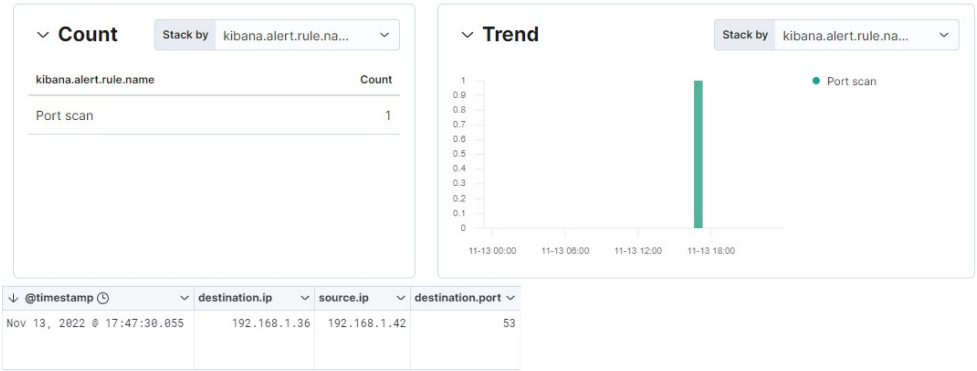

Figura 5.4: Escaneo del puerto 53 (UDP)

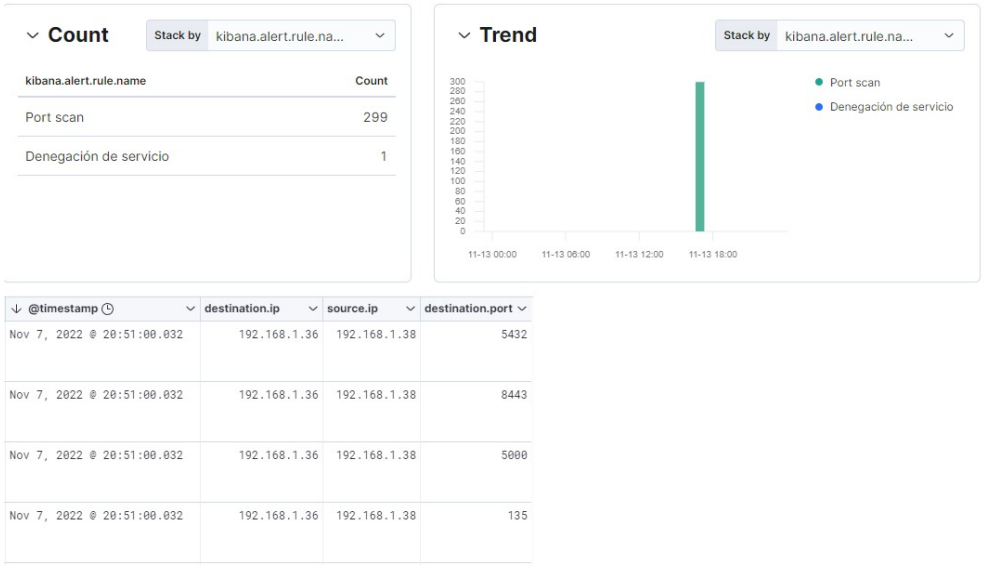

Figura 5.5: Escaneo de puertos TCP fragmentando paquetes

5.3. Inicios de sesión

Uno de los eventos de seguridad recolectados por la integridad *Elastic security* son los inicios de sesión fallidos por los usuarios del equipo securizado. Para forzar los intentos fallidos de inicio de sesión, se introducen de forma manual contraseñas erróneas múltiples veces.

En la Figura 5.7 se muestran los resultados obtenidos. Como se puede observar, el sistema ha detectado una gran cantidad de inicios de sesión fallidos seguidos de inicios de sesión exitosos en un período de tiempo corto.

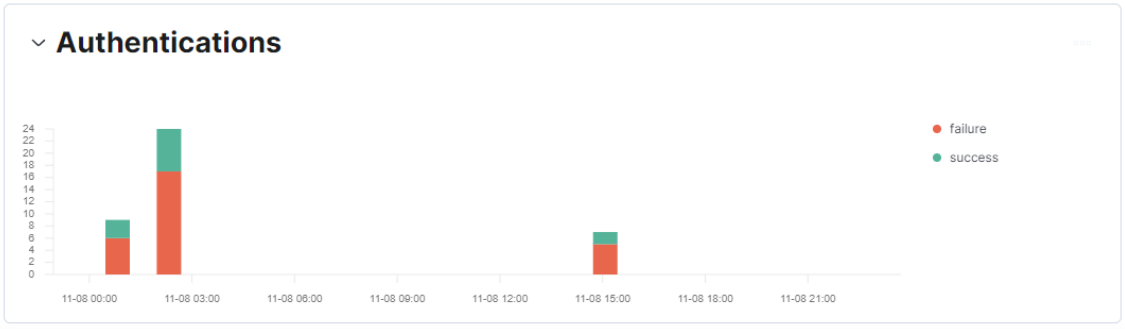

Figura 5.7: Inicios de sesión exitosos y fallidos

5.4. Integridad de ficheros

Uno de los *Beats* utilizados es *Auditbeat*, un agente especializado en auditar eventos relacionados con ficheros. Para esta prueba se fuerzan múltiples alertas creando, modificando el contenido, cambiando el nombre y borrando ficheros en la carpeta

SYSTEM32 de un equipo Windows. El fichero se crea con el nombre *Prueba.txt*, a lo largo de las pruebas el nombre cambia a *CambioElNombre.txt*.

En la Figura 5.8 se muestran las alertas generadas y la información recopilada por *Auditbeat*. Como se puede observar, el sistema ha detectado la creación del fichero *Prueba.txt* en el directorio SYSTEM32. En la alerta inmediatamente posterior, se detecta la modificación de los atributos del fichero *Prueba.txt* en el mismo directorio en el que se ha creado. La siguiente alerta muestra la creación de fichero .txt en el mismo directorio que el fichero *Prueba.txt*, en este caso con el nombre *CambioElNombre.txt*. Para finalizar, se detecta el borrado del fichero *Prueba.txt* en el directorio SYSTEM32.

Actions	file.path	file.owner	file.extension	event.category	event.action	Rule
	C:\Windows\System32\Prueba.txt	BUILTIN\Administradores	txt	file	created	Integridad de ficheros
Actions	file.path	file.owner	file.extension	event.category	event.action	Rule
	C:\Windows\System32\Prueba.txt	BUILTIN\Administradores	txt	file	updated, attributes_modified	Integridad de ficheros
Actions	file.path	file.owner	file.extension	event.category	event.action	Rule
	C:\Windows\System32\CambioElNombre.txt	BUILTIN\Administradores	txt	file	created	Integridad de ficheros
Actions	file.path	file.owner	file.extension	event.category	event.action	Rule
	C:\Windows\System32\Prueba.txt	—	—	file	deleted	Integridad de ficheros

Figura 5.8: Alertas generadas por eventos con ficheros

5.5. Detección de vulnerabilidades

Mediante el *Wazuh agent*, el SIEM es capaz de detectar vulnerabilidades en aplicaciones instaladas en el equipo securizado. Para esta prueba se ha instalado la versión 2.7.10 de *Python*. Se trata de una versión desactualizada con múltiples vulnerabilidades [22].

Vulnerabilities (9) — Export formatted

Filter or search

Name	Version	Architecture	Severity	CVE	CVSS2 Score	CVSS3 Score	Detection Time
Python 2.7.10 (64-bit)	2.7.10150	x64	High	CVE-2022-0391	5	7.5	Nov 8, 2022 @ 17:50:55.000
Python 2.7.10 (64-bit)	2.7.10150	x64	Medium	CVE-2021-3733	4	6.5	Nov 8, 2022 @ 17:50:55.000
Python 2.7.10 (64-bit)	2.7.10150	x64	Medium	CVE-2021-23336	4	5.9	Nov 8, 2022 @ 17:50:55.000
Python 2.7.10 (64-bit)	2.7.10150	x64	High	CVE-2019-9674	5	7.5	Nov 8, 2022 @ 17:50:55.000
Python 2.7.10 (64-bit)	2.7.10150	x64	Medium	CVE-2017-18207	4.3	6.5	Nov 8, 2022 @ 17:50:55.000
Python 2.7.10 (64-bit)	2.7.10150	x64	High	CVE-2017-17522	6.8	8.8	Nov 8, 2022 @ 17:50:55.000
Python 2.7.10 (64-bit)	2.7.10150	x64	Critical	CVE-2015-20107	10	9.8	Nov 8, 2022 @ 17:50:55.000
Wireshark 3.6.7 64-bit	3.6.7	x86	High	CVE-2022-3725	0	7.5	Nov 8, 2022 @ 17:50:55.000
Wireshark 3.6.7 64-bit	3.6.7	x86	Medium	CVE-2022-3190	0	5.5	Nov 8, 2022 @ 17:50:55.000

Figura 5.9: Vulnerabilidades detectadas

En la Figura 5.9 se muestran las vulnerabilidades detectadas por *Wazuh*. Entre las vulnerabilidades detectadas se observan no solo las esperadas, sino también las relativas al software de captura de paquetes de red *Wireshark*. Para evitar estas vulnerabilidades

detectadas sería recomendable actualizar los softwares vulnerables a la última versión disponible.

Capítulo 6

Conclusiones y trabajo futuro

Los sistemas SIEM son soluciones de seguridad basada en software con capacidad de monitorizar la red y los dispositivos conectados a ella. De esta manera, permiten detectar, responder y neutralizar ciberamenazas de prácticamente cualquier tipo.

En este trabajo se ha diseñado la estructura de un sistema SIEM utilizando como núcleo el software *Elastic*, así como múltiples agentes e integraciones que proporcionan los servicios mínimos de un SIEM. Además, se han añadido más funcionalidades al sistema mediante herramientas externas como *Snort* o *Wazuh*.

La intención original de este trabajo consistía en implementar las funciones del SIEM a través de las integraciones disponibles en *Kibana*. Sin embargo, el estudio previo realizado demuestra que los *Beats* siguen siendo la mejor opción. No obstante, no se descarta la posibilidad de migrar a las integraciones en futuras actualizaciones. Con respecto a las defensas dedicadas a proteger el entorno de *malware*, se ha demostrado que *Elastic* es una solución a tener en cuenta. Con respecto a la detección de *malware*, se depende totalmente de la *blocklist* predefinida por *Elastic security*. Esta lista no se puede editar con el plan gratuito, por lo que no se puede explotar todo el potencial de la integración sin asumir un gasto económico.

En este trabajo se ha utilizado el software *Snort* como principal herramienta para detectar los intentos de reconocimiento. Sin embargo, el preprocesador de *Snort* dedicado a detectar escaneos de puertos no detecta nada al realizarlos a puertos únicos o con un tiempo determinado entre los paquetes. Para evitar este punto débil se propone una regla que detecta todos los paquetes sospechosos, independientemente del número y del tiempo entre ellos. En las pruebas realizadas se ha comprobado que uno de los procesadores detecta un ataque de denegación de servicio. En este caso se ha detectado la denegación de servicio al realizar el escaneo de puertos fragmentando los paquetes en partes de 8 bytes.

El agente *Auditbeat* dota al SIEM de integridad de ficheros. En las pruebas realizadas se observa que detecta la creación, el borrado y la modificación de los ficheros. Sin

embargo, cambiar el nombre de un fichero existente se detecta como la creación de un nuevo fichero. Además, *Auditbeat* no detecta la modificación de los atributos de seguridad de un fichero para un usuario concreto.

Por último, Elastic carece de un método efectivo con el que generar informes de vulnerabilidades, pese a disponer de agentes recopiladores de información. En esta parte se ha decidido trabajar con el software Wazuh ya que tiene un escáner de vulnerabilidades integrado. Sin embargo, desde el punto de vista del análisis de la información no es el mejor diseño, ya que la información queda repartida en dos servidores distintos.

6.1. Trabajo futuro

Como trabajo futuro se planean distintas líneas de acción:

Investigación XDR. En este trabajo no se ha profundizado en la emergente tecnología XDR. Sin embargo, la presencia de algunos índices en Elasticsearch indican que la tecnología XDR y la solución *Elastic security* no son incompatibles. Esta tecnología permitiría identificar amenazas sofisticadas, realizar un seguimiento de las amenazas en varios componentes del sistema y perfeccionar la velocidad de detección.

Integración de *Logstash*. Aunque en este trabajo no se ha trabajado con *Logstash*, esta podría ser una herramienta a tener en cuenta. Gracias a *Logstash* se pueden introducir datos de muchas fuentes de información diferentes. En este caso, podría ser interesante analizar la sinergia entre *Logstash* y *Snort*.

Exploración de las opciones de Snort. En este trabajo se ha utilizado la herramienta *Snort* para detectar los escaneos de puertos. Sin embargo, *Snort* ofrece un gran abanico de opciones en sus reglas. Continuar estudiando dichas opciones es una manera de seguir mejorando el SIEM y la calidad de las alertas generadas.

Bibliografía

[1] Benjamin Delpy 'gentilwiki'. *Mimikatz*. [Online; https://github.com/gentilkiwi/mimikatz]. Accessed on November 22, 2022. Nov. de 2022.

[2] Marcello Cinque, Domenico Cotroneo y Antonio Pecchia. "Challenges and Directions in Security Information and Event Management (SIEM)". En: ago. de 2018. DOI: 10.1109/ISSREW.2018.00-24.

[3] A. Daneels y W. Salter. "What is SCADA?" En: *Conf. Proc. C* 991004 (1999). Ed. por D. Bulfone y A. Daneels, págs. 339-343.

[4] Elastic. *Auditbeat overview*. [Online; https://www.elastic.co/guide/en/beats/auditbeat/8.3/auditbeat-overview.html]. Accessed on November 22, 2022.

[5] Elastic. *Beats*. [Online; https://www.elastic.co/es/beats/]. Accessed on November 22, 2022. Nov. de 2022.

[6] Elastic. *Blocklist*. [Online; https://www.elastic.co/guide/en/security/8.3/blocklist.html]. Accessed on November 22, 2022. Nov. de 2022.

[7] Elastic. *Configure an integration policy for Endpoint and Cloud Security*. [Online; https://www.elastic.co/guide/en/security/8.3/configure-endpoint-integration-policy.html]. Accessed on November 22, 2022. Nov. de 2022.

[8] Elastic. *Detection and Alerts*. [Online; https://www.elastic.co/guide/en/security/current/detection-engine-overview.html]. Accessed on November 22, 2022. Nov. de 2022.

[9] Elastic. *Encrypt traffic between Elastic Agents, Fleet Server and Elasticsearch*. [Online; https://www.elastic.co/guide/en/fleet/8.3/secure-connections.html#_encrypt_traffic_between_elastic_agents_fleet_server_and_elasticsearch]. Accessed on November 22, 2022. Nov. de 2022.

[10] Elastic. *File Integrity Monitoring*. [Online; https://docs.elastic.co/integrations/fim]. Accessed on November 22, 2022. Nov. de 2022.

[11] Elastic. *Fleet and Elastic Agent overview*. [Online; https://www.elastic.co/guide/en/fleet/8.3/fleet-overview.html]. Accessed on November 22, 2022. Nov. de 2022.

[12] Elastic. *Logstash*. [Online; https://www.elastic.co/es/logstash/]. Accessed on November 22, 2022. Nov. de 2022.

[13] Elastic. *Network Packet Capture*. [Online; https://docs.elastic.co/integrations/network_traffic]. Accessed on November 22, 2022. Nov. de 2022.

[14] Elastic. *Packetbeat overview*. [Online; https://www.elastic.co/guide/en/beats/packetbeat/8.3/packetbeat-overview.html]. Accessed on November 22, 2022. Nov. de 2022.

[15] Elastic. *Set up Fleet server*. [Online; https://www.elastic.co/guide/en/fleet/current/fleet-server.html]. Accessed on November 22, 2022. Nov. de 2022.

[16] Elastic. *What is Elasticsearch?* [Online; https://www.elastic.co/es/what-is/elasticsearch]. Accessed on November 22, 2022. Nov. de 2022.

[17] Elastic. *What is Kibana?* [Online; https://www.elastic.co/es/what-is/kibana]. Accessed on November 22, 2022. Nov. de 2022.

[18] Dede Fadhilah y Marza Ihsan Marzuki. "Performance Analysis of IDS Snort and IDS Suricata with Many-Core Processor in Virtual Machines Against Dos/DDoS Attacks". En: *2020 2nd International Conference on Broadband Communications, Wireless Sensors and Powering (BCWSP)*. 2020, págs. 157-162.

[19] HASecuritySolutions. *VulnWhisperer*. [Online; https://github.com/HASecuritySolutions/VulnWhisperer]. Accessed on November 22, 2022. Nov. de 2022.

[20] Nattawat Khamphakdee, Nunnapus Benjamas y Saiyan Saiyod. "Improving Intrusion Detection System based on Snort rules for network probe attack detection". En: *2014 2nd International Conference on Information and Communication Technology (ICoICT)*. 2014, págs. 69-74. DOI: 10.1109/ICoICT.2014.6914042.

[21] Lockheed Martin. *Intelligence-Driven Computer Network DefenseInformed by Analysis of Adversary Campaigns andIntrusion Kill Chains*. [Online; https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Intel-Driven-Defense.pdf]. Accessed on November 22, 2022. Nov. de 2022.

[22] MITRE. *Search CVE List*. [Online; https://cve.mitre.org/cve/search_cve_list.html]. Accessed on November 22, 2022. Nov. de 2022.

[23] Nmap. *Técnicas de sondeo de puertos*. [Online; https://nmap.org/man/es/man-port-scanning-techniques.html]. Accessed on November 22, 2022. Nov. de 2022.

[24] Robert Radvanovsky y Jacob Brodsky. *Handbook of SCADA/Control Systems*. Taylor y Francis Group, 2016, pág. 441. ISBN: 978-1498717076.

[25] Pascal Roques. *UML in Practice. The Art of Modeling Software Systems Demonstrated through Worked Examples and Solutions*. Wiley, 2004, pág. 312. ISBN: 978-0470848319.

[26] Scipag. *Script vulners*. [Online; https://github.com/scipag/vulscan]. Accessed on November 22, 2022. Nov. de 2022.

[27] Peter Shekindo. *How to implement logging in your REST service by using Elasticsearch*. [Online; https://dev.to/clickpesa/how-to-implement-logging-in-your-rest-service-by-using-elasticsearch-part-2-21fm]. Accessed on November 22, 2022. Nov. de 2022.

[28] Michael Sikorski. *Practical Malware Analysis. The Hands-On Guide to Dissecting Malicious Software*. No Starch Press, 2012. ISBN: 978-1593272906.

[29] William Stallings. *Network Security Essentials : Applications and Standards, International Edition Applications and Standards*. Pearson Education, Limited, 2013, pág. 448. ISBN: 9780273793366.

[30] Tenable. *Nessus Essentials*. [Online; `https://es-la.tenable.com/products/nessus/nessus-essentials`]. Accessed on November 22, 2022. Nov. de 2022.

[31] Brian Van Leeuwen, William Stout y Vincent Urias. "MTD assessment framework with cyber attack modeling". En: *2016 IEEE International Carnahan Conference on Security Technology (ICCST)*. 2016, págs. 1-8.

[32] Y. S. Vasiliev, P. D. Zegzhda y V. I. Kuvshinov. "Modern Problems of cibersecurity". En: *Nonlinear Phenomena in Complex Systems* 17.3 (2014) (mayo de 2014), págs. 210-214. ISSN: 1561-4085.

[33] Wazuh. *Vulnerability detection, How it works?* [Online; `https://documentation.wazuh.com/current/user-manual/capabilities/vulnerability-detection/how-it-works.html`]. Accessed on November 22, 2022. Nov. de 2022.

[34] Wazuh. *Wazuh agent*. [Online; `https://documentation.wazuh.com/current/installation-guide/wazuh-agent/index.html`]. Accessed on November 22, 2022. Nov. de 2022.

Anexos A

Horas de trabajo

En la Figura A.1 se muestra el tiempo invertido en cada una de las áreas de este trabajo. En la Figura A.2 se muestra un diagrama de Gantt ilustrando la evolución temporal. Como se puede observar, se comienza con un estudio general de los sistemas SIEM y de los sistemas SCADA. Es durante este estudio donde se formalizan las funcionalidades del sistema SIEM. Posteriormente, se estudian los software *Elastic* y *Wazuh*. Tras conocer el software implantado en el sistema SCADA y el software donde se desarrolla el sistema SIEM, se procede al estudio de la documentación y a la instalación de la *Elastic Stack*, así como de sus agentes y demás tecnología. Después, se configuran las reglas a través de la integración *Elastic Security* y se procede a realizar las pruebas relativas a la detección de *malware*. Tras detectar que el agente *Packetbeat* no era lo suficientemente completo, se procede a estudiar la documentación de *Snort*. Tras su despliegue y la configuración de las reglas de detección contra técnicas de reconocimiento mediante escaneos de red, se vuelven a realizar las pruebas pertinentes. A continuación, se realizan las pruebas para estudiar la integridad de ficheros proporcionada por el agente *Auditbeat* y las relativas a la detección de inicios de sesión. Para estudiar el detector de vulnerabilidades, se realizan pruebas con múltiples soluciones hasta elegir el escáner de vulnerabilidades integrado en *Wazuh*. Para comprobar la escalabilidad del sistema, se procede a desplegar el sistema SIEM en un escenario escalable y a securizar las conexiones. Finalmente, se procede la elaboración de la memoria.

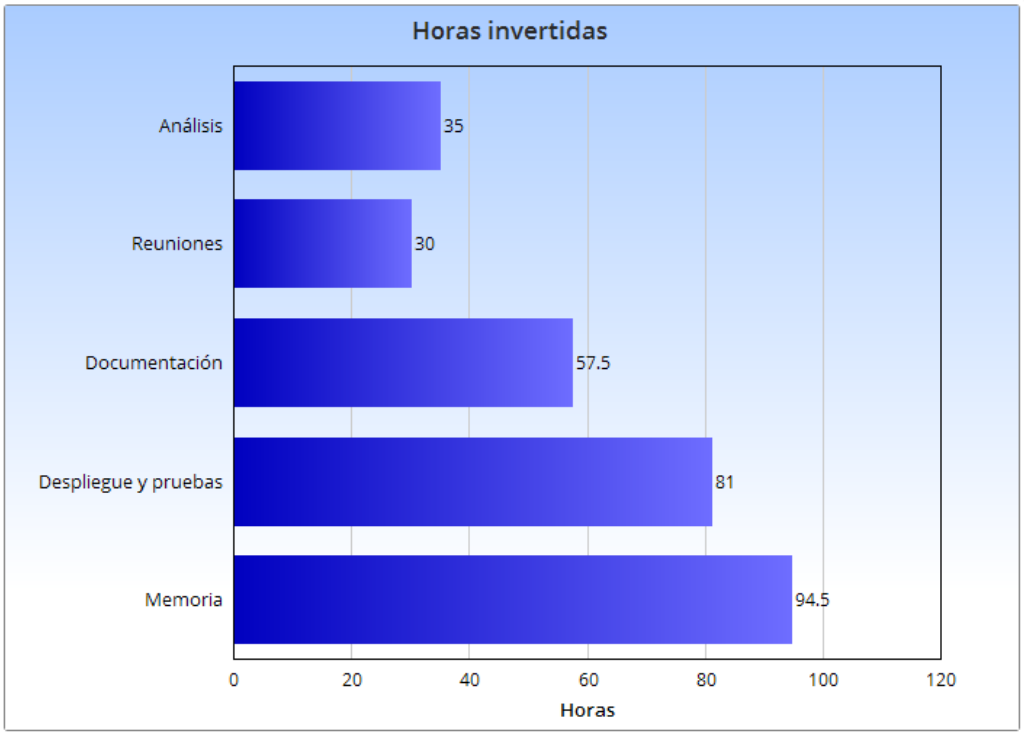

Figura A.1: Horas invertidas (total: 298 horas)

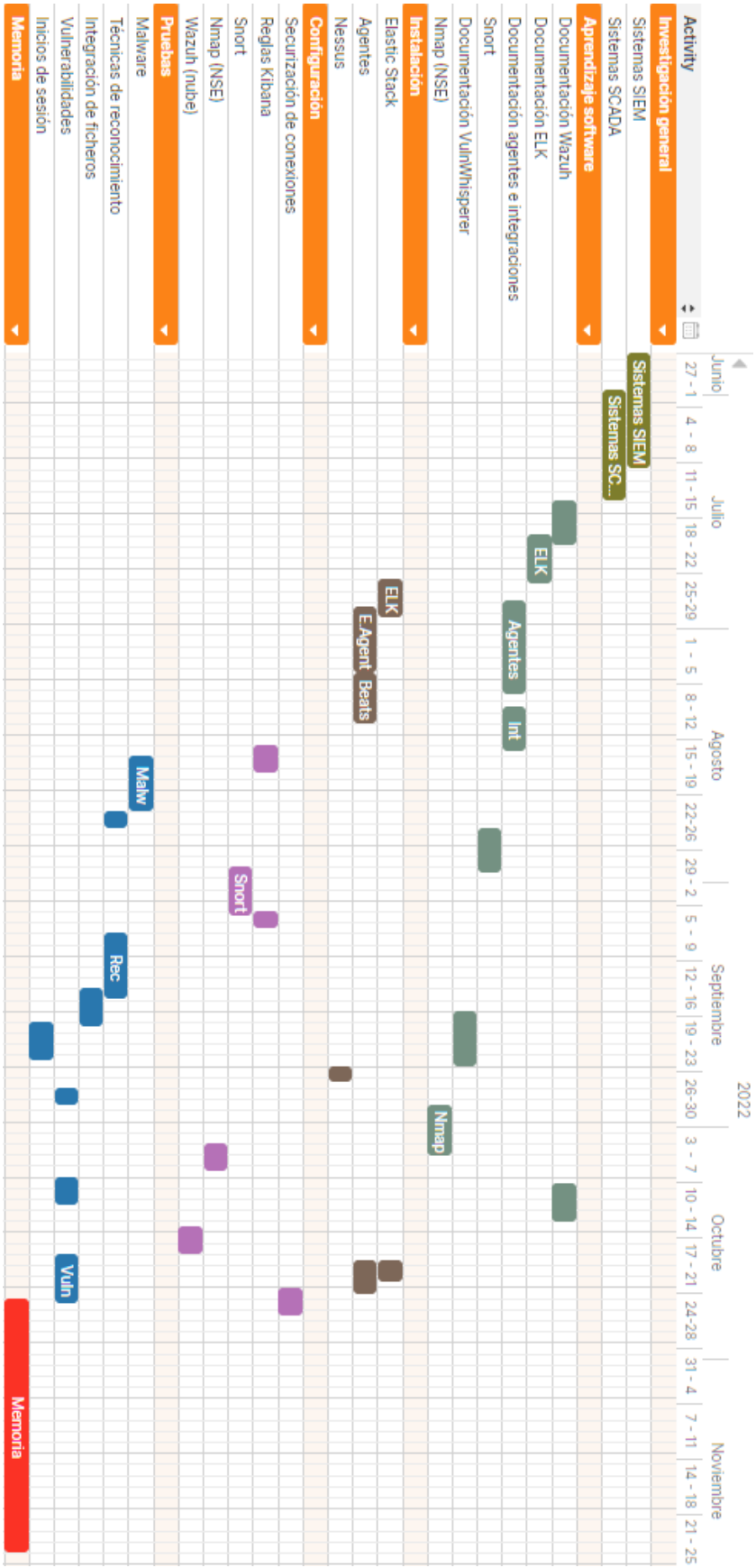

Figura A.2: Diagrama de Gantt