Universidad de Valladolid


Escuela de Ingeniería Informática

**TRABAJO FIN DE GRADO**

Grado en Ingeniería Informática
Mención en Ingeniería de Software

# Implementación de una infraestructura SIEM para la monitorización y detección de amenazas cibernéticas

**Autor:** Dario Merino Porras

Universidad de Valladolid


Escuela de Ingeniería Informática

**TRABAJO FIN DE GRADO**

Grado en Ingeniería Informática
Mención en Ingeniería de Software

# Implementación de una infraestructura SIEM para la monitorización y detección de amenazas cibernéticas

**Autor:** Dario Merino Porras

**Tutor:** Valentín Cardeñoso Payo

# Agradecimientos

Me gustaría agradecer a mi familia, en especial a mi hermana y a mi madre, por el apoyo incondicional que me han brindado a lo largo de toda la carrera. Su confianza y ánimo han sido fundamentales en este camino.

Quiero expresar también mi agradecimiento a todos los profesores de la Escuela de Ingeniería Informática de Valladolid, gracias a quienes he podido descubrir y profundizar en distintos aspectos del mundo de la informática. En particular, me gustaría dar las gracias a mi tutor del TFG por su orientación y apoyo durante estos meses.

Finalmente, gracias a todas las personas que han estado presentes en esta etapa y han hecho posible la culminación de este proyecto.

**Resumen**

El incremento de los ciberataques en los últimos años ha convertido la protección de los sistemas informáticos en un aspecto fundamental para garantizar la seguridad y la continuidad operativa de las organizaciones. La implementación de sistemas capaces de prevenir, detectar y responder en tiempo real ante posibles amenazas se ha vuelto una necesidad crítica para reducir al mínimo el impacto de los incidentes de seguridad.

El objetivo principal de este proyecto es diseñar e implementar una arquitectura SIEM capaz de detectar amenazas cibernéticas en un entorno simulado. Para ello, se ha realizado una investigación previa que ha permitido identificar las herramientas más adecuadas para construir esta infraestructura. Posteriormente, se ha realizado su configuración e implementación. En concreto, el SIEM desarrollado está compuesto por Kafka y la pila ELK, que son tecnologías ampliamente utilizadas en entornos de ciberseguridad para el procesamiento y visualización de eventos.

Como fuentes de datos se han integrado Winlogbeat, encargado de recolectar eventos de sistemas Windows, y Zeek, un sistema de detección de intrusiones que analiza el tráfico de red. Los eventos recolectados por estas dos herramientas se envían al SIEM, donde son procesados, almacenados y analizados con el fin de detectar comportamientos sospechosos.

Una vez desplegada la infraestructura, se han llevado a cabo diversas pruebas con el fin de comprobar que el sistema creado es capaz de recolectar, procesar y analizar logs, generando alertas automáticas ante la detección de comportamientos potencialmente maliciosos en los equipos monitorizados.

Este proyecto permite comprender cómo se construye desde cero una infraestructura SIEM y la importancia que tiene en la detección temprana y respuesta frente a amenazas cibernéticas.

**Abstract**

The increase in cyberattacks in recent years has made the protection of computer systems a fundamental aspect for ensuring the security and operational continuity of organizations. Implementing systems capable of preventing, detecting, and responding to potential threats in real time has become critical to minimizing the impact of security incidents.

The main objective of this project is to design and implement a SIEM architecture capable of detecting cyberthreats in a simulated environment. To this end, preliminary research was conducted to identify the most appropriate tools for building this infrastructure. It was subsequently configured and implemented. Specifically, the SIEM developed is composed of Kafka and the ELK stack, technologies widely used in cybersecurity environments for event processing and visualization.

Winlogbeat, which collects events from Windows systems, and Zeek, an intrusion detection system that analyzes network traffic, were integrated as data sources. The events collected by these two tools are sent to the SIEM, where they are processed, stored, and analyzed to detect suspicious behavior.

Once the infrastructure was deployed, various tests were conducted to verify that the system was capable of collecting, processing, and analyzing logs, generating automatic alerts when potentially malicious behavior is detected on the monitored devices.

This project provides an understanding of how a SIEM infrastructure is built from scratch and its importance in early detection and response to cyber threats.

# Índice general

# Índice de cuadros

# Índice de figuras

# Índice de listados

# Capítulo 1

# Introducción

En los últimos años, se ha registrado un aumento significativo de los ciberataques. Según datos del Foro Económico Mundial, nueve de cada diez organizaciones sufrieron al menos un ataque cibernético en 2024. Respecto al año 2023, se ha registrado un incremento del 75 % en ciberataques, y no solo ha aumentado el número de ciberataques, también su complejidad. Esto se debe a la implementación de inteligencia artificial lo que ha permitido realizar ataques más sofisticados y personalizados.

Existen muchos tipos de ciberataques, pero todos tienen en común que pueden generar pérdidas económicas colosales. Se estima que para el año 2025, los costos asociados al crimen cibernético superarán los 10,5 billones de dólares. Es por esto que la implementación de sistemas que permitan prevenir, detectar y responder frente a ataques se ha convertido en un pilar de la seguridad informática.

Aquí es donde entra en juego la arquitectura SIEM (Security Information and Event Management).

La implementación de una arquitectura SIEM es esencial, ya que permite a las organizaciones detectar, analizar y responder a incidentes de seguridad en tiempo real, gracias a la recopilación y análisis de eventos procedentes de diversas fuentes.

En este proyecto se pretende implementar una arquitectura SIEM que recoja y analice logs procedentes de diversas fuentes para detectar patrones asociados a posibles amenazas, y tener la capacidad de responder frente a estas.

## 1.1 Motivación

El ámbito de la ciberseguridad está en constante evolución, y cada vez se observan ataques más sofisticados. A esto se le suma la aparición de la inteligencia artificial, lo que permite a los atacantes automatizar ataques más complejos y personalizados, poniendo en riesgo las infraestructuras de las organizaciones.

Ante esta situación, se necesitan sistemas capaces de prevenir, detectar y responder ante ataques. La implementación de un sistema SIEM es una solución clave para hacer frente a posibles amenazas. El desarrollo de una infraestructura escalable y eficiente, como la que se propone en este proyecto, es esencial para asegurar la protección de una red.

Dado mi enorme interés personal por la ciberseguridad, he decidido desarrollar más a fondo un tema relacionado con el Blue Team, en concreto la implementación de una infraestructura SIEM, que tiene un papel

fundamental en la monitorización, análisis y respuesta a incidentes de ciberseguridad. Esta implementación es esencial en el SOC (Security Operations Center) de una organización. En este centro operativo, se monitoriza la red en tiempo real, lo que permite una vigilancia constante de los equipos para detectar y responder rápidamente ante cualquier actividad sospechosa.

Comprender cómo funciona esta arquitectura SIEM es un buen punto de partida para introducirse en el mundo de la ciberseguridad, ya que permite entender cómo funciona una de las partes más importantes del Blue Team. Este es el motivo principal por el cual escogí este tema para desarrollar el proyecto. Para entender la importancia que puede llegar a tener un SIEM es fundamental conocer ¿Qué es un SIEM? ¿Por qué tiene un rol crucial en el ámbito de la ciberseguridad?

# Capítulo 2

# Objetivos y Alcance

## 2.1 Objetivos

El objetivo principal de este trabajo es diseñar, implementar y configurar una infraestructura SIEM para la monitorización y detección de amenazas en un entorno empresarial simulado.

Esta infraestructura tiene la capacidad de recopilar, procesar y analizar los registros generados por los equipos monitorizados, con la finalidad de detectar posibles amenazas de seguridad y poder responder ante incidentes reales.

## 2.2 Objetivos específicos

Los objetivos específicos de este trabajo son los siguientes:

- Estudio de las herramientas implicadas (Winlogbeat, Zeek, Kafka, ELK (Elasticsearch, Logstash, Kibana), Caldera).
- Instalación, configuración y dockerización de la infraestructura (Kafka, ELK).
- Instalación y configuración del software encargado de generar eventos.
- Instalación y configuración de Caldera para la simulación de ataques.
- Creación y configuración de un servidor Active Directory, creación de un dominio e incorporación de al menos un equipo miembro.
- Registrar eventos (logs) en los equipos monitorizados.
- Centralización de logs usando Apache Kafka.
- Implementación de la pila ELK para procesar, analizar y visualizar los logs centralizados.
- Implementar políticas ILM (Information Lifecycle Management) para gestionar el almacenamiento de logs.
- Creación de al menos 3 reglas de detección para identificar patrones asociados a posibles amenazas.

- Creación de una interfaz para visualizar los logs y alertas.
- Simulación de ataques con Caldera.
- Verificar que las reglas de detección creadas generan alertas en Kibana al simular ataques.

# Capítulo 3

# Metodología

La metodología SCRUM es un marco ágil para la gestión de proyectos que promueve la flexibilidad y colaboración. Esta metodología de trabajo es usada en proyectos en los que los requisitos pueden evolucionar a lo largo del tiempo o donde pueden surgir nuevos objetivos durante el desarrollo. Permite adaptarse rápidamente a nuevos cambios y está diseñada para buscar la eficiencia en cada uno de sus ciclos iterativos, conocidos como sprints. Un sprint es un periodo de tiempo corto y definido, donde se desarrollan pequeñas entregas del producto planificadas previamente, lo que permite avanzar de manera continua.

El motivo por el que escogí la metodología SCRUM para planificar el desarrollo de mi proyecto es porque permite adaptarse a objetivos cambiantes y nuevas necesidades durante el proyecto. Su metodología basada en ciclos iterativos (Sprints), permite la adaptación continua de la planificación.

## 3.1 Artefactos en SCRUM

Los artefactos son los documentos o herramientas que SCRUM utiliza para gestionar el desarrollo del proyecto. Cada artefacto tiene un propósito específico dentro del proyecto y todos ellos garantizan que el proyecto se desarrolle correctamente.

- **Product Backlog:** Es una lista priorizada de tareas que deben completarse para cumplir con los objetivos del proyecto. Esta lista puede ser modificada durante el desarrollo del proyecto, permitiendo adaptarse a los cambios que puedan surgir.
- **Sprint Backlog:** Para cada sprint, se seleccionan las tareas del Product Backlog que deben completarse en ese ciclo específico. El Sprint Backlog es una lista de tareas que el equipo debe realizar durante el sprint.
- **Incremento:** Al final de cada sprint, se obtiene una versión del producto que contiene los avances realizados durante ese ciclo. Este incremento tiene que ser funcional y entregable.

## 3.2 Roles en SCRUM

Los roles en SCRUM son esenciales para el correcto desarrollo del proyecto. Cada rol cuenta con unas responsabilidades, que garantizan una correcta implementación de la metodología ágil:

- **Product Owner:** Es la persona encargada de definir y establecer una prioridad para los requisitos del proyecto, gestionando las tareas del Product Backlog.
- **Scrum Master:** Es la persona encargada de asegurar que el equipo siga las prácticas y metodologías de SCRUM, también elimina obstáculos que pueden afectar en el desarrollo del proyecto.
- **Equipo de Desarrollo:** Está compuesto por las personas que realizan las tareas necesarias para desarrollar el producto.

## 3.3 Eventos en SCRUM

Los eventos en SCRUM son actividades planificadas que ayudan a estructurar los ciclos de trabajo. Estos eventos ayudan a aumentar la coordinación entre todos los miembros del proyecto y juegan un papel clave a la hora de revisar y ajustar el progreso del proyecto.

- **Sprint Planning:** Es una reunión en la que se definen las tareas que se realizarán durante el siguiente sprint.
- **Daily Scrum:** Es una reunión diaria en la que cada miembro del equipo de desarrollo comunica su progreso, problemas que ha tenido y sus próximos pasos.
- **Sprint Review:** Es una reunión al final de cada sprint en la que el equipo de desarrollo expone el trabajo realizado al Product Owner para recibir retroalimentación.
- **Sprint Retrospective:** Es una reunión que también se lleva a cabo al final de cada sprint para reflexionar sobre lo que se ha hecho bien y lo que se puede mejorar.

## 3.4 División del trabajo en SCRUM

Una división del trabajo correcta y realista es esencial en SCRUM para garantizar que el equipo de desarrollo pueda avanzar de forma ordenada. El trabajo se organiza a través de elementos como las historias de usuario y las tareas, que permiten dividir los requisitos del proyecto en actividades más pequeñas. Esto ayuda a la planificación del trabajo y asegura que cada miembro del equipo tenga clara su labor durante todos los ciclos del proyecto.

- **Historias de usuario:** Breve descripción de funcionalidades o características que el sistema debe tener, desde la perspectiva del usuario final.
- **Tareas:** Es la unidad más pequeña de trabajo y son asignadas durante el Sprint Planning a una única persona para ser completada a lo largo del Sprint.

## 3.5 Adaptación particular

Dado que este proyecto va a ser desarrollado por una única persona, ha sido necesario adaptar la metodología Scrum a un entorno individual. El alumno asume los roles de Scrum Master y de equipo de desarrollo, mientras que el tutor Valentín Cardeñoso Payo cumple la función de Product Owner, supervisando la evolución del proyecto y validando los entregables más relevantes.

Al tratarse de un proyecto desarrollado por una única persona, no se realizan reuniones Daily Scrum, ya que no existe un equipo con el que coordinar el progreso diario. Por ese mismo motivo, las Sprint Planning han sido

sustituidas por una planificación autónoma, en la que se definen las tareas y los criterios de aceptación de cada sprint.

## 3.6 Product Backlog

| ID | DESCRIPCIÓN |
|---|---|
| 1 | Investigación de las herramientas implicadas |
| 2 | Diseño y configuración de la red |
| 3 | Instalación y dockerización de Apache Kafka |
| 4 | Instalación y dockerización de la pila ELK |
| 5 | Instalación y configuración de las herramientas encargadas de generar logs |
| 6 | Configuración de Kafka para la centralización de logs |
| 7 | Configuración de ELK para la recepción, procesamiento, análisis y visualización de los logs |
| 8 | Creación e implementación de políticas ILM y reglas de detección de posibles amenazas en Kibana |
| 9 | Creación de dashboards en Kibana para la visualización de logs y alertas |
| 10 | Instalación de CALDERA (Software para simular ataques) |
| 11 | Simulación de ataques con CALDERA |

Cuadro 3.1: Product Backlog

## 3.7 Planificación inicial de los sprints

| Planificación inicial del proyecto | |
|---|---|
| **Fecha de inicio** | 25 de Noviembre de 2024 |
| **Fecha de finalización** | 17 de Marzo de 2025 |
| **Semanas estimadas** | 16 |
| **Duración del proyecto** | 300 horas |
| **Horas/Semana** | 18,75 |
| **Horas/Día** | 2,7 horas |
| **Duración de los sprints** | Variable en función del Sprint |

Cuadro 3.2: Planificación inicial del proyecto

| Sprint | Duración | Fecha inicio | Fecha finalización |
|---|---|---|---|
| Sprint 1 | 16 Horas | 25/11/2024 | 30/11/2024 |
| Sprint 2 | 16 Horas | 01/12/2024 | 6/12/2024 |
| Sprint 3 | 14 Horas | 07/12/2024 | 12/12/2024 |
| Sprint 4 | 22 Horas | 13/12/2024 | 21/12/2024 |
| Sprint 5 | 25 Horas | 22/12/2024 | 31/12/2024 |
| Sprint 6 | 24 Horas | 01/01/2025 | 09/01/2025 |
| Sprint 7 | 22 Horas | 10/01/2025 | 18/01/2025 |
| Sprint 8 | 150 Horas | 19/01/2025 | 16/03/2025 |

Cuadro 3.3: Fechas estimadas de los Sprints

## 3.8 Sprints

### SPRINT 1

En este sprint se llevarán a cabo las tareas de investigación de las herramientas implicadas en el desarrollo del proyecto.

| Historia 1: Investigación de herramientas implicadas | |
|---|---|
| **Descripción** | Investigar las herramientas implicadas para comprender su funcionamiento y poder implementarlas de forma correcta en el proyecto. |
| **Criterios de aceptación** | Conocer el funcionamiento de las herramientas y el rol que tienen en el sistema. |
| **Tareas** | Investigar y seleccionar las herramientas con las que se va a desarrollar el proyecto.<br>Investigar el funcionamiento de cada herramienta dentro del sistema. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 16 Horas |
| **Dependencias** | Ninguna. |

### SPRINT 2

En este sprint se diseñará y configurará la red de máquinas virtuales que alojarán las herramientas necesarias para montar la infraestructura del proyecto, asegurando que todas las VMs puedan comunicarse entre sí sin ningún problema. Además, se creará y configurará un servidor Active Directory, así como un usuario miembro de este. El servidor simulará el entorno de una empresa y el usuario representará a un trabajador de la misma.

| Historia 2: Diseño y configuración de la red | |
|---|---|
| **Descripción** | Diseñar y configurar la red de máquinas virtuales que alojarán las herramientas implicadas y asegurar que las máquinas puedan comunicarse correctamente gracias a la red configurada. |

| | |
|---|---|
| **Criterios de aceptación** | La red virtual ha sido correctamente creada.<br>Todas las máquinas virtuales de la red pueden comunicarse entre sí. |
| **Tareas** | Diseñar la red.<br>Configurar la red.<br>Verificar que la conexión entre máquinas. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 8 Horas |
| **Dependencias** | Ninguna. |

| **Historia 3: Creación y configuración de un servidor Active Directory** | |
|---|---|
| **Descripción** | Creación y configuración de un servidor AD, promoción del servidor a controlador de dominio y adición de un usuario al dominio creado. |
| **Criterios de aceptación** | El servidor AD ha sido correctamente creado y se ha realizado la promoción a controlador de dominio.<br>El usuario ha sido creado correctamente en el dominio y puede iniciar sesión con sus credenciales de AD. |
| **Tareas** | Instalación y promoción del servidor AD.<br>Creación de usuario y configuración de la máquina del usuario.<br>Verificación del usuario creado iniciando sesión. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 8 Horas |
| **Dependencias** | Depende de la Historia 2. |

## SPRINT 3

En este sprint se llevará a cabo la instalación, dockerización y configuración de las herramientas Kafka y ELK.

| **Historia 4: Instalación y dockerización de Apache Kafka** | |
|---|---|
| **Descripción** | Instalar y dockerizar Apache Kafka para centralizar los logs generados en los equipos monitorizados. |
| **Criterios de aceptación** | Kafka se ejecuta de forma correcta.<br>Es posible acceder a los servicios de Kafka desde otras máquinas.<br>La configuración de Kafka es correcta y permite la creación de topics.<br>La configuración de Kafka es correcta y permite la recepción y consumición de logs de los tópicos. |
| **Tareas** | Descargar e instalar Docker.<br>Configurar el archivo docker-compose.yml.<br>Levantar el contenedor que aloja Kafka.<br>Crear un topic de prueba.<br>Generar logs que se almacenen en el tópico de prueba.<br>Consumir los logs almacenados en el tópico de prueba desde otra máquina. |
| **Prioridad** | Alta |

| | |
|---|---|
| **Estimación de esfuerzo** | 7 Horas |
| **Dependencias** | Depende de la Historia 2. |

| **Historia 5: Instalación y dockerización de ELK** | |
|---|---|
| **Descripción** | Instalar y dockerizar ELK para poder procesar y visualizar los logs que se almacenan en Kafka. |
| **Criterios de aceptación** | Elasticsearch, Logstash y Kibana se ejecutan correctamente dentro de sus respectivos contenedores Docker.<br>Todos los servicios están configurados y se comunican entre ellos de forma correcta.<br>Es posible acceder a Kibana desde un navegador. |
| **Tareas** | Descargar e instalar Docker.<br>Configurar los archivos docker-compose.yml correspondientes para que Elasticsearch, Logstash y Kibana se ejecuten en contenedores separados.<br>Verificar que Kibana es accesible desde un navegador. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 7 Horas |
| **Dependencias** | Depende de las Historias 2 y 3. |

## SPRINT 4

En este sprint se instalarán y configurarán Winlogbeat y Zeek, que son las herramientas encargadas de generar logs en los equipos monitorizados. También se establecerá la configuración necesaria en Kafka para recibir los logs de las diversas fuentes en un mismo sitio y tener la información centralizada.

| **Historia 6: Instalación y configuración de las herramientas encargadas de generar logs** | |
|---|---|
| **Descripción** | Instalación de las herramientas encargadas de generar logs y configuración de las mismas para que envíen los logs generados a Kafka. |
| **Criterios de aceptación** | Las herramientas de registro de eventos están instaladas y configuradas.<br>Las herramientas generan logs correctamente. |
| **Tareas** | Instalar las herramientas.<br>Configurar las herramientas.<br>Hacer pruebas para verificar que se están generando logs. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 17 Horas |
| **Dependencias** | Depende de las Historias 2 y 3. |

| **Historia 7: Configuración de Kafka para centralizar la recepción de logs** | |
|---|---|
| **Descripción** | Configurar Kafka para que reciba todos los logs generados por las múltiples fuentes que se usarán en el proyecto. |
| **Criterios de aceptación** | La configuración de Kafka permite recibir logs de diversas fuentes. |

| | |
|---|---|
| | Los logs almacenados son accesibles desde ELK. |
| **Tareas** | Crear los topics necesarios para recibir los logs de las diversas fuentes.<br>Verificar que los logs generados en las pruebas anteriores están llegando de forma correcta a Kafka. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 5 Horas |
| **Dependencias** | Depende de la Historia 3 y 5. |

## SPRINT 5

Este sprint está completamente dedicado a configurar Elasticsearch, Logstash y Kibana para que sean capaces de consumir, procesar, almacenar y visualizar los logs generados en los equipos monitorizados.

| **Historia 8: Configuración de ELK** | |
|---|---|
| **Descripción** | Configurar ELK para recibir, procesar y visualizar los logs almacenados en Kafka. |
| **Criterios de aceptación** | Logstash recibe los logs de Kafka y los procesa de forma correcta.<br>Elasticsearch almacena los logs.<br>Kibana permite visualizar los logs procesados. |
| **Tareas** | Configurar Logstash para consumir y procesar los logs de Kafka.<br>Configurar Elasticsearch para almacenar los logs procesados por Logstash.<br>Configurar Kibana para la visualización de logs desde el navegador. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 25 Horas |
| **Dependencias** | Depende de las Historias 5 y 6. |

## SPRINT 6

Durante este sprint, se implementarán las ILM con las que se gestionan los ciclos de vida de los logs. También se crearán reglas de detección de posibles amenazas en Kibana. Por último, en este sprint se crearán los dashboards con los que se podrá monitorizar el estado de la red de forma clara y donde se podrán ver las alertas generadas en base a las reglas de detección creadas.

| **Historia 9: Implementación de políticas ILM y reglas de detección en Kibana** | |
|---|---|
| **Descripción** | Creación e implementación de políticas ILM y reglas de detección de posibles amenazas en Kibana para poder detectar posibles incidentes de seguridad. |
| **Criterios de aceptación** | Las políticas ILM están implementadas de forma correcta.<br>Las reglas de detección deben estar implementadas de forma correcta. |
| **Tareas** | Creación y configuración de políticas ILM.<br>Creación e implementación de reglas de detección. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 4 Horas |

| | |
|---|---|
| **Dependencias** | Depende de la Historia 7. |

| **Historia 10: Creación de dashboard sobre Zeek en Kibana** | |
|---|---|
| **Descripción** | Creación de dashboard para visualizar logs y alertas generados por la herramienta Zeek, permitiendo de forma visual monitorizar la red y filtrar búsquedas. |
| **Criterios de aceptación** | Los dashboards en Kibana permiten visualizar los logs y las alertas generadas por Zeek de forma clara.<br>Se pueden realizar búsquedas filtradas. |
| **Tareas** | Selección de los campos relevantes de los logs generados por Zeek.<br>Creación de gráficos que muestren la información relevante de forma clara y permita filtrar búsquedas.<br>Creación de una vista para visualizar las alertas referentes a Zeek. |
| **Prioridad** | Media |
| **Estimación de esfuerzo** | 12 Horas |
| **Dependencias** | Depende de la Historia 7. |

| **Historia 11: Creación de dashboard sobre winlogbeat en Kibana** | |
|---|---|
| **Descripción** | Creación de dashboard para visualizar logs y alertas generados por la herramienta winlogbeat, permitiendo de forma visual monitorizar la red y filtrar búsquedas. |
| **Criterios de aceptación** | Los dashboards en Kibana permiten visualizar los logs y las alertas generadas por Winlogbeat de forma clara.<br>Se pueden realizar búsquedas filtradas. |
| **Tareas** | Selección de los campos relevantes de los logs generados por Winlogbeat.<br>Creación de gráficos que muestren la información relevante de forma clara y permita filtrar búsquedas.<br>Creación de una vista para visualizar las alertas referentes a Winlogbeat. |
| **Prioridad** | Media |
| **Estimación de esfuerzo** | 8 Horas |
| **Dependencias** | Depende de la Historia 7. |

## SPRINT 7: Instalación de CALDERA y Simulación de Ataques

En este sprint se llevará a cabo la instalación y configuración de CALDERA, que será el software que se utilizará para simular ataques y poner a prueba la infraestructura creada a lo largo del proyecto. Por último, se llevarán a cabo las simulaciones de los ataques con CALDERA.

| **Historia 12: Instalación, configuración y preparación de CALDERA** | |
|---|---|
| **Descripción** | Instalar y configurar CALDERA para simular ataques y realizar la investigación necesaria para determinar qué ataques realizar y cómo configurar su ejecución. |

| | |
|---|---|
| **Criterios de aceptación** | CALDERA está correctamente instalado.<br>Se puede acceder a CALDERA y está correctamente configurado para simular los ataques. |
| **Tareas** | Descargar e instalar CALDERA.<br>Configurar CALDERA.<br>Investigación sobre los tipos de ataques que se van a realizar y cómo realizarlos.<br>Realizar una prueba para asegurar que permite simular ataques. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 12 Horas |
| **Dependencias** | Ninguna. |

| **Historia 13: Simulación de ataques con CALDERA** | |
|---|---|
| **Descripción** | Simular ataques usando CALDERA para poner a prueba la infraestructura implementada. |
| **Criterios de aceptación** | CALDERA simula los ataques de forma correcta.<br>La infraestructura detecta los ataques generados por CALDERA.<br>Las alertas generadas por los ataques de CALDERA se pueden visualizar desde los dashboards creados en Kibana. |
| **Tareas** | Configuración de los ataques en CALDERA.<br>Simular los ataques.<br>Comprobar en Kibana que la infraestructura detecta los ataques y genera alertas. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 10 Horas |
| **Dependencias** | Depende de la historia 11. |

## SPRINT 8: Redacción de la memoria

En este sprint se redactará la memoria del proyecto, un documento que recogerá todo el trabajo realizado. La redacción se llevará a cabo utilizando LaTex.

| **Historia 14: Redacción de la memoria** | |
|---|---|
| **Descripción** | Redactar la memoria del proyecto utilizando Latex. |
| **Criterios de aceptación** | El documento detalla de forma clara todo el proceso de implementación del SIEM, incluyendo la investigación, la puesta en contexto y las pruebas realizadas para verificar que la infraestructura implementada funciona correctamente. |
| **Tareas** | Redacción de la memoria del proyecto utilizando Latex. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 150 Horas |
| **Dependencias** | Depende de la finalización de todos los sprints anteriores. |

## 3.9 Riesgos

En este apartado se describen los principales riesgos que podrían afectar al desarrollo del proyecto. De cada riesgo se analiza la probabilidad de que ocurra y el impacto que tendría en el desarrollo del proyecto. Para cada uno se establece un plan de contingencia que permita mitigar las consecuencias negativas en caso de que el riesgo ocurra.

**Escala de probabilidad:**

- Alta: >50 %
- Media: 25 % - 50 %
- Baja: <25 %

**Impacto:**

- Alto: Repercusión directa en la capacidad de entregar el proyecto a tiempo.
- Medio: Retraso moderado, puede provocar pequeños ajustes en la planificación.
- Bajo: Pequeño retraso, asumible sin modificar el plan preestablecido.

Riesgos identificados para el proyecto:

| **R01** | |
|---|---|
| **Riesgo** | Avería en el equipo en el que se desarrolla el proyecto. |
| **Probabilidad** | Baja |
| **Impacto** | Alto |
| **Plan de contingencia** | Se realizarán copias de seguridad periódicas de las máquinas virtuales. En caso de avería se desplegará la infraestructura en otro equipo. |

| **R02** | |
|---|---|
| **Riesgo** | Falta de espacio en disco en el equipo en el que se desarrolla el proyecto. |
| **Probabilidad** | Media |
| **Impacto** | Alto |
| **Plan de contingencia** | Se monitorizará el espacio libre en disco y se eliminarán archivos innecesarios antes de realizar tareas críticas, ya que trabajar con máquinas virtuales y contenedores requiere una cantidad considerable de almacenamiento. |

| **R03** | |
|---|---|
| **Riesgo** | Rendimiento limitado por los recursos asignados a las VMs. |
| **Probabilidad** | Alta |
| **Impacto** | Medio |
| **Plan de contingencia** | Se monitorizará el uso de CPU y RAM y se distribuirán los recursos entre VMs, también se cerrarán servicios no esenciales. |

| R04 | |
|---|---|
| **Riesgo** | Fallos en la red interna entre las VMs. |
| **Probabilidad** | Media |
| **Impacto** | Medio |
| **Plan de contingencia** | Se revisarán las configuraciones de cada máquina virtual en VirtualBox y se revisará la conectividad entre estas después de cada cambio relevante. |

| R05 | |
|---|---|
| **Riesgo** | Dificultad técnica inesperada. |
| **Probabilidad** | Alta |
| **Impacto** | Alto |
| **Plan de contingencia** | Se dedicará más tiempo a investigar la tecnología utilizada. Si el problema persiste, se plantearán cambios en el alcance para que el proyecto no se demore excesivamente. Si fuera necesario, se pospondrá la entrega a la convocatoria extraordinaria. |

| R06 | |
|---|---|
| **Riesgo** | Falta de tiempo en las fases finales del proyecto. |
| **Probabilidad** | Media |
| **Impacto** | Alto |
| **Plan de contingencia** | Se evaluará cuáles son las tareas menos críticas para descartarlas en caso de que el avance del proyecto no permita completarlas a tiempo. |

| R07 | |
|---|---|
| **Riesgo** | Sobrecarga de tareas por compaginar el TFG con otras responsabilidades. |
| **Probabilidad** | Alta |
| **Impacto** | Alto |
| **Plan de contingencia** | Se planificará el TFG de forma realista. Si el proyecto se retrasa se redistribuirá la carga durante los fines de semana, en caso de ser necesario se pospondrá la entrega a la convocatoria extraordinaria. |

| R08 | |
|---|---|
| **Riesgo** | Enfermedad o indisposición personal. |
| **Probabilidad** | Baja |
| **Impacto** | Medio |
| **Plan de contingencia** | Se reorganizará el calendario para recuperar las horas perdidas. |

## 3.10 Estimación de los costes

En este apartado se presenta una estimación de los costes con el objetivo de establecer un presupuesto aproximado necesario para la realización del proyecto. Se han tenido en cuenta tanto los recursos materiales como la mano de obra durante las 300 horas de duración previstas en la planificación.

### 3.10.1 Presupuesto Hardware

Para el coste total del hardware necesario para desarrollar el proyecto, se ha estimado un coste proporcional en función de la vida útil de cada componente y las 300 horas previstas de trabajo. Esto permite realizar una aproximación realista del valor de los recursos empleados.

| | Precio | Vida útil estimada | Coste total |
|---|---|---|---|
| Ordenador portátil | 700€ | 3000 Horas | 70€ |
| Monitor 1 auxiliar | 200€ | 5000 Horas | 12€ |
| Monitor 2 auxiliar | 200€ | 5000 Horas | 12€ |
| Ratón | 30€ | 2000 Horas | 4,5€ |
| Teclado | 20€ | 2000 Horas | 3€ |

Cuadro 3.26: Estimación de costes de hardware

### 3.10.2 Presupuesto Mano de Obra

El sueldo medio de un ingeniero dedicado a la ciberseguridad en España ronda los 33.375€ al año para aquellos con menos experiencia, como es el caso. Esto se traduce en 16,06€/hora. Teniendo en cuenta que la duración del proyecto es de 300 horas, hace un total de 4818€.

### 3.10.3 Presupuesto Gastos Adicionales

Además del hardware y la mano de obra, se han tenido en cuenta otros gastos asociados al desarrollo del proyecto, como el consumo eléctrico y el acceso a internet. Se han calculado en base a un coste mensual aproximado y la duración del proyecto.

| | Coste al mes | Total |
|---|---|---|
| Coste de electricidad | 5€ | 35€ |
| Internet | 30€ | 210€ |

Cuadro 3.27: Estimación de gastos adicionales

### 3.10.4 Presupuesto Total del Proyecto

La suma total de los tres apartados anteriores da un presupuesto total estimado de 5164,5€. Esto sirve como referencia para tener una idea aproximada del coste que podría suponer desarrollar un proyecto de estas características.

| | Coste |
|---|---|
| Presupuesto Hardware | 101,5€ |
| Mano de obra | 4818€ |
| Gastos adicionales | 245€ |
| **Presupuesto Total** | **5164,5€** |

Cuadro 3.28: Presupuesto total del proyecto

# Capítulo 4

# Marco Conceptual

Este capítulo tiene como objetivo desarrollar los conceptos esenciales para comprender el propósito de este proyecto. En él, se describen los conceptos clave relacionados con la ciberseguridad, así como las herramientas que se utilizarán.

En concreto, se profundizará en la arquitectura SIEM, explicando su funcionamiento y los beneficios de su implementación en el ámbito de la ciberseguridad.

## 4.1 Introducción

La ciberseguridad es el conjunto de prácticas, herramientas y metodologías empleadas para proteger los sistemas informáticos, frente a ataques maliciosos. Esto no solo abarca la protección de la infraestructura tecnológica, sino también la integridad, disponibilidad y confidencialidad de los datos almacenados en estos sistemas. Para ello se implementan “barreras” para evitar que los atacantes se aprovechen de las vulnerabilidades de un sistema.

En un mundo cada vez más digitalizado la protección de los datos y sistemas es cada vez más importante. El papel de la ciberseguridad es fundamental en sectores críticos como la sanidad, las finanzas o la infraestructura de un país, donde un fallo de seguridad puede repercutir en las organizaciones o incluso en la seguridad pública, ya que no solo se encarga de proteger la información, también garantiza la continuidad de los servicios esenciales.

Dentro de este ámbito, hay varios enfoques respecto a la defensa y evaluación de la seguridad de un sistema. Por un lado, tenemos al Blue Team, que se encarga de defender los sistemas de la organización. Esta tarea incluye la monitorización de los sistemas, la implementación de medidas de seguridad, la respuesta a incidentes y la mejora de la infraestructura encargada de proteger a la organización de posibles amenazas. Por otro lado, tenemos al Red Team, cuya tarea principal es identificar y explotar vulnerabilidades para poner a prueba los sistemas de la organización, como podría hacerlo un atacante real. El objetivo de este equipo es detectar brechas de seguridad que el Blue Team haya podido pasar por alto. Para mejorar la colaboración entre ambos equipos, surgió el Purple Team, cuya finalidad es facilitar la comunicación entre el Blue Team y el Red Team de forma que los ataques ejecutados puedan ser más realistas y las defensas más efectivas.

## 4.2 Arquitectura SIEM

Un SIEM es un sistema diseñado para centralizar y gestionar los registros de actividad generados por los distintos dispositivos y servicios de una red. Su principal función es facilitar la detección de amenazas en tiempo real y mejorar la capacidad de respuesta ante incidentes mediante la recopilación, correlación y análisis de dichos registros. Al unificar toda la información en una única plataforma, el SIEM permite monitorizar de forma continua la actividad de la red y detectar comportamientos que puedan indicar una posible intrusión o actividad maliciosa.

Las características más importantes de un SIEM son:

- **Integración de múltiples fuentes de datos:** Un SIEM tiene la capacidad de recopilar información generada por diversas fuentes, como firewalls, endpoints, sistemas de detección de intrusiones, entre otros. Esto permite centralizar todos los eventos detectados en los equipos monitorizados para tener una visión global de la red.
- **Correlación de eventos:** Gracias a la recopilación de eventos de múltiples fuentes, el SIEM puede correlacionar sucesos asociados a posibles amenazas, que de haber sido analizados por separado, podrían haber pasado desapercibidos.
- **Detección en tiempo real:** El tiempo es un factor muy importante de cara a responder ante posibles amenazas. Los SIEM permiten detectar comportamientos sospechosos en tiempo real y permiten actuar de manera proactiva para evitar o minimizar los posibles daños.
- **Automatización de tareas:** La capacidad de responder de forma automática frente a la detección de patrones sospechosos es realmente importante, ya que minimiza el tiempo de respuesta frente a posibles amenazas.
- **Histórico de datos:** Un SIEM también puede funcionar como un histórico que almacene todos los eventos que se han registrado en los equipos monitorizados. Esto es muy útil para realizar una reconstrucción de eventos o redactar un informe en un análisis forense.
- **Escalabilidad y flexibilidad:** Esta arquitectura tiene la capacidad de adaptarse a las necesidades de las organizaciones y gestionar fácilmente variaciones en el volumen de datos manejado.

Gracias a todas estas capacidades, los SIEM se han convertido en una parte fundamental de la defensa de las organizaciones, además de ser un tema realmente interesante para adentrarse en el mundo de la ciberseguridad.

## 4.3 Componentes de un SIEM

Un SIEM está compuesto por una serie de elementos que trabajan de manera conjunta y cada uno con una función específica:

- **Fuentes:** Son los sistemas encargados de generar información sobre lo que ocurre en los equipos monitorizados. Los datos recopilados son registros de eventos generados por estos sistemas, los cuales contienen información sobre las actividades detectadas. Se implementan múltiples fuentes con la finalidad de tener información clara de lo que pasa tanto cada equipo como en la red.
- **Recolector de eventos centralizado:** Una vez generados los eventos por las fuentes, todos estos tienen que ser recolectados y centralizados para poder analizarlos. Este es el objetivo de este componente. La centralización es clave para gestionar grandes volúmenes de datos.
- **Procesador de eventos:** Este componente se encarga de aplicar filtros, normalizar los datos recolectados o añadir información adicional para poder realizar un mejor análisis. Esta parte del proceso es fundamental,

ya que permite estructurar los datos para que sea más fácil interpretarlos, esto ayuda a la hora de crear alertas o simplemente para detectar un comportamiento anómalo leyendo la información de un log.

- **Almacenamiento de eventos:** Los eventos recolectados son almacenados durante un periodo de tiempo establecido en la política ILM. La optimización de este almacenamiento es importante, ya que es aquí donde se realizan las consultas filtradas para las investigaciones.

- **Correlacionador de eventos:** Encargado de encontrar eventos de diversas fuentes que estén asociados al mismo comportamiento y sean un indicativo de actividad maliciosa. Esto se utiliza para detectar amenazas más sofisticadas, que de haber sido analizados los eventos por separado no se habrían catalogado como un comportamiento sospechoso.

- **Alertas:** Una vez que los logs han sido recopilados y almacenados, se definen una serie de reglas que se ejecutan de forma periódica para analizar el comportamiento registrado. Cuando estas reglas detectan que se cumplen ciertas condiciones preestablecidas, se genera un alerta de forma automática. Esto permite a los analistas identificar rápidamente actividades potencialmente maliciosas y responder antes posibles ataques.

- **Interfaz de visualización y análisis:** Proporciona a los analistas de ciberseguridad una herramienta con la que visualizar los eventos y alertas generadas. Desde esta interfaz se pueden realizar búsquedas filtradas de eventos lo cual permite investigar en profundidad cada alerta. Visualizar los eventos con gráficos ayuda a detectar patrones o comportamientos sospechosos.

## 4.4 Fuentes de datos para SIEM

Para que un SIEM pueda detectar comportamientos sospechosos en un sistema, es necesario recopilar información sobre lo que está sucediendo en distintos puntos de este. Esta información proviene de diversas herramientas conocidas como fuentes, que generan logs en tiempo real sobre la actividad del sistema. En este apartado se muestran los tipos de fuentes más comunes que pueden intergrarse en un SIEM:

**Firewall:** Software encargado de controlar el tráfico de red. Su función es bloquear o permitir conexiones en función de reglas predefinidas, como por ejemplo la procedencia o la reputación de la IP de origen.

**IDS (Intrusion Detection System):** Herramienta encargada de monitorizar el tráfico de red y las actividades sospechosas que indiquen intentos de intrusión. En el caso de detectar un comportamiento anómalo, genera alertas con la información recopilada.

**IPS (Intrusion Prevention System):** Herramienta similar al IDS, con la diferencia de que un IPS, además de detectar intrusiones tiene la capacidad de tomar acciones para prevenirlas, como bloquear el tráfico categorizado como malicioso en tiempo real.

**EDR (Endpoint Detection and Response):** Herramienta que monitoriza la actividad en tiempo real de cada equipo. Además de detectar comportamientos sospechosos, permite responder aislando el equipo o bloqueando comportamientos inusuales.

**Herramientas de recolección de eventos:** Estas herramientas recopilan los eventos generados por los sistemas operativos y aplicaciones. Su objetivo es proporcionar visibilidad sobre las actividades del sistema, detectar comportamientos sospechosos y aumentar la información para poder relacionar eventos.

## 4.5 Comportamientos sospechosos

En el ámbito de la ciberseguridad existen numerosos tipos de ataques. Cada uno de ellos es distinto, pero la mayoría dejan evidencias y siguen patrones que con las herramientas adecuadas, pueden ser identificados.

La implementación de un SIEM permite recopilar y analizar los eventos generados por los equipos monitorizados, con el objetivo de detectar en tiempo real comportamientos que puedan estar relacionados con un intento de ataque o un ataque ya materializado.

A continuación, se describen una serie de comportamientos sospechosos que pueden ser indicativos de un intento de ataque o de un sistema comprometido:

**Ataque de denegación de servicio distribuido (DDoS):** Este tipo de ataque busca saturar un sistema con el objetivo de hacer que los servicios que ofrece no estén disponibles. El origen de las solicitudes que saturan el sistema proviene de múltiples dispositivos, lo que hace más complicado detenerlo.

Este ataque se puede identificar si se observa un aumento significativo del volumen de peticiones hacia un mismo destino.

**Ataque de fuerza bruta:** Consiste en intentar adivinar la contraseña de un usuario probando todas las combinaciones posibles hasta encontrar la correcta. Estos ataques son lentos pero el uso de herramientas que permiten automatizarlos hace que sean peligrosos, sobre todo frente a contraseñas débiles.

Este ataque se puede detectar monitorizando el volumen de eventos de autenticación fallida. En el momento que se ve un alto número de eventos de autenticación fallida desde la misma IP o para el mismo usuario puede ser un indicativo de un ataque de este tipo.

**Ransomware:** Ataque que consiste en cifrar los archivos de un sistema infectado, impidiendo al usuario acceder a ellos, y exigiendo un rescate a cambio de la clave con la que se descifran los archivos.

Este ataque se puede detectar si se observa la ejecución de procesos sospechosos y una gran cantidad de archivos modificados en un mismo equipo en un corto periodo de tiempo.

**Command and Control (C2C):** Canal de comunicación que establece un atacante entre un sistema comprometido y un servidor controlado por él. A través de este canal el atacante puede filtrar información del sistema infectado o enviar nuevas órdenes para continuar con la intrusión.

Para detectar este comportamiento es necesario monitorizar las conexiones de los equipos de nuestra red hacia el exterior, para verificar si el destino de las peticiones es legítimo.

**Escaneo de puertos:** Es una técnica utilizada en las fases iniciales de un ataque, que permite identificar los servicios que están expuestos en un sistema. Una vez detectados los puertos abiertos y los servicios asociados, el atacante puede intentar explotar vulnerabilidades existentes en los servicios expuestos. Identificar este comportamiento a tiempo puede resultar clave para frenar futuros ataques.

Este comportamiento puede detectarse si se observan numerosas conexiones desde la misma IP hacia distintos puertos de un mismo equipo en un corto periodo de tiempo.

**Movimiento lateral:** Una vez comprometido un sistema, el atacante se desplaza hacia otros equipos de la red para obtener acceso a más privilegios y por lo tanto a recursos más sensibles.

Este comportamiento se puede identificar detectando conexiones inusuales entre equipos o autenticaciones de un mismo usuario en distintos equipos.

**Accesos fuera de horarios:** Detectar un acceso fuera del horario laboral puede ser indicativo de movimiento lateral o un comportamiento malicioso.

Estos son algunos de los ataques o comportamientos sospechosos más habituales. Gracias a la implementación de un SIEM, se pueden crear alertas personalizadas con las que monitorizar cualquier tipo de evento o patrón.

## 4.6 Evolución histórica de los SIEM

En los inicios de la informática, los sistemas de seguridad no eran tan avanzados como hoy en día. La protección se basaba principalmente en la restricción de accesos y autenticaciones básicas mediante contraseñas simples.

En los años 90, especialmente debido a la llegada de internet, la protección de los sistemas y datos empezó a cobrar importancia. Fue en este momento cuando surgieron los primeros IDS e IPS, creados para identificar tráfico sospechoso y bloquear posibles ataques. Sin embargo, estas herramientas trabajaban de forma independiente y no tenían la capacidad de correlacionar eventos procedentes de diversas fuentes.

Debido al crecimiento y sofisticación de los ataques cibernéticos, los equipos encargados de gestionar la seguridad se enfrentaban al reto de centralizar y analizar de forma eficiente toda la información de las diferentes fuentes. Fue en la década de los 2000, donde se vieron los primeros SIEM, que combinaban 2 tecnologías ya existentes:

- **SIM (Security Information Management):** Encargado de gestionar y almacenar la información relacionada con la seguridad de los equipos monitorizados.
- **SEM (Security Event Management):** Encargado de monitorear en tiempo real los eventos de seguridad y detectar comportamientos inusuales.

La fusión de estas dos tecnologías permitió tener una visión global y centralizada de los equipos que se quieren proteger, esto facilitó la detección temprana de posibles ataques. Con el tiempo, las herramientas encargadas de monitorizar los sistemas han ido evolucionando, implementando mejoras como la inteligencia artificial o automatización que ha hecho que estos sistemas sean esenciales en el ámbito de la ciberseguridad.

# Capítulo 5

# Soluciones Existentes

Antes de desplegar un SIEM es esencial conocer las distintas alternativas tecnológicas disponibles. La mayoría de los SIEMs se componen de varios módulos funcionales, entre los que se incluye la ingesta de logs, el procesamiento de eventos en tiempo real, el almacenamiento de datos y la visualización de información.

Cada uno de estos componentes puede ser implementado mediante diferentes herramientas, ya sean OSS (Open Source Software) o comerciales, con distintas características en cuanto a rendimiento, facilidad de uso o adecuación con los requisitos del proyecto.

En este capítulo se describirán las herramientas más utilizadas en la actualidad para cada una de estas funciones, con la finalidad de proporcionar una visión general sobre las diferentes opciones en el diseño de una arquitectura SIEM.

Además, se incluyen distintas herramientas IDS y opciones de recolección de eventos en Windows, que son las fuentes implementadas en este trabajo, responsables de generar los logs y alimentar al SIEM con información.

## 5.1 Ingesta de logs

**Logstash**, OSS: Es una herramienta que permite ingestar y procesar datos en tiempo real, forma parte de la pila ELK. Su función es recolectar, transformar y normalizar logs de diferentes fuentes y enviarlos a sistemas de almacenamiento. Su compatibilidad con numerosos formatos de datos hace que sea una de las mejores opciones en sistemas complejos. Entre sus limitaciones está el alto consumo de recursos, sobre todo en proyectos con un alto volumen de datos.

**Fluentd**, OSS: Es una herramienta de recolección de logs diseñada para unificar la ingesta y el consumo de logs en formato JSON. Destaca por su flexibilidad, lo que permite adaptarla a diferentes entornos. Además, puede operar con muy baja carga de memoria, lo que lo hace adecuado para entornos con recursos limitados. Esta herramienta requiere una gestión de la configuración propia, no cuenta con soporte comercial más allá de la comunidad.

**Splunk Universal Forwarder**, Comercial: Herramienta muy ligera especialmente diseñada para recolectar logs en equipos cliente y enviarlos a un indexador de Splunk Enterprise. Está optimizado para integrarse de forma eficiente con la plataforma Splunk, garantizando una transmisión fiable y segura de los datos. Entre sus desventajas está que esta herramienta depende del ecosistema Splunk, además esta herramienta no realiza ninguna

transformación a los logs, los envía en bruto.

## 5.2 Procesamiento de eventos

**Apache Kafka**, OSS: Plataforma distribuida de mensajería y procesamiento de datos en tiempo real. Permite la transmisión y almacenamiento de grandes cantidades de datos de manera eficiente y escalable. En Kafka se pueden organizar los mensajes en "topics" lo que permite que los productores publiquen eventos y los consumidores los procesen en tiempo real. Sus principales ventajas son la escalabilidad horizontal y el bajo tiempo de latencia.

**RabbitMQ**, OSS: Sistema de mensajería basado en el protocolo AMQP (Advanced Message Queuing Protocol). Actúa como message broker, recibe mensajes de los productores y los distribuye a los consumidores de manera asíncrona. Destaca por su facilidad de configuración pero puede presentar limitaciones en entornos con cargas de eventos elevadas en comparación a otras herramientas como Kafka.

**Azure Event Hubs**, Comercial: Servicio creado para recibir eventos en tiempo real ofrecido por Microsoft Azure. Permite recolectar y procesar eventos de múltiples fuentes. Su integración con otros productos de Azure facilita la construcción de arquitecturas basadas en la nube. Tiene la capacidad de ingestar logs de forma masiva y es fácil de implementar, pero su uso está ligado al ecosistema Azure, lo cual requiere una suscripción de pago.

## 5.3 Almacenamiento de logs

**Elasticsearch**, OSS/Comercial: Es un motor de búsqueda y análisis que permite procesar y almacenar logs en tiempo real. En un SIEM se emplea como repositorio central de logs. Su capacidad de indexación permite realizar búsquedas eficientes, lo cual ayuda a detectar patrones sospechosos. Su arquitectura distribuida permite escalar horizontalmente de forma sencilla, aunque su elevado consumo de memoria y almacenamiento puede afectar al rendimiento si no se configura adecuadamente.

**Splunk Enterprise**, Comercial: Esta plataforma indexa, almacena y permite realizar búsquedas en tiempo real. Actúa como un almacén central de logs y su principal ventaja es su potente motor de indexación y su búsqueda avanzada que permite correlacionar eventos, detectar patrones sospechosos y generar alertas en tiempo real. Actualmente es muy utilizado ya que permite gestionar grandes volúmenes de datos y ofrece una gran personalización en la creación de dashboards. Sin embargo es muy costoso y requiere una licencia por volumen de datos.

**MongoDB**, OSS/Comercial: Se trata de una base de datos NoSQL orientada a documentos JSON que ofrece una escalabilidad horizontal integrada. Es útil para almacenar grandes volúmenes de datos como registros de eventos. Su modelo basado en documentos facilita la inserción y consulta de datos. Sin embargo no está optimizada para consultas complejas, donde suele ser preferible recurrir a bases de datos tradicionales.

## 5.4 Visualización

**Kibana**, OSS/Comercial: Es una herramienta de visualización de logs que forma parte de la pila ELK. Permite visualizar e interactuar con los logs almacenados en Elasticsearch desde una interfaz web. Kibana facilita la creación de dashboards interactivos con gráficos y tablas, representando de forma visual la información de los logs y proporcionando a los analistas de seguridad una visión general de lo que está sucediendo en los equipos monitorizados. Su funcionamiento depende completamente de Elasticsearch, y en entornos con altos volúmenes de datos puede generar problemas si no se optimiza adecuadamente.

**Grafana**, OSS/Comercial: Es una plataforma de visualización y análisis de logs que permite crear dashboards interactivos para monitorear logs en tiempo real. A diferencia de Kibana puede conectarse a múltiples fuentes de datos como Elasticsearch, Prometheus o Grafana Loki, lo que la hace útil en entornos donde existen múltiples orígenes de datos. Sin embargo, esta herramienta está más enfocada al análisis de métricas que a la visualización de logs, por lo que para la visualización detallada de registros es necesario combinarla con otras herramientas.

**Splunk Dashboards**, Comercial: Splunk Enterprise a través de su lenguaje de búsqueda permite generar gráficos, tablas e informes interactivos. Destaca por su flexibilidad ya que cada consulta se puede representar y se integra perfectamente con el resto de herramientas del ecosistema Splunk. La desventaja de esta herramienta es su alto coste y su dependencia del resto de herramientas Splunk.

## 5.5 IDS opciones

**Snort**, OSS: Sistema de detección de intrusiones que permite analizar el tráfico de red en tiempo real utilizando un conjunto de reglas predefinidas. Es muy utilizado debido a su fiabilidad y una de sus principales características es su alta capacidad de personalización mediante reglas escritas de forma manual, lo que permite adaptarlo a las necesidades de cada sistema.

**Suricata**, OSS: Sistema de detección y prevención de intrusiones que analiza el tráfico de red en tiempo real. Incluye la inspección profunda de paquetes y detección de anomalías. Esta herramienta destaca por su gran capacidad para analizar un gran volumen de tráfico sin afectar al rendimiento, gracias a su arquitectura paralela.

**Zeek**, OSS: Herramienta de monitoreo de red enfocada en la detección de intrusiones a través del análisis del tráfico de red. Es especialmente útil para generar registros detallados del tráfico de red. Además, ofrece un enfoque basado en eventos y scripting, lo que permite personalizar la detección de comportamientos anómalos y adaptar su funcionamiento a las necesidades específicas de cada entorno.

### Herramientas de recolección de eventos de Windows

**Winlogbeat**, OSS: Herramienta diseñada para recolectar y enviar eventos del registro de Windows a un SIEM desarrollado por Elastic. Permite capturar eventos relacionados con autenticación, acceso a archivos y actividades de procesos. Su principal ventaja es su fácil configuración y su bajo consumo de recursos.

**NXLog**, OSS/Comercial: Herramienta de recolección de eventos en sistemas Windows, Linux y Unix. Creada para recolectar y enviar los eventos a SIEMs o bases de datos. Es una herramienta compatible con distintos protocolos y formatos, pero su configuración es más compleja que herramientas específicas como Winlogbeat.

**Wazuh Agent**, OSS: Agente de monitoreo que permite recolectar eventos del sistema operativo y monitorizar la integridad de archivos, configuraciones y actividad sospechosa. Su principal ventaja es que, además de recolectar datos, realiza un análisis local mediante reglas predefinidas, lo que le permite generar alertas en tiempo real ante comportamientos anómalos. Sin embargo, su despliegue requiere la integración con un servidor Wazuh, lo que aumenta la complejidad de la infraestructura.

# Capítulo 6

# Plataforma Tecnológica

En este capítulo se describe la arquitectura conceptual y lógica de la plataforma tecnológica diseñada para este TFG, cuyo objetivo es la experimentación con una infraestructura tipo SIEM. Se detalla tanto el diseño general del sistema como las herramientas empleadas para su desarrollo, implementación y despliegue:

## 6.1 Diseño de plataforma

El diseño de la plataforma se basa en una arquitectura modular y escalable que permite integrar múltiples fuentes de datos en un sistema centralizado de procesamiento y visualización de logs.

La infraestructura del SIEM está compuesta por Kafka, Logstash, Elasticsearch y Kibana. Cada componente se encarga respectivamente de recibir, procesar, almacenar y visualizar los datos. Estos datos provienen de Winlogbeat y Zeek, que actúan como fuentes de datos generando los logs necesarios para monitorizar la actividad del entorno de pruebas.

Por un lado, se ha implementado Winlogbeat para la recolección de eventos en entornos Windows, lo cual permite obtener información detallada de eventos clave de sistema operativo, relacionados con la actividad del usuario y el propio sistema. Por otro lado, se ha integrado el IDS Zeek para monitorizar el tráfico de red del entorno de pruebas, lo cual permite recopilar datos en tiempo real sobre las comunicaciones dentro de la red.

Ambas herramientas envían sus logs a Kafka, que actúa como intermediario clasificando la información en distintos topics en función de la fuente origen. Esto permite organizar los datos, lo que facilita su tratamiento posterior.

Los logs recibidos por Kafka son consumidos por Logstash, encargado de procesar los datos para, posteriormente, almacenarlos en Elasticsearch, el cual actúa como base de datos del sistema, donde se guarda toda la información generada por las fuentes. Por último, Kibana se utiliza como herramienta de visualización y análisis. Desde su interfaz se visualizan los datos almacenados en Elasticsearch, y también permite la creación de reglas que generan alertas en tiempo real al detectar comportamientos sospechosos.

El despliegue de los componentes que forman el SIEM se ha realizado mediante contenedores Docker, lo que permite crear un entorno aislado, fácilmente gestionable, y escalable. Esto facilita tanto el mantenimiento de la infraestructura como la ampliación en caso de necesidad.

La plataforma construida permite la recolección, procesamiento y visualización de eventos de forma eficiente, modular y adaptable, lo cual la convierte en una en una solución útil para la monitorización y detección de amenazas en tiempo real.

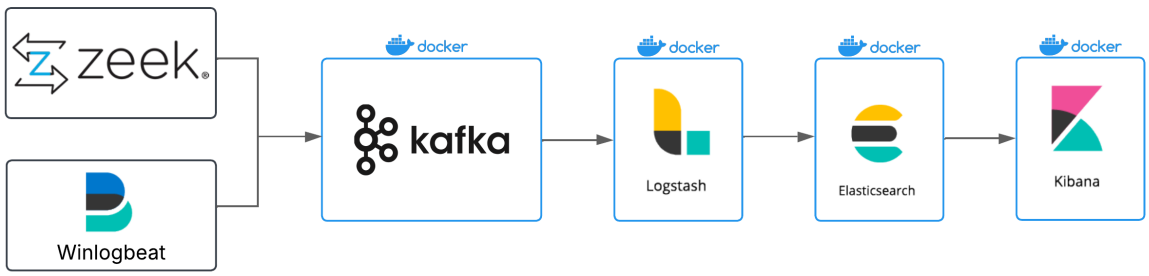


Figura 6.1: Diagrama de la plataforma: Kafka, ELK y fuentes de logs

## 6.2 Herramientas

### 6.2.1 VirtualBox

Herramienta de virtualización de código abierto desarrollada por Oracle. Permite ejecutar múltiples sistemas operativos como máquinas virtuales sobre un equipo físico. Esto hace posible la creación de entornos aislados y configurables, para realizar pruebas o simulaciones sin necesidad de modificar el sistema operativo de tu equipo principal. Su facilidad de uso y su amplia compatibilidad con múltiples plataformas la hacen una herramienta ampliamente utilizada en entornos académicos y profesionales.

En este proyecto se utilizará VirtualBox para crear un entorno de pruebas controlado que simule una infraestructura realista. En este entorno se desplegarán diversas máquinas virtuales que permitirán implementar y poner a prueba los componentes del SIEM.

### 6.2.2 Kafka

Kafka es una plataforma que sirve para recoger, procesar y enviar datos en tiempo real de manera rápida y fiable. Funciona con un sistema Publish-Subscribe, en el que los productores envían los datos a “topics” específicos y los consumidores se suscriben a estos “topics” para recibir los mensajes. Esta herramienta es útil para recopilar, procesar y almacenar grandes volúmenes de logs procedentes de diversas fuentes.

El sistema está formado por uno o más brokers. Cada broker es un servidor que almacena y distribuye los logs producidos por las fuentes. Cada topic en Kafka puede dividirse en varias particiones, lo que permite paralelizar el procesamiento de datos para mejorar la eficiencia del sistema. A su vez, cada partición puede replicarse en distintos brokers estableciendo un factor de replicación, lo que garantiza la tolerancia a fallos, ya que existirán tantas copias de los datos como se haya establecido en el factor de replicación. Esto hace que Kafka permita establecer una infraestructura robusta ante fallos o pérdidas de datos.

El motivo por el que decidí usar esta herramienta es porque permite manejar datos procedentes de múltiples fuentes de forma organizada y sencilla, lo cual es fundamental para desarrollar un SIEM. Además, Kafka se integra perfectamente con la pila ELK, formando una estructura completa para recopilar, procesar, almacenar y visualizar datos.

### 6.2.3 ELK

La pila ELK es un conjunto de herramientas de código abierto que sirven para la recopilación, procesamiento, almacenamiento y visualización de logs en tiempo real. Está compuesta por tres herramientas principales:

#### Logstash

Herramienta de ingesta y procesamiento de datos que permite recopilar, transformar y normalizar logs procedentes de diversas fuentes antes de enviarlos a Elasticsearch. Logstash recibe datos de Kafka suscribiéndose a los topics correspondientes de cada fuente y los procesa mediante pipelines personalizados. Este procesamiento permite eliminar información redundante, estructurar los datos para facilitar el análisis o enriquecer los logs con más datos.

#### Elasticsearch

Motor de búsqueda diseñado para indexar, almacenar y analizar grandes cantidades de datos en tiempo real. Su arquitectura permite realizar consultas rápidas y eficientes sobre grandes volúmenes de datos.

La unidad básica de almacenamiento en Elasticsearch es el índice, que agrupa logs con una estructura similar. Cada vez que Logstash envía un evento, este se almacena dentro de un índice determinado. Los índices permiten organizar la información cronológicamente y por fuente de origen, lo que facilita el análisis posterior.

Una práctica habitual es generar índices de forma diaria, de forma que la información recopilada cada día se almacena en un índice distinto. Estos índices suelen seguir una estructura de nombre basada en la fuente de datos y fecha, por ejemplo winlogbeat.YYYY.MM.dd.

Elasticsearch funciona mediante una arquitectura distribuida basada en nodos, donde cada uno de ellos es un servidor individual que forma parte de un clúster. Cada nodo almacena parte de los datos. Por defecto, Elasticsearch está configurado con un factor de replicación de 1, es decir, que cada fragmento de datos tiene una copia en otro nodo. Este factor de replicación también se puede configurar manualmente. La replicación de los datos no solo mejora la tolerancia a fallos, sino que ayuda a mejorar la eficiencia en las consultas, ya que se puede distribuir la carga de trabajo entre los nodos que contienen los mismos datos.

En entornos donde se gestionan un gran volumen de datos, almacenar logs de forma ilimitada puede suponer un problema de rendimiento y un elevado consumo de recursos. Para evitarlo, Elasticsearch ofrece un sistema denominado Index Lifecycle Management (ILM), que permite aplicar políticas automáticas de gestión sobre los índices en función de su antigüedad o tamaño.

Una política ILM define las fases que determinan el estado de los índices a lo largo de su ciclo de vida:

- **Hot**: El índice está activo y recibe escrituras.
- **Warm**: El índice ya no recibe datos, pero se consulta con frecuencia.
- **Cold**: El índice apenas recibe consultas, pero los datos se conservan por motivos legales o históricos.
- **Delete**: El índice se elimina automáticamente una vez alcanzada la edad definida por la política.

Las condiciones que determinan cuando un índice debe pasar de una fase a otra se especifican en los archivos de configuración ILM. Esto permite optimizar el uso de los recursos y una gestión automática de los datos.

**Kibana**

Interfaz que permite visualizar e interactuar con los datos almacenados en Elasticsearch. Esta herramienta permite crear *dashboards* personalizados con tablas y gráficos para representar los datos de una forma más visual, obteniendo así una idea más clara de lo que sucede en los equipos monitorizados. Además permite realizar consultas y crear filtros personalizados para identificar comportamientos sospechosos rápidamente y poder tomar decisiones de forma temprana. También permite crear reglas, que al cumplirse una serie de condiciones definidas, generan una alerta de forma automática, lo cual permite detectar comportamientos sospechosos en tiempo real.

El motivo por el que decidí usar estas herramientas es que proporcionan todas las funciones esenciales para implementar un SIEM. Además tienen una gran compatibilidad con otras herramientas, son de código abierto y permiten una gran personalización.

### 6.2.4 Docker

Docker es una plataforma de código abierto que permite la creación, despliegue y ejecución de aplicaciones dentro de contenedores ligeros y portátiles. Estos contenedores contienen todas las dependencias, bibliotecas y configuraciones necesarias para que la aplicación que se encuentra en su interior se ejecute correctamente en cualquier entorno.

El uso de Docker simplifica la implementación del SIEM, ya que Kafka y cada componente de la pila ELK se desplegarán en contenedores separados. Esto permite aislar cada componente del sistema, lo que mejora la gestión al evitar conflictos entre los diferentes servicios. Además, Docker facilita la administración ya que permite gestionar toda la infraestructura desde un único archivo de configuración Docker-Compose.

### 6.2.5 Zeek

Sistema de detección de intrusiones, se encarga de monitorizar el tráfico y genera logs de eventos de red. A diferencia de otros IDS basados en firmas, Zeek utiliza un enfoque basado en eventos y scripting, lo que permite personalizar la detección de comportamientos sospechosos y adaptar su funcionamiento a distintos entornos.

Decidí implementar el IDS Zeek como una de las fuentes de datos para el SIEM, porque genera registros de tráfico de red detallados, lo cual es fundamental para realizar un buen análisis, además es open source y sencillo de implementar.

### 6.2.6 Winlogbeat

Herramienta desarrollada por Elastic que captura y transmite logs de eventos de Windows. Permite recopilar registros referentes a eventos de seguridad, logs de aplicaciones y otros eventos generados por sistemas Windows para poder analizarlos en busca de actividades sospechosas.

Opté por usar Winlogbeat porque es una herramienta fácil de usar que ofrece información muy valiosa para detectar comportamientos sospechosos y monitorear un entorno Windows en tiempo real. Además, al estar desarrollada por Elastic, su integración es sencilla y eficiente.

### 6.2.7 Visual Studio Code

Editor de código fuente gratuito y multiplataforma desarrollado por Microsoft. Diseñado para ofrecer un entorno de desarrollo flexible y eficiente, con soporte para numerosos lenguajes de programación. Una de sus características más destacadas es la integración de un terminal incorporado, que permite ejecutar comandos directamente desde el editor, facilitando tareas como el lanzamiento de scripts o la visualización de logs en tiempo real.

En este proyecto se ha optado por utilizar Visual Studio Code como herramienta principal para la edición y gestión de archivos de configuración, así como para el desarrollo de los scripts relacionados con la infraestructura. Su compatibilidad con Docker, permite administrar contenedores, construir imágenes y monitorizar el estado de los servicios desde el propio entorno de trabajo. Además, su terminal integrado resulta muy útil para ejecutar comandos y visualizar los logs generados por los distintos componentes del sistema, lo que facilita la depuración del código.

### 6.2.8 Overleaf

Plataforma en línea utilizada para la edición de documentos en LaTeX. Está especialmente diseñada para la creación de documentos científicos y técnicos. Permite escribir, compilar y visualizar documentos en tiempo real desde cualquier navegador, sin instalar software adicional. Entre sus funcionalidades destacan el control de versiones, el trabajo colaborativo, el uso de plantillas y la gestión de la bibliografía.

En este proyecto se utilizará Overleaf para redactar la memoria del TFG, gracias a las facilidades que ofrece LaTeX para crear un documento claro y profesional.

### 6.2.9 ChatGPT

Herramienta basada en inteligencia artificial desarrollada por OpenAI. Es un modelo de lenguaje avanzado entrenado para generar texto coherente, responder preguntas y asistir en tareas de programación. Su capacidad para entender el lenguaje natural y generar contenido lo convierte en un recurso muy útil, especialmente para tareas de redacción o desarrollo de software.

En este proyecto se utilizará ChatGPT como herramienta de apoyo durante las distintas fases del proyecto. Se usará para generar scripts, así como para la resolución de dudas técnicas que puedan surgir en el proceso de implementación de la infraestructura. Además, se empleará para facilitar la redacción de la memoria en LaTeX, contribuyendo a agilizar el proceso de documentación del proyecto.

### 6.2.10 Lucid.app

Herramienta en línea especializada en la creación de diagramas, mapas conceptuales o esquemas técnicos. Su interfaz visual, intuitiva y fácil de usar, permite diseñar diagramas de forma clara y rápida. Dispone de una amplia biblioteca de iconos, figuras y elementos predefinidos, y además ofrece la posibilidad de importar imágenes personalizadas. También permite exportar los diagramas en diferentes formatos.

Se ha decidido usar esta herramienta debido a la facilidad que ofrece para crear diagramas detallados, aspecto clave a la hora de documentar la arquitectura del SIEM. Esta herramienta contribuirá a representar visualmente la infraestructura desarrollada durante el proyecto, ayudando a explicar de forma clara el diseño y funcionamiento del sistema implementado.

# Capítulo 7

# Implementación

En este apartado se describe la creación y configuración de las máquinas virtuales que forman el entorno de pruebas. Todas ellas están alojadas en mi equipo personal y comparten una única interfaz de red local, sin acceso a internet.

Para la instalación de las herramientas utilizadas, se ha habilitado una interfaz de red temporal configurada como adaptador puente, que permite a las máquinas virtuales conectarse a internet a través de la conexión de mi equipo. Una vez finalizada la instalación y configuración se ha deshabilitado esta interfaz. Además, en las máquinas virtuales con SO Windows se ha creado una carpeta compartida con mi equipo, lo que permite transferir archivos de manera rápida y sencilla a las VMs.

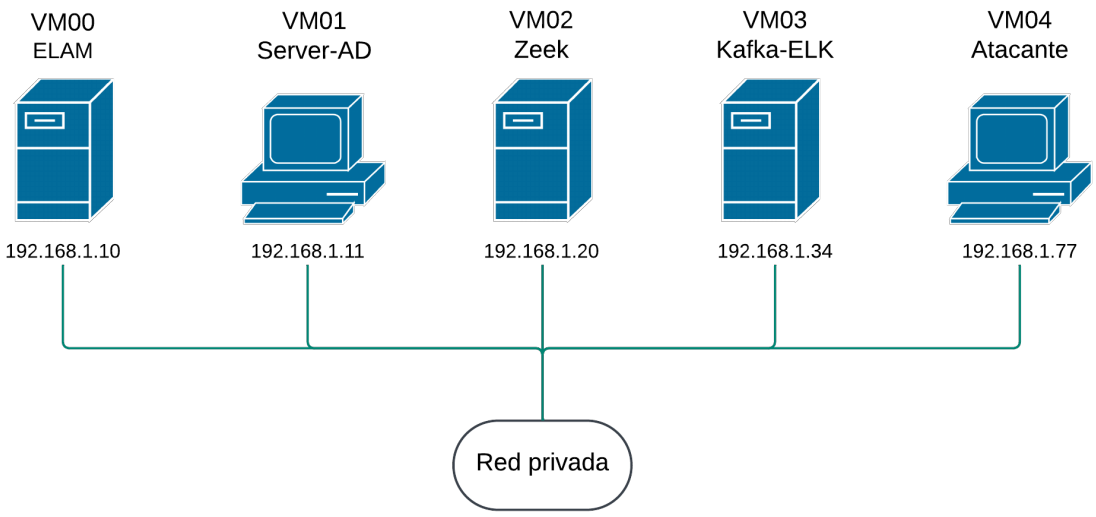


Figura 7.1: Diagrama de red de la infraestructura

## 7.1 VM00 - Servidor Active Directory

Esta máquina cumple la función de **controlador de dominio** y se encarga de gestionar las cuentas de usuario, los equipos y las políticas de seguridad en el entorno Windows. Proporciona los servicios de Active Directory y DNS.

Ha sido creada con el objetivo de replicar un escenario lo más parecido posible a la realidad. En la mayoría de empresas, se utilizan los controladores de dominio para centralizar la autenticación, y la aplicación de políticas de grupo para gestionar usuarios y equipos. Esto permite simular los ataques en un entorno similar a la realidad.

**Características:**

| Nombre | Función | CPU | RAM | Disco | SO |
|---|---|---|---|---|---|
| VM00 | Servidor AD | 2 Núcleos | 1 GiB | 20GB | Windows Server 2022 |

Cuadro 7.1: Características de la VM00 - Servidor Active Directory

**Creación y configuración de la VM**

```
# Creación de la VM
$ VBoxManage createvm --name "Windows_Server_AD" --ostype "Windows2022_64" --register

# Configuración de recursos de la VM
$ VBoxManage modifyvm "Windows_Server_AD" --memory 1024 --cpus 2 --nic1 intnet --intnet1
    "LabNetwork"

# Creación del disco duro
$ VBoxManage createhd --filename "/home/dario/Escritorio/TFGDario/VMs/Windows_Server_AD/
    Windows_Server_AD.vdi" --size 20000

# Creación del controlador de almacenamiento
$ VBoxManage storagectl "Windows_Server_AD" --name "SATA Controller" --add sata --
    controller IntelAhci

# Montaje del disco duro
$ VBoxManage storageattach "Windows_Server_AD" --storagectl "SATA Controller" --port 0 --
    device 0 --type hdd --medium "/home/dario/Escritorio/TFGDario/VMs/Windows_Server_AD/
    Windows_Server_AD.vdi"

# Montaje de la ISO de instalación
$ VBoxManage storageattach "Windows_Server_AD" --storagectl "SATA Controller" --port 1 --
    device 0 --type dvddrive --medium "/home/dario/Escritorio/TFGDario/VMs/ISOs/
    SERVER_EVAL_x64FRE_es-es.iso"

# Montaje de VBoxGuestAdditions
$ VBoxManage storageattach "Windows_Server_AD" --storagectl "SATA Controller" --port 2 --
    device 0 --type dvddrive --medium "/usr/share/virtualbox/VBoxGuestAdditions.iso"

# Configuración del arranque desde DVD
$ VBoxManage modifyvm "Windows_Server_AD" --boot1 dvd
```

Listado 7.1: Creación y configuración de la VM00

**Configuración de red**

Para establecer la **configuración de red** introducimos los siguientes comandos, con los que asignamos la IP estática **192.168.1.10** a la interfaz de red **Ethernet** y establecemos el servidor DNS:

```
New-NetIPAddress -InterfaceAlias "Ethernet" -IPAddress 192.168.1.10 -PrefixLength 24 -
    DefaultGateway 192.168.1.1
Set-DnsClientServerAddress -InterfaceAlias "Ethernet" -ServerAddresses 192.168.1.10
```

Listado 7.2: Configuración de red de la VM00

La configuración de red final de la VM00 es:

```
$ ipconfig

Configuración IP de Windows

Adaptador de Ethernet Ethernet:
   Sufijo DNS específico para la conexión. . :
   Vínculo: dirección IPv6 local. . . : fe80::ddae:10ac:e94c:5711%5
   Dirección IPv4. . . . . . . . . . . . : 192.168.1.10
   Máscara de subred . . . . . . . . . . : 255.255.255.0
   Puerta de enlace predeterminada . . . . . : 192.168.1.1
```

**Creación y configuración del AD**

Asignamos el nombre **Server-AD** al equipo

```
Rename-Computer -NewName "Server-AD"
```

Con el siguiente comando instalamos y configuramos un nuevo bosque de Active Directory, en el que creamos el dominio **PERSIA.UVA.ES**, convertimos el equipo en controlador de dominio y también instalamos el servicio DNS.

```
Install-ADDSForest -DomainName "PERSIA.UVA.ES" -InstallDNS
```

Creamos el usuario **usuarioVictima** miembro del dominio **PERSIA.UVA.ES** y el cual se va a utilizar para iniciar sesión desde la VM01 en el dominio.

```
New-ADUser -Name "usuarioVictima" -GivenName "Usuario" -Surname "Victima" -
    UserPrincipalName "usuarioVictima@PERSIA.UVA.ES" -SamAccountName "usuarioVictima" -
    AccountPassword (ConvertTo-SecureString "Calendario1" -AsPlainText -Force) -Enabled \
    $true
```

**Instalación y configuración de Winlogbeat**

Mediante la carpeta compartida transferimos el archivo **winlogbeat-8.17.0-windows-x86_64.zip** desde mi equipo a la VM, al descomprimirlo ejecutamos el instalador **winlogbeat.exe**.

En el archivo de configuración de **winlogbeat.yml** se especifica el tipo de eventos que recopilar. En este caso se están recolectando los eventos relacionados con **Application**, **System**, **Security**, **PowerShell** y **Sysmon**. Los eventos recopilados se envían a Kafka **192.168.1.34:9092** mediante el topic **winlogbeat-logs**.

Además, se ha añadido un campo personalizado llamado **origin**, que en este caso tiene el valor **Server-AD** para identificar el origen de los logs. Aparte de ser enviados a Kafka, los logs también se almacenan de forma local en **C:/ProgramData/winlogbeat/logs**.

**Código winlogbeat.yml**:

```
###################### Winlogbeat Configuration ########################

winlogbeat.event_logs:
  - name: Application
    ignore_older: 72h
  - name: System
  - name: Security
  - name: Microsoft-Windows-Sysmon/Operational
  - name: Windows PowerShell
    event_id: 400, 403, 600, 800
  - name: Microsoft-Windows-PowerShell/Operational
    event_id: 4103, 4104, 4105, 4106

output.kafka:
  hosts: ["192.168.1.34:9092"]
  topic: "winlogbeat-logs"

setup.kibana:
  host: "http://192.168.1.34:5601"

processors:
  - add_cloud_metadata:
      enabled: false
  - add_fields:
      target: ''
      fields:
        origin: "Server-AD"

logging.level: info
logging.to_files: true
logging.files:
  path: C:/ProgramData/winlogbeat/logs
  name: winlogbeat.log
  keepfiles: 7
  permissions: 0644
  rotateeverybytes: 10485760
```

Listado 7.3: Archivo winlogbeat.yml de configuración de Winlogbeat de la VM00

## 7.2 VM01 - Miembro del dominio

Representa un **equipo de usuario** del dominio **PERSIA.UVA.ES** gestionado por VM00. Simula la actividad habitual de un empleado en una empresa, realizando inicios de sesión, peticiones web o ejecución de archivos.

Este comportamiento genera logs que pueden ser importantes desde el punto de vista de la monitorización, permitiendo detectar comportamientos sospechosos o no habituales.

**Características**

| Nombre | Función | CPU | RAM | Disco | SO |
|---|---|---|---|---|---|
| VM01 | Miembro AD | 2 Núcleos | 1 GiB | 30GB | Windows 10 |

Cuadro 7.2: Características de la VM01 - Equipo miembro del dominio PRESIA.UVA.ES

**Creación y configuración de la VM**

```
# Creación de la VM
$ VBoxManage createvm --name "Windows_10_Cliente1" --ostype "Windows10_64" --register

# Configuración de recursos de la VM
$ VBoxManage modifyvm "Windows_10_Cliente1" --memory 1024 --cpus 2 --nic1 intnet --
    intnet1 "LabNetwork"

# Creación del disco duro
$ VBoxManage createhd --filename "/home/dario/Escritorio/TFGDario/VMs/Windows_10_Cliente1
    /Windows_10_Cliente1.vdi" --size 30000

# Creación del controlador de almacenamiento
$ VBoxManage storagectl "Windows_10_Cliente1" --name "SATA Controller" --add sata --
    controller IntelAhci

# Montaje del disco duro
$ VBoxManage storageattach "Windows_10_Cliente1" --storagectl "SATA Controller" --port 0
    --device 0 --type hdd --medium "/home/dario/Escritorio/TFGDario/VMs/
    Windows_10_Cliente1/Windows_10_Cliente1.vdi"

# Montaje de la ISO de instalación
$ VBoxManage storageattach "Windows_10_Cliente1" --storagectl "SATA Controller" --port 1
    --device 0 --type dvddrive --medium "/home/dario/Escritorio/TFGDario/VMs/ISOs/
    Win10_22H2_Spanish_x64v1.iso"

# Montaje de VBoxGuestAdditions
$ VBoxManage storageattach "Windows_10_Cliente1" --storagectl "SATA Controller" --port 2
    --device 0 --type dvddrive --medium "/usr/share/virtualbox/VBoxGuestAdditions.iso"

# Configuración del arranque desde DVD
$ VBoxManage modifyvm "Windows_10_Cliente1" --boot1 dvd
```

Listado 7.4: Creación y configuración de la VM01

### Configuración de red

Para establecer la **configuración de red** de esta máquina se utilizan los siguientes comandos, con los que se asignamos la dirección IP estática **192.168.1.11** a la interfaz de red **Ethernet** y configuramos el **servidor DNS** principal con la IP **192.168.1.10**, que es la VM00.

```
New-NetIPAddress -InterfaceAlias "Ethernet" -IPAddress 192.168.1.11 -PrefixLength 24 -
    DefaultGateway 192.168.1.1
Set-DnsClientServerAddress -InterfaceAlias "Ethernet" -ServerAddresses 192.168.1.10
```

Listado 7.5: Configuración de red de la VM01

La configuración de red final de la VM01 es:

```
$ ipconfig

Configuración IP de Windows

Adaptador de Ethernet Ethernet:
   Sufijo DNS específico para la conexión. . :
   Vínculo: dirección IPv6 local. . . : fe80::1ff:79a1:30c7:ed38%8
   Dirección IPv4. . . . . . . . . . . . . . : 192.168.1.11
   Máscara de subred . . . . . . . . . . . . : 255.255.255.0
   Puerta de enlace predeterminada . . . . . : 192.168.1.1
```

Con los siguientes comandos cambiamos el nombre del equipo a **ELAM** y agregamos la máquina al dominio **PERSIA.UVA.ES**, lo cual nos va a permitir iniciar sesión con el **usuarioVictima** del dominio desde la VM01.

```
Rename-Computer -NewName "ELAM" -Force -Restart
Add-Computer -DomainName "PERSIA.UVA.ES" -Credential "PERSIA.UVA.ES\textbackslash
    Administrador" -Restart
```

### Instalación de Winlogbeat

El proceso de instalación y configuración de **Winlogbeat** es igual al de la VM00, con el pequeño detalle de que en este caso el campo personalizado **origin** tiene el valor **usuarioVictima** para identificar el origen de los logs.

**Código winlogbeat.yml**:

```
###################### Winlogbeat Configuration ########################

winlogbeat.event_logs:
  - name: Application
    ignore_older: 72h
  - name: System
  - name: Security
  - name: Microsoft-Windows-Sysmon/Operational
  - name: Windows PowerShell
    event_id: 400, 403, 600, 800
  - name: Microsoft-Windows-PowerShell/Operational
    event_id: 4103, 4104, 4105, 4106

output.kafka:
  hosts: ["192.168.1.34:9092"]
  topic: "winlogbeat-logs"

setup.kibana:
  host: "http://192.168.1.34:5601"

processors:
  - add_cloud_metadata:
      enabled: false
  - add_fields:
      target: ''
      fields:
        origin: "usuarioVictima"

logging.level: info
logging.to_files: true
logging.files:
  path: C:/ProgramData/winlogbeat/logs
  name: winlogbeat.log
  keepfiles: 7
  permissions: 0644
  rotateeverybytes: 10485760
```

Listado 7.6: Archivo winlogbeat.yml de configuración de Winlogbeat de la VM01

## 7.3 VM02 - Sistema de detección de intrusiones - Zeek

Esta máquina cuenta con **Zeek** instalado como **IDS** y está configurada para monitorizar el tráfico de red generado por las máquinas VM00 y VM01.

Todos los logs referentes a la actividad del **tráfico de red** son enviados a la Kafka mediante **Filebeat** para que puedan ser procesados y analizados, permitiendo generar alertas en base a comportamientos sospechosos y analizar la actividad de las VMs monitorizadas.

**Características**

| Nombre | Función | CPU | RAM | Disco | SO |
|---|---|---|---|---|---|
| VM02 | IDS | 2 Núcleos | 2 GiB | 20GB | Ubuntu 24.04.1 LTS |

Cuadro 7.3: Características de la VM02 - IDS Zeek

**Creación y configuración de la VM**

```
# Creación de la VM
$ VBoxManage createvm --name "Ubuntu_Sonda" --ostype "Ubuntu_64" --register

# Configuración de recursos de la VM
$ VBoxManage modifyvm "Ubuntu_Sonda" --memory 2048 --cpus 2 --nic1 intnet --intnet1 "
    LabNetwork"

# Creación del disco duro
$ VBoxManage createhd --filename "/home/dario/Escritorio/TFGDario/VMs/Ubuntu_Sonda/
    Ubuntu_Sonda.vdi" --size 20000

# Creación del controlador de almacenamiento
$ VBoxManage storagectl "Ubuntu_Sonda" --name "SATA Controller" --add sata --controller
    IntelAhci

# Montaje del disco duro
$ VBoxManage storageattach "Ubuntu_Sonda" --storagectl "SATA Controller" --port 0 --
    device 0 --type hdd --medium "/home/dario/Escritorio/TFGDario/VMs/Ubuntu_Sonda/
    Ubuntu_Sonda.vdi"

# Montaje de la ISO de instalación
$ VBoxManage storageattach "Ubuntu_Sonda" --storagectl "SATA Controller" --port 1 --
    device 0 --type dvddrive --medium "/home/dario/Escritorio/TFGDario/VMs/ISOs/ubuntu
    -24.04.1-desktop-amd64.iso"

# Configuración del arranque desde DVD
$ VBoxManage modifyvm "Ubuntu_Sonda" --boot1 dvd
```

Listado 7.7: Creación y configuración de la VM02

**Configuración de red**

Editamos el archivo **/etc/netplan/01-netcfg.yaml** para establecer la IP estática **192.168.1.20** a la interfaz de red **enp0s3** y establecer el servidor DNS principal como **192.168.1.10**, que es la VM00:

```
network:
  version: 2
  renderer: networkd
  ethernets:
```

```
    enp0s3:
      dhcp4: no
      addresses:
        - 192.168.1.20/24
      routes:
        - to: default
          via: 192.168.1.1
      nameservers:
        addresses:
          - 192.168.1.10
```

Listado 7.8: Configuración de red de la VM02

Una vez editado el archivo, ejecutamos el siguiente comando para aplicar la nueva configuración:

```
sudo netplan apply
```

La configuración de red final de la VM02 es:

```
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen
    100
    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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
    default qlen 1000
    link/ether 08:00:27:cd:dd:f6 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.20/24 brd 192.168.1.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fecd:ddf6/64 scope link
       valid_lft forever preferred_lft forever
```

**Instalación de Zeek**

Instalamos las dependencias necesarias para compilar e instalar Zeek:

```
sudo apt install -y cmake make gcc g++ flex bison libpcap-dev libssl-dev python3 python3-
    pip zlib1g-dev swig libmaxminddb-dev

sudo apt install curl -y
```

Importamos de la clave GPG del repositorio Zeek, esto garantiza que los paquetes del repositorio sean auténticos:

```
curl -fsSL https://download.opensuse.org/repositories/security:zeek/xUbuntu\_22.04/
    Release.key | sudo gpg --dearmor -o /usr/share/keyrings/security\_zeek-archive-
    keyring.gpg
```

Creamos el archivo **zeek.list** para añadir el repositorio de Zeek a la configuración de APT, lo que nos permite instalar Zeek desde fuentes confiables:

```
deb [signed-by=/usr/share/keyrings/security_zeek-archive-keyring.gpg] http://download.
    opensuse.org/repositories/security:zeek/xUbuntu_22.04/ /
```

Actualizamos los repositorios e instalamos Zeek:

```
sudo apt update
sudo apt install zeek
```

### Configuración de Zeek

En el archivo de configuración **/opt/zeek/etc/node.cfg** especificamos el **tipo de nodo Zeek**, la interfaz de red que se va a monitorizar y el host. El tipo de nodo establecido es **standalone**, lo que quiere decir que Zeek se ejecutará de forma independiente y esta máquina será la única encargada de capturar el tráfico de red.

```
[zeek]
type=standalone
host=localhost
interface=enp0s3
```

Listado 7.9: Archivo /opt/zeek/etc/node.cfg de configuración de Zeek de la VM02

Ahora editamos el archivo **/opt/zeek/share/zeek/site/local.zeek** para habilitar la exportación de logs en formato **JSON**. Para ello añadimos la siguiente línea:

```
redef LogAscii::use\_json = T;
```

### Instalación y configuración de Filebeat

Descargamos e instalamos Filebeat:

```
curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.10.1-amd64.
    deb
sudo dpkg -i filebeat-8.10.1-amd64.deb
```

Editamos el archivo de configuración **/etc/filebeat/filebeat.yml**, donde definimos los archivos de logs que Filebeat debe enviar a Kafka. En este caso, Filebeat enviará al **topic zeek-logs** de Kafka los archivos **conn.log** y **dns.log**, que contienen la información sobre las conexiones de red y las peticiones DNS observadas por Zeek.

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /opt/zeek/logs/current/conn.log
    - /opt/zeek/logs/current/dns.log

output.kafka:
  hosts: ["192.168.1.34:9092"]
  topic: "zeek-logs"
  codec.format:
    string: '%{[message]}'
```

Listado 7.10: Archivo /etc/filebeat/filebeat.yml de configuración de Filebeat de la VM02

Por último activamos el **modo promiscuo** para que Zeek pueda monitorizar todo el tráfico entre las otras VMs

```
VBoxManage modifyvm "Ubuntu\_Sonda" --nicpromisc1 allow-all
```

## 7.4 VM03 - Kafka & ELK

En esta máquina se encuentran desplegados los servicios de **Zookeeper**, **Kafka**, **Logstash**, **Elasticsearch** y **Kibana** en contenedores Docker. Su función es centralizar la recolección, el procesamiento, el almacenamiento y la visualización de los logs generados en el entorno de pruebas.

Recibe los logs generados por Winlogbeat y Zeek. Esto permite una visión centralizada de la actividad en los equipos monitorizados, lo que facilita la detección de incidentes de seguridad.

**Características**

| Nombre | Función | CPU | RAM | Disco | SO |
|---|---|---|---|---|---|
| VM03 | Kafka-ELK | 2 Núcleos | 8 GiB | 30GB | Ubuntu 24.04.1 LTS |

Cuadro 7.4: Características de la VM03 - Kafka & ELK

**Creación y configuración de la VM**

```
# Creación de la VM
$ VBoxManage createvm --name "Ubuntu_Kafka" --ostype "Ubuntu_64" --register

# Configuración de recursos de la VM
$ VBoxManage modifyvm "Ubuntu_Kafka" --memory 8192 --cpus 2 --nic1 intnet --intnet1 "
    LabNetwork"

# Creación del disco duro
$ VBoxManage createhd --filename "/home/dario/Escritorio/TFGDario/VMs/Ubuntu_Kafka/
    Ubuntu_Kafka.vdi" --size 13000

# Creación del controlador de almacenamiento
$ VBoxManage storagectl "Ubuntu_Kafka" --name "SATA Controller" --add sata --controller
    IntelAhci

# Montaje del disco duro
$ VBoxManage storageattach "Ubuntu_Kafka" --storagectl "SATA Controller" --port 0 --
    device 0 --type hdd --medium "/home/dario/Escritorio/TFGDario/VMs/Ubuntu_Kafka/
    Ubuntu_Kafka.vdi"

# Montaje de la ISO de instalación
$ VBoxManage storageattach "Ubuntu_Kafka" --storagectl "SATA Controller" --port 1 --
    device 0 --type dvddrive --medium "/home/dario/Escritorio/TFGDario/VMs/ISOs/ubuntu
    -24.04.1-desktop-amd64.iso"

# Configuración del arranque desde DVD
$ VBoxManage modifyvm "Ubuntu_Kafka" --boot1 dvd
```

Listado 7.11: Creación y configuración de la VM03

**Configuración de red**

Editamos el archivo **/etc/netplan/01-netcfg.yaml** para asignar la IP estática **192.168.1.34** a la interfaz de red **enp0s3**:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
```

```
      dhcp4: no
      addresses:
        - 192.168.1.34/24
      routes:
        - to: default
          via: 192.168.1.1
      nameservers:
        addresses:
          - 192.168.1.10
```

Listado 7.12: Archivo /etc/netplan/01-netcfg.yaml de configuración de red de la VM03

Una vez editado archivo ejecutamos el siguiente comando para aplicar la nueva configuración:

```
sudo netplan apply
```

La configuración de red final de la VM03 es:

```
kafka@kafka-VirtualBox:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen
    1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever

2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
    default qlen 1000
    link/ether 08:00:27:e9:b3:a5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.34/24 brd 192.168.1.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fee9:b3a5/64 scope link
       valid_lft forever preferred_lft forever
```

Una vez finalizada la configuración de red de la VM03 procedemos con la instalación de **Docker** y **Docker Compose**.

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

Para comprobar que está instalado ejecutamos los siguientes comandos, con los que podremos ver las versiones de Docker y Docker Compose instaladas:

```
$ docker -v
Docker version 28.0.2, build 0442a73

$ docker compose version
Docker Compose version v2.34.0
```

Seguimos con la instalación de **Visual Studio Code** desde la web oficial de Microsoft. Este editor de código facilitará la organización y el desarrollo del código.

### 7.4.1 Despliegue de la infraestructura

El código necesario para desplegar la infraestructura de Kafka y ELK en contenedores Docker está organizado de la siguiente forma:

```
kafka-elk-docker/
  docker-compose.yml
  .env
  logstash/
    config/
      logstash.yml
    pipeline/
      pipeline.conf
  elasticsearch/
    elasticsearch.yml
  kibana/
    kibana.yml
```

Listado 7.13: Organización de archivos para el despliegue del SIEM de la VM03

El archivo principal utilizado para desplegar la infraestructura es **docker-compose.yml**, en el que están definidos cinco contenedores, tres volúmenes persistentes y una red interna. Este archivo nos permite desplegar automáticamente todos los servicios necesarios para recibir, procesar, almacenar y visualizar logs.

Cada contenedor se encarga de desplegar un servicio distinto, todos ellos están conectados entre sí mediante una red interna llamada **elk_network**, la cual utiliza el driver **bridge**. Este tipo de driver crea una red virtual interna para los contenedores y permite que se comuniquen entre sí sin exponerse a redes externas.

Los volúmenes definidos en el archivo sirven para almacenar de forma persistente configuraciones y datos, por lo que estos no se perderán si los contenedores se reinician o eliminan. Además, para modificar la configuración por defecto de los servicios que lo requieran se montan volúmenes mediante **bind mounts**, lo cual permite establecer la configuración con archivos ubicados en la propia VM. De esta forma, se modifica la configuración por defecto de los servicios y se garantiza que los cambios realizados se mantengan incluso tras la eliminación de los contenedores.

Los servicios definidos en el **docker-compose** son los siguientes:

**Zookeeper:** este servicio se encarga de gestionar los **brokers** de Kafka. Un broker es una unidad de procesamiento de Kafka que almacena y distribuye los logs que llegan a los **topics**. Este proyecto cuenta con un solo broker, puesto que el volumen de datos no es elevado, pero es una opción muy importante para escalar la capacidad de almacenamiento y procesamiento en sistemas con un elevado número de eventos por segundo.

**Kafka:** servicio encargado de recibir y almacenar los logs generados por las distintas fuentes en tiempo real. Estos se almacenan temporalmente en los topics hasta que son consumidos por Logstash.

La configuración de este servicio cuenta con 2 listeners:

- **Puerto 9093 (Internal):** utilizado para la comunicación entre contenedores dentro de elk_network. Logstash se conecta a través de este puerto para consumir logs.
- **Puerto 9092 (External):** expuesto a la red, permite que otras VMs envíen logs a Kafka desde fuera del entorno de Docker.

Este servicio depende de Zookeeper, por lo que se ha establecido una dependencia para que no se lance hasta que el servicio de Zookeeper esté activo.

Este servicio monta un volumen:

- **kafka_data**: donde se almacenan los topics, logs y offsets de Kafka.

También se ha configurado un **healthcheck** con el que podremos comprobar que el servicio se ha levantado correctamente.

**Elasticsearch:** servicio encargado de almacenar los logs. El sistema de almacenamiento se organiza mediante índices, en este caso existen dos índices: uno para los logs de Winlogbeat y otro para los de Zeek. Este servicio también permite realizar búsquedas en los índices mediante consultas.

Este contenedor monta dos volúmenes:

- **elasticsearch_data**: donde se almacenan los logs indexados en Elasticsearch.
- **elasticsearch.yml**, bind mount: archivo con las configuraciones necesarias para inicializar el servicio.

Elasticsearch se encuentra expuesto en el puerto **9200**, al cual se conectan otros servicios para enviar datos o realizar consultas.

Además, este servicio consta de un **healthcheck**, el cual se encarga de verificar que el estado del **clúster** sea **green**, lo cual quiere decir que Elasticsearch está funcionando correctamente y los datos están disponibles.

**Kibana:** servicio que proporciona la interfaz web desde la que podremos visualizar los logs almacenados en Elasticsearch. Permite crear **dashboards** para visualizar los logs y realizar búsquedas aplicando filtros sobre los logs. También se pueden configurar reglas de detección, que generan alertas automáticas cuando identifican un comportamiento sospechoso en los logs.

Este servicio depende de Elasticsearch por lo que no se lanza hasta que este esté activo y en estado saludable.

Se accede desde el navegador mediante la dirección: **http://localhost:5601**

Kibana monta 2 volúmenes:

- **kibana_data**: almacena los dashboards, filtros y configuraciones de usuario.
- **kibana.yml**, bind mount: contiene la configuración necesaria para lanzar el servicio y establecer la conexión con Elasticsearch.

**Logstash:** servicio que actúa como intermediario entre Kafka y Elasticsearch. Su función es recibir los logs de Kafka, procesarlos y enviarlos a Elasticsearch. La transferencia de datos entre estos servicios se realiza a través de la red **elk_network**. Además, el procesamiento de los datos por parte de Logstash permite normalizarlos o filtrarlos.

Este contenedor depende tanto de Kafka como de Elasticsearch, por lo que no se inicia hasta que ambos estén activos y saludables.

Monta dos volúmenes:

- **pipeline.conf**, bind mount: contiene el código que especifica el tratamiento de los logs que llegan desde Kafka antes de enviárselos a Elasticsearch.
- **logstash.yml**, bind mount: contiene la configuración con la que se desplegará el servicio.

**Código docker-compose.yml**:

```
version: '3.8'


services:
 zookeeper:
   image: zookeeper:3.7
   container_name: zookeeper
   environment:
```

```
    - ZOOKEEPER_SERVER_ID=1
    - ZOOKEEPER_LISTENER_PORT=2181
  ports:
    - "2181:2181"
  networks:
    - elk_network


kafka:
  image: confluentinc/cp-kafka:7.3.0
  container_name: kafka
  environment:
    - KAFKA_ZOOKEEPER_CONNECT=zookeeper:2181
    - KAFKA_ADVERTISED_LISTENERS=INTERNAL://kafka:9093,EXTERNAL://192.168.1.34:9092
    - KAFKA_LISTENER_SECURITY_PROTOCOL_MAP=INTERNAL:PLAINTEXT,EXTERNAL:PLAINTEXT
    - KAFKA_LISTENERS=INTERNAL://0.0.0.0:9093,EXTERNAL://0.0.0.0:9092
    - KAFKA_LISTENER_NAMES=INTERNAL,EXTERNAL
    - KAFKA_INTER_BROKER_LISTENER_NAME=INTERNAL
    - KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR=1
  ports:
    - 9092:9092
    - 9093:9093
  networks:
    - elk_network
  depends_on:
    - zookeeper
  volumes:
    - kafka_data:/var/lib/kafka/data
  healthcheck:
    test: ["CMD", "kafka-topics", "--list", "--bootstrap-server", "localhost:9092"]
    interval: 5s
    timeout: 10s
    retries: 10


elasticsearch:
  image: docker.elastic.co/elasticsearch/elasticsearch:8.17.0
  container_name: elasticsearch
  environment:
    - ELASTIC_PASSWORD=${ELASTIC_PASSWORD}
    - ES_JAVA_OPTS=-Xms4g -Xmx4g
    - ingest.geoip.downloader.enabled=false
    - xpack.security.enabled=false
  ports:
    - 9200:9200
  networks:
    - elk_network
  volumes:
    - ./elasticsearch/config/elasticsearch.yml:/usr/share/elasticsearch/config/
        elasticsearch.yml
    - elasticsearch_data:/usr/share/elasticsearch/data
  healthcheck:
    test: ["CMD-SHELL", "curl -s --fail http://localhost:9200/_cluster/health?
        wait_for_status=green&timeout=30s"]
    interval: 30s
    timeout: 30s
    retries: 3


kibana:
  image: docker.elastic.co/kibana/kibana:8.17.0
  container_name: kibana
```

```
    environment:
      - ELASTICSEARCH_HOSTS=http://elasticsearch:9200
      - ELASTICSEARCH_USERNAME=kibana_system
      - ELASTICSEARCH_PASSWORD=${KIBANA_PASSWORD}
    ports:
      - 5601:5601
    networks:
      - elk_network
    depends_on:
      elasticsearch:
        condition: service_healthy
    volumes:
      - ./kibana/config/kibana.yml:/usr/share/kibana/config/kibana.yml
      - kibana_data:/usr/share/kibana/data


  logstash:
    image: docker.elastic.co/logstash/logstash:8.17.0
    container_name: logstash
    environment:
      - xpack.monitoring.elasticsearch.hosts=http://elasticsearch:9200
      - xpack.monitoring.elasticsearch.username=elastic
      - xpack.monitoring.elasticsearch.password=${ELASTIC_PASSWORD}
    ports:
      - "5044:5044"
    networks:
      - elk_network
    volumes:
      - ./logstash/config/logstash.yml:/usr/share/logstash/config/logstash.yml
      - ./logstash/pipeline:/usr/share/logstash/pipeline
    depends_on:
      elasticsearch:
        condition: service_healthy
      kafka:
        condition: service_healthy


networks:
  elk_network:
    driver: bridge


volumes:
  kibana_data:
  elasticsearch_data:
  kafka_data:
```

Listado 7.14: Archivo docker-compose.yml de la VM03

En el mismo directorio que el **docker-compose.yml** se encuentra el archivo **.env**, que es donde se almacenan las variables de entorno de los servicios definidos en **docker-compose**. Esto permite separar la información sensible, como las contraseñas del propio archivo de configuración.

**Código .env**:

```
ELASTIC_PASSWORD=aj48bi99PP10tt
LOGSTASH_INTERNAL_PASSWORD=03as34er45gh89
KIBANA_PASSWORD=hg10PA03rb45PG
```

Listado 7.15: Archivo .env de la VM03

Los dos siguientes archivos corresponden a la configuración del servicio de **Logstash**. En el archivo **logstash.yml** se definen los parámetros necesarios para ejecutar el servicio correctamente, como la dirección en la que escuchará **http.host: 0.0.0.0** para permitir conexiones desde el entorno Docker. También se define la ruta del **pipeline** con el que se procesarán los datos y la ubicación de los logs y datos generados por este servicio. Se ha habilitado la monitorización con X-Pack, que sirve para generar datos del rendimiento de Logstash y luego poder visualizarlo desde Kibana.

**Código logstash.yml**:

```
http.host: 0.0.0.0
path.config: /usr/share/logstash/pipeline
path.data: /usr/share/logstash/data
path.logs: /usr/share/logstash/logs
xpack.monitoring.elasticsearch.hosts: ${xpack.monitoring.elasticsearch.hosts}
xpack.monitoring.elasticsearch.password: ${xpack.monitoring.elasticsearch.password}
xpack.monitoring.elasticsearch.username: ${xpack.monitoring.elasticsearch.username}
```

Listado 7.16: Archivo logstash.yml de la VM03

En el archivo **pipeline.conf** se definen los procesos que Logstash aplicará a los logs generados por Winlogbeat y Zeek. En él se especifica cómo deben consumirse, procesarse y enviarse los logs. El archivo está dividido en tres secciones:

- **Input:** En esta parte se definen dos entradas de datos, una por cada topic de Kafka. Logstash se suscribe a los topics **winlogbeat-logs** y **zeek-logs** a través del puerto **9093**, que corresponde con el listener interno de Kafka.
- **Filter:** En esta sección se definen los procesos que Logstash aplicará a los logs para transformarlos, filtrarlos o enriquecerlos. En se ha implementado un filtro **JSON** que permite a Logstash interpretar correctamente los logs, ya que tanto Winlogbeat como Zeek los generan en este formato. Además se ha añadido la opción **skip_on_invalid_json** para que los logs que contengan errores o estén en otros formatos no se procesen.
- **Output:** En este apartado se definen los índices de Elasticsearch a los que se enviarán los datos procesados, en este caso se envían a dos índices distintos para separar los datos en función de la fuente que los generó.

**Código pipeline.conf**:

```
input {
 kafka {
   bootstrap_servers => "kafka:9093"
   topics => ["winlogbeat-logs"]
   group_id => "logstash-winlogbeat"
   auto_offset_reset => "earliest"
   consumer_threads => 2
   decorate_events => "basic"
 }

 kafka {
   bootstrap_servers => "kafka:9093"
   topics => ["zeek-logs"]
   group_id => "logstash-zeek"
   auto_offset_reset => "earliest"
   consumer_threads => 2
   decorate_events => "basic"
 }
}
```

```
filter {
 json {
   source => "message"
   skip_on_invalid_json => true
 }
}

output {
 if [agent][type] == "winlogbeat" {
   elasticsearch {
     hosts => ["http://elasticsearch:9200"]
     index => "winlogbeat-%{+YYYY.MM.dd}"
     user => "elastic"
     password => "aj48bi99PP10tt"
   }
 } else {
   elasticsearch {
     hosts => ["http://elasticsearch:9200"]
     index => "zeek-%{+YYYY.MM.dd}"
     user => "elastic"
     password => "aj48bi99PP10tt"
   }
 }

 stdout {
   codec => rubydebug
 }
}
```

Listado 7.17: Archivo pipeline.conf de la VM03

La configuración del servicio de **Elasticsearch** se define en el archivo **elasticsearch.yml**. En él se especifican las rutas donde se guardarán los datos y logs generados por este servicio, el puerto en el que se expone Elasticsearch, que es el **9200**, y se ha establecido el valor de **network.host: 0.0.0.0** para permitir conexiones desde cualquier interfaz en el entorno Docker.

También se ha habilitado la seguridad con X-Pack, lo cual añade una serie de funcionalidades como autenticación y control de accesos que mejoran la protección de los datos almacenados

Desde este archivo también se configura el número de nodos del clúster de Elasticsearch. Esto permite escalar el sistema añadiendo nuevos nodos para repartir la carga, mejorar el rendimiento y establecer un sistema de alta disponibilidad. También se pueden configurar los nodos para funciones específicas, como nodos de datos, nodos de ingesta y nodos maestros, que se encargan de la gestión del clúster.

En este caso, el sistema consta de un solo nodo, que es suficiente para el volumen de datos generado en el entorno de pruebas.

**Código elasticsearch.yml**:

```
path.data: /usr/share/elasticsearch/data
path.logs: /usr/share/elasticsearch/logs
network.host: 0.0.0.0
http.port: 9200
xpack.security.enabled: true
discovery.type: single-node
ingest.geoip.downloader.enabled: false
```

Listado 7.18: Archivo elasticsearch.yml de la VM03

Por último, tenemos el archivo **kibana.yml**, donde se definen los parámetros necesarios para lanzar correctamente el servicio de **Kibana**. En el archivo se especifica el puerto donde se ejecutará la interfaz web. También se indica la dirección de **Elasticsearch** a la que Kibana se conectará para acceder a los datos.

Además, se configura la clave de cifrado que utiliza **X-Pack** para proteger los objetos guardados en Kibana, como dashboards, reglas o configuraciones.

**Código kibana.yml**:

```
server.port: 5601
server.host: "0.0.0.0"
elasticsearch.hosts: ["http://elasticsearch:9200"]
elasticsearch.username: "kibana_system"
elasticsearch.password: "hg10PA03rb45PG"
logging:
  appenders:
    file:
      type: file
      fileName: /usr/share/kibana/logs/kibana.log
      layout:
        type: json
  root:
    appenders:
      - default
      - file
pid.file: /usr/share/kibana/kibana.pid
xpack.encryptedSavedObjects.encryptionKey: "3jcvMPWvDpyodB6UBoLUl4R4o9t5zZY2pZ/W3ayTE9k="
```

Listado 7.19: Archivo kibana.yml de la VM03

Una vez creados todos los archivos mencionados anteriormente, ya está todo listo para desplegar todos los servicios definidos en el **docker-compose.yml**. Para ello, ejecutaremos el siguiente comando:

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

Este comando lanza todos los contenedores en segundo plano. Para confirmar que todos los servicios se han iniciado correctamente, ejecutamos el siguiente comando:

```
docker ps -a
```

Este comando muestra el estado de cada contenedor. Si el **STATUS** de cada contenedor es "**Up (healthy)**"para los que tienen configurado un **healthcheck** y "**Up**"para los demás, quiere decir que el servicio está levantado y funcionando correctamente.

Una vez comprobado que todos los servicios están levantados, seguimos con la creación de los **topics** en Kafka. Esto solo será necesario realizarlo una vez, ya que la configuración se guardará de forma persistente en el volumen **kafka_data**. Crearemos dos topics, **winlogbeat-logs** donde se enviarán los logs generados por Winlogbeat y **zeek-logs** para los logs de Zeek.

Con los siguientes comandos crearemos los **topics**:

```
docker exec -it kafka kafka-topics --create --topic winlogbeat-logs --bootstrap-server
    localhost:9092

docker exec -it kafka kafka-topics --create --topic zeek-logs --partitions 1 --
    replication-factor 1 --bootstrap-server localhost:9092
```

En los que definimos el nombre del topic, el número de particiones y el factor de replicación.

Estas opciones son muy importantes en entornos en los que el número de eventos por segundo es elevado. El número de particiones permite dividir el flujo de datos para ser procesado en paralelo. El factor de replicación define el número de réplicas que tendrá cada partición para garantizar la disponibilidad de los datos si uno de los nodos falla. Esto permite crear un sistema escalable y robusto.

En este caso se han mantenido los valores mínimos (**–partitions 1 –replication-factor 1**) ya que el sistema solo dispone de un **broker** y el volumen de datos no es elevado.

Una vez creados los **topics**, el sistema ya está preparado para recibir los logs de las fuentes.

Ahora con el servicio de Kibana levantado accedemos a **http://localhost:5601**, donde veremos la interfaz principal.

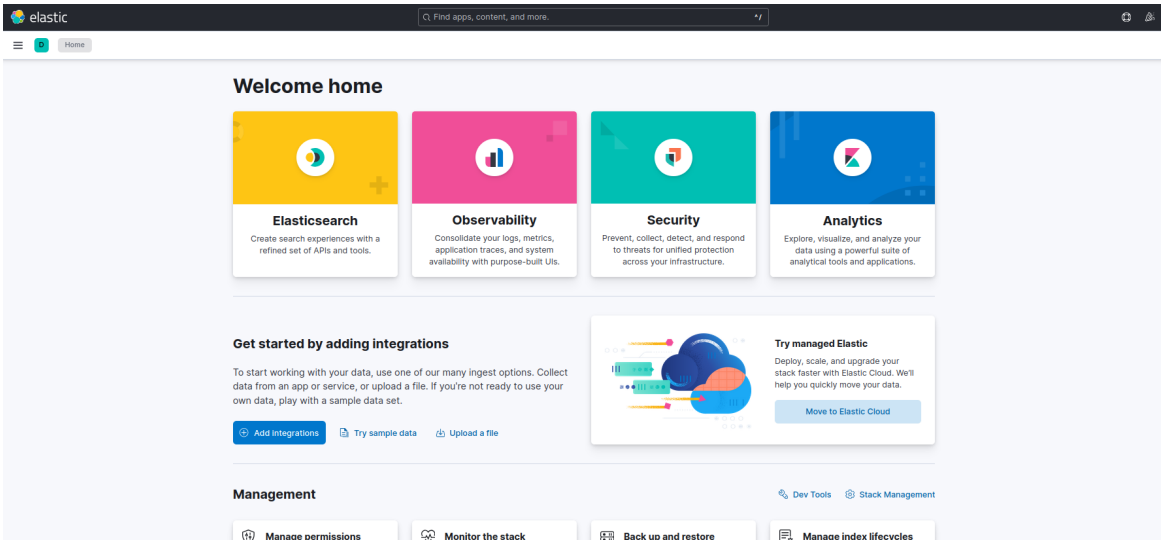


Figura 7.2: Interfaz principal de Kibana

Dentro de Kibana crearemos los **Data Views**, que son vistas que permiten visualizar y realizar consultas a los datos almacenados en Elasticsearch. Kibana facilita una interfaz para la creación de estos donde tendremos que introducir el nombre del Data View y el patrón con el que agrupará todos los datos de los índices que empiecen por un prefijo concreto:

| Nombre Data View | Prefijo |
|---|---|
| Winlogbeat | `winlogbeat-*` |
| Zeek | `zeek-*` |

Cuadro 7.5: Patrones utilizados para agrupar índices en Kibana

Esto es porque se pueden generar múltiples índices en Elasticsearch con ese prefijo, por ejemplo uno por cada día (**winlogbeat-2025.05.06**, **winlogbeat-2025.05.07**). De esta forma todos los datos se agruparán en el mismo **Data View** en Kibana.

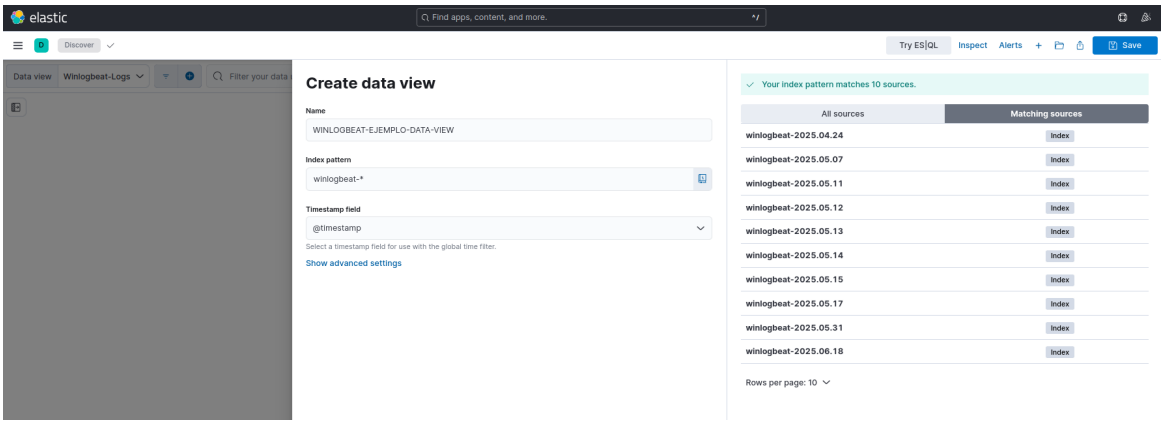


Figura 7.3: Ejemplo creación Data View

### 7.4.2 Creación de reglas

Tras crear los Data Views seguimos con la creación de las reglas de detección, estas son consultas automatizadas que sirven para generar alertas cuando se cumplen unas condiciones específicas en los logs indexados en Elasticsearch.

Kibana ofrece una interfaz para crear y ejecutar las reglas en tiempo real. Estas reglas se ejecutan periódicamente y generan una alerta en caso de cumplir las condiciones definidas en ellas.

Estas reglas se pueden definir de distintas maneras, por un lado tenemos las reglas de tipo **Log threshold** que no requieren escribir código y directamente se definen las condiciones en la interfaz que ofrece Kibana. Para consultas más avanzadas Kibana también permite crear reglas basadas en querys **ES|QL** (Elasticsearch Query Language).

Además, Kibana permite configurar acciones automáticas que se ejecutan en el momento que la regla se activa. En este caso, hemos creado un conector que enviará una alerta al **Data View ALERTAS**, de tal forma que, cada vez que se active cualquier regla, el Data View se actualizará y podremos ver de forma centralizada qué alerta se activó y a qué hora.

**Regla 1: Más de 100 eventos de login fallido en el equipo ELAM**

Esta regla genera una alerta cuando detecta 100 o más eventos de login fallido (evento 4625 de Windows) en el host ELAM del dominio PERSIA.UVA.ES en un intervalo de menos de 5 minutos.

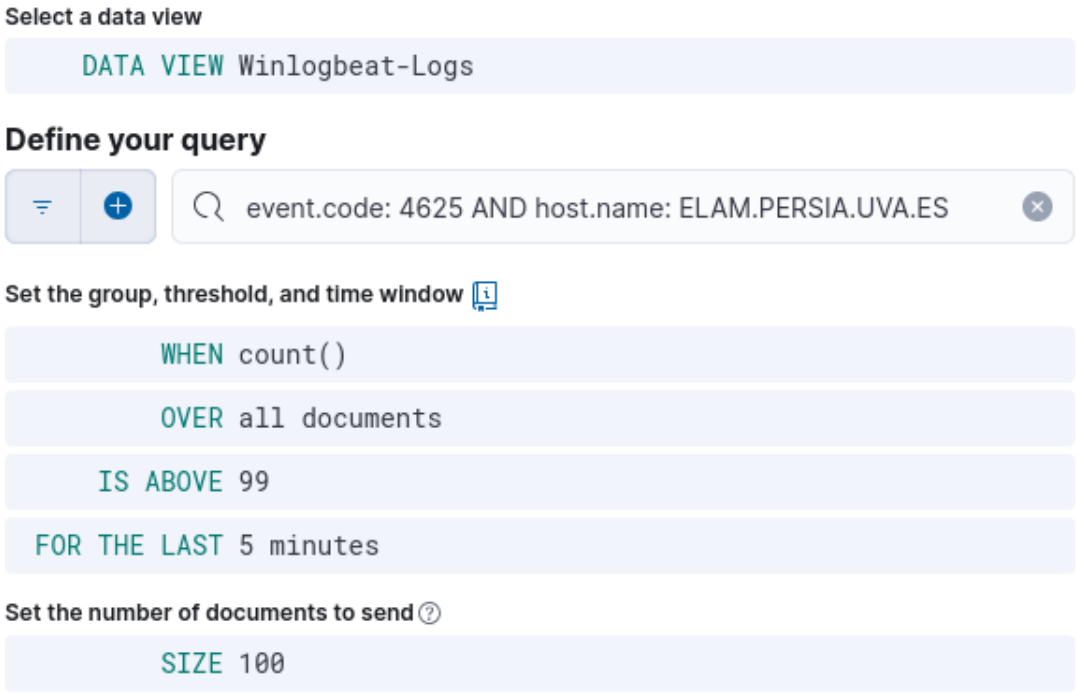


Figura 7.4: Regla de detección de fuerza bruta configurada en Kibana

Un elevado número de fallos de login en poco tiempo en un equipo puede ser indicativo de un ataque de fuerza bruta.

En el momento que salta esta alerta, permite investigar los eventos fallidos y determinar si se trata de un ataque real, y en ese caso si ha tenido éxito.

Detectar estos ataques en tiempo real permite aplicar medidas de contención de forma inmediata, lo que ayuda a mitigar el impacto del ataque y a proteger los sistemas frente a intrusiones.

**Regla 2: Escaneo de puertos interno desde ELAM**

Esta regla genera una alerta cuando se detectan peticiones desde el equipo ELAM hacia el servidor Server-AD contra más de 50 puertos distintos en un intervalo de 2 minutos.

Para ello se ha creado una consulta en ES|QL que verifica este comportamiento cada minuto:

```
FROM "zeek-*"
| WHERE id.orig_h == "192.168.1.11" AND id.resp_h == "192.168.1.10"
| STATS unique_ports = count_distinct(id.resp_p) BY id.orig_h
| WHERE unique_ports > 50
```

Un escaneo de puertos desde un equipo de nuestra red interna es un comportamiento muy sospechoso, ya que esto se utiliza para identificar los servicios expuestos en otros sistemas y evaluar si son vulnerables.

Esta actividad puede indicar un intento de movimiento lateral dentro de la red, sobre todo si este comportamiento se realiza desde un equipo comprometido.

Una vez detectado este comportamiento, se investigará para determinar si se trata de una actividad controlada, ya que es habitual realizar auditorías internas en busca de vulnerabilidades.

En caso de no ser una actividad legítima, el detectarla en tiempo real permite actuar de forma proactiva, conteniendo el equipo antes de que explote una posible vulnerabilidad.

**Regla 3: Peticiones desde Server-AD hacia una IP externa**

Esta regla detecta si Server-AD establece conexiones con IP externas.

Para ello se ha creado una consulta en ES|QL que verifica este comportamiento cada 5 minutos:

```
FROM "zeek-*"
| WHERE id.orig_h == "192.168.1.10"
 AND NOT id.resp_h IN ("192.168.1.10", "192.168.1.11", "192.168.1.20", "192.168.1.34")
```

Es importante monitorizar las peticiones hacia el exterior desde equipos de nuestra red interna y más si el sistema es crítico, como en este caso que se trata de un controlador de dominio.

Detectar peticiones hacia IPs externas desconocidas puede ser indicativo de ataques como:

- Exfiltración de datos
- Conexión hacia servidores **Command and Control** que permiten a un atacante tener control remoto sobre un equipo comprometido
- Descarga de **malware** en el equipo

Detectar a tiempo este tipo de comportamiento permite investigar si se trata de un comportamiento inusual o malicioso. En caso de ser una intrusión, podremos aislar el equipo para minimizar el impacto del ataque.

Todas las reglas creadas aparecerán en el apartado de Rules de Kibana. Desde esta interfaz podremos gestionar cada una de ellas, verificar su estado o ver si han generado alguna alerta.

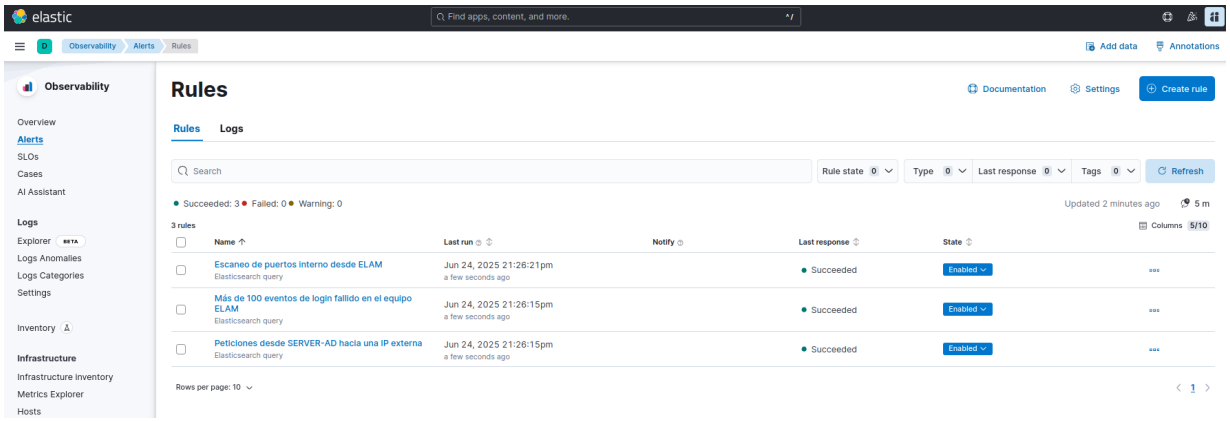


Figura 7.5: Interfaz Rules de Kibana

### 7.4.3 Creación de dashboards

Por último, creamos los dashboards, donde podremos visualizar e interactuar con los datos generados por Winlogbeat y Zeek.

**Dashboard Directorio Activo**

Este dashboard muestra la información de los eventos de Directorio Activo, a partir de la información de los logs generados por Winlogbeat. Para representar la información más relevante de cara a realizar una investigación, se han seleccionado los siguientes campos de los logs, que serán los representados en gráficas en el dashboard:

- **Task**: Tipo de tarea (User Account Management, Logon, Policy Change, Privilege Use, etc).

- **TargetUserName**: Nombre del usuario afectado por el evento.
- **Event Code**: Código de evento de Windows.
- **Hostname**: Equipo donde se ha producido el evento.
- **Source IP**: IP desde la que se ha producido el evento.
- **Error Code**: Código de error asociado al evento.
- **Error Code Sub**: Subcódigo de error asociado al evento.
- **Keyword**: Descripción del tipo de evento (Auditoría correcta, Elevación de Token, Creación de proceso, etc).
- **SubjectUserName**: Nombre del usuario que realizó el evento.

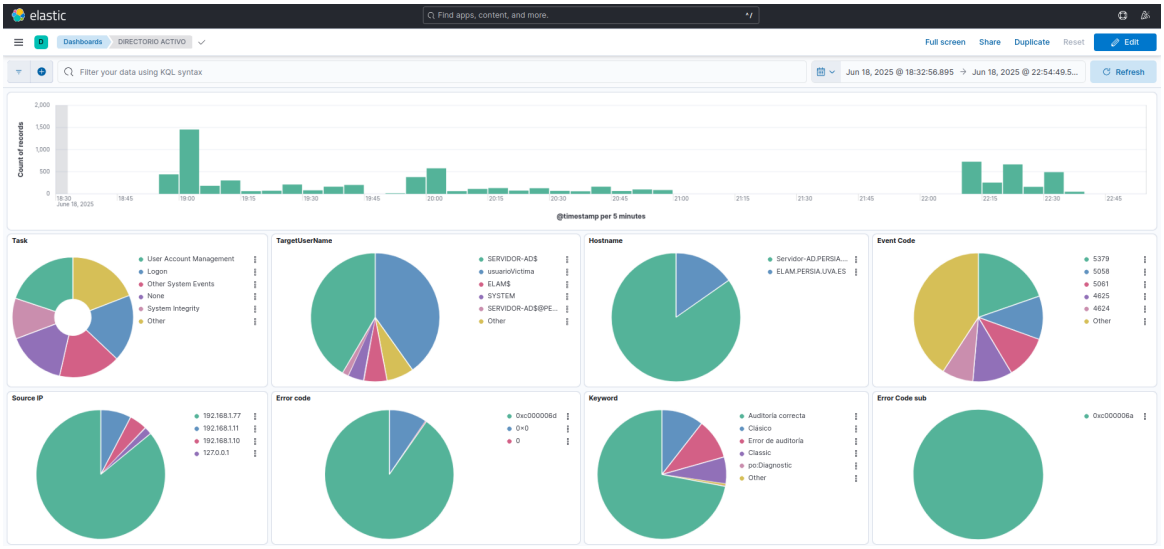


Figura 7.6: Dashboard de visualización de eventos de Winlogbeat en Kibana

Kibana permite interactuar con los gráficos para aplicar filtros de búsqueda, lo que facilita la investigación.

Además de la información de los Dashboards, también se pueden ver los logs en bruto desde el propio Data View de **winlogbeat-logs**, donde podremos filtrar para buscar cadenas específicas en todo el contenido de los logs, o buscar valores concretos en campos específicos.

### Dashboard Zeek IDS

Este dashboard muestra la información relacionada con los eventos de red, a partir de la información de los logs generados por el IDS Zeek. Para representar de forma visual la información más relevante se han seleccionado los siguientes campos:

**Campos seleccionados para las gráficas de Zeek:**

- **Source IP**: IP origen de la conexión.
- **Destination IP**: IP destino de la conexión.
- **Destination Port**: Puerto destino de la conexión.
- **Connection State**: Estado de la conexión:
  - `SF`: Establecimiento y finalización normal de la conexión.

- `SO`: Intento de conexión sin respuesta.
- `REJ`: Conexión rechazada.
- `RSTO`: Conexión abortada por el origen.
- `OTH`: Tráfico sin una conexión completa (no se detectó el paquete SYN).

- **Protocolo**: Protocolo utilizado en la conexión (TCP, UDP, ICMP, etc).
- **Service**: Servicio identificado en la conexión (DNS, HTTP, SSH, SMB, RDP, etc).
- **Source Port**: Puerto origen de la conexión.

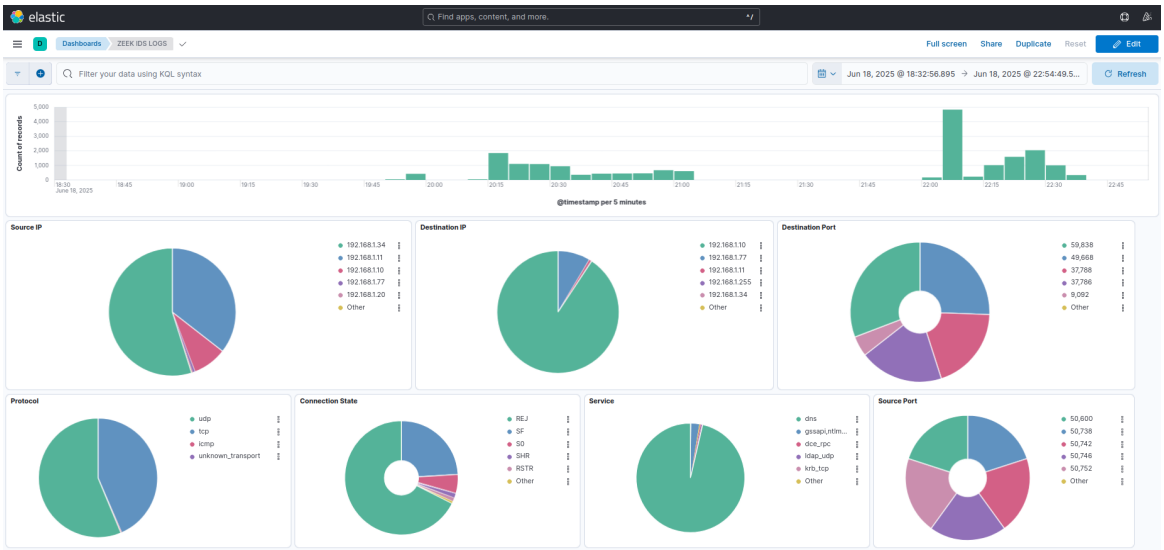


Figura 7.7: Dashboard de visualización de eventos del IDS Zeek en Kibana

Al igual que en el dashboard de Directorio Activo, desde este también se pueden aplicar filtros interactuando con los gráficos, lo que facilita la identificación de conexiones sospechosas.

Además, se pueden ver y filtrar los logs en bruto desde el Data View de **zeek-logs**.

## 7.5 VM04 - Atacante

Esta máquina simula a un **atacante externo** dentro del entorno de pruebas. La finalidad de esta máquina es ejecutar ataques controlados que generen tráfico y eventos maliciosos con el objetivo de comprobar el funcionamiento del sistema de detección y monitorización implementado.

**Características:**

| Nombre | Función | CPU | RAM | Disco | SO |
|---|---|---|---|---|---|
| VM04 | Atacante | 2 Núcleos | 2 GiB | 20GB | Ubuntu 24.04.1 LTS |

Cuadro 7.6: Características de la VM04 - Atacante

**Creación y configuración de la VM**

```
# Creación de la VM
$ VBoxManage createvm --name "Ubuntu_Caldera" --ostype "Ubuntu_64" --register

# Configuración de recursos de la VM
$ VBoxManage modifyvm "Ubuntu_Caldera" --memory 2048 --cpus 2 --nic1 intnet --intnet1 "
    LabNetwork"

# Creación del disco duro
$ VBoxManage createhd --filename "/home/dario/Escritorio/TFGDario/VMs/Ubuntu_Caldera/
    Ubuntu_Caldera.vdi" --size 20000

# Creación del controlador de almacenamiento
$ VBoxManage storagectl "Ubuntu_Caldera" --name "SATA Controller" --add sata --controller
     IntelAhci

# Montaje del disco duro
$ VBoxManage storageattach "Ubuntu_Caldera" --storagectl "SATA Controller" --port 0 --
    device 0 --type hdd --medium "/home/dario/Escritorio/TFGDario/VMs/Ubuntu_Caldera/
    Ubuntu_Caldera.vdi"

# Montaje de la ISO de instalación
$ VBoxManage storageattach "Ubuntu_Caldera" --storagectl "SATA Controller" --port 1 --
    device 0 --type dvddrive --medium "/home/dario/Escritorio/TFGDario/VMs/ISOs/ubuntu
    -24.04.1-desktop-amd64.iso"

# Configuración del arranque desde DVD
$ VBoxManage modifyvm "Ubuntu_Caldera" --boot1 dvd
```

Listado 7.20: Creación y configuración de la VM04

### Configuración de red

Editamos el archivo **/etc/netplan/01-netcfg.yaml** para asignar la IP estática **192.168.1.77** a la interfaz de red **enp0s3**, facilitando así la comunicación con el resto de máquinas del entorno de pruebas.

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
      dhcp4: no
      addresses:
        - 192.168.1.77/24
```

Listado 7.21: Archivo /etc/netplan/01-netcfg.yaml de configuración de red de la VM04

Una vez editado el archivo, se ejecuta el siguiente comando para aplicar la nueva configuración:

```
sudo netplan apply
```

La configuración de red final de la VM04 es:

```
atacante@atacante-VirtualBox:~\$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen
    1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
```

```
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group
    default qlen 1000
    link/ether 08:00:27:d4:a2:64 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.77/24 brd 192.168.1.255 scope global noprefixroute enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fed4:a264/64 scope link
       valid_lft forever preferred_lft forever
```

## 7.6 Despliegue de la infraestructura

En este apartado se resumen todos los comandos necesarios para iniciar y verificar el funcionamiento de cada componente de la infraestructura.

### VM00 y VM01

Lanzamiento de Winlogbeat en ambas máquinas, para ello ejecutaremos el siguiente comando en PowerShell:

```
Start-Service winlogbeat
```

Comprobaremos que Winlogbeat está activo con el siguiente comando:

```
Get-Service winlogbeat
```

La salida que recibiremos si está funcionando es:

```
Status          Name            DisplayName
------          ----            -----------
Running         winlogbeat      winlogbeat
```

### VM02

Ejecución de Filebeat, que será el encargado de enviar los logs generados por Zeek a Kafka. Para ello, ejecutamos el siguiente comando:

```
systemctl start filebeat
```

Y comprobamos que está en funcionamiento con:

```
systemctl status filebeat
```

Ejecución de Zeek con el siguiente comando:

```
zeekctl deploy
```

Ahora comprobaremos que el IDS está en funcionamiento:

```
zeekctl status
```

### VM03

Ejecución del docker-compose.yml, que levantará los servicios de Kafka, Zookeeper, Logstash, Elasticsearch y Kibana:

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

Una vez ejecutados los comandos anteriores, la infraestructura estará lista para recibir, procesar y visualizar los eventos generados en las máquinas monitorizadas. Ahora podremos llevar a cabo las pruebas para validar la capacidad de detección del sistema ante comportamientos sospechosos.

# Capítulo 8

# Pruebas

En este capítulo se desarrollarán una serie de escenarios de ataque para comprobar la eficacia del sistema implementado. Para ello se han creado 3 scripts con ayuda de chatGPT que simulan distintos comportamientos maliciosos hacia o desde las VM00 y VM01. Estas pruebas permiten comprobar si el sistema es capaz de detectar las actividades sospechosas provocadas por los scripts.

Se han elegido tres escenarios que representan distintos tipos de actividad maliciosa comunes: un ataque de fuerza bruta, un escaneo de puertos interno y peticiones hacia una IP externa desconocida. Cada uno de estos casos pone a prueba distintos componentes del sistema implementado. Por un lado, Winlogbeat se encarga de monitorizar los eventos de autenticación en los equipos Windows; por otro, Zeek detecta las conexiones inusuales asociadas al escaneo de puertos interno y al tráfico externo no autorizado. Estos escenarios permiten poner a prueba el sistema creado y validar que tanto Winlogbeat como Zeek recogen los eventos y los envían al SIEM, donde las reglas creadas generarán alertas en tiempo real frente a estos comportamientos sospechosos.

Se ha decidido poner a prueba el sistema con estos escenarios concretos porque representan situaciones que pueden darse en un entorno real. Además, son ejemplos de comportamientos que, si no se detectan a tiempo, pueden derivar en una intrusión, un movimiento lateral o una exfiltración de datos. Con estas pruebas se pretende verificar que el sistema es capaz de identificar este tipo de amenazas en tiempo real y generar alertas que permitan actuar con rapidez para reducir el impacto o evitar riesgos mayores.

## 8.1 Primer escenario - Ataque de fuerza bruta

Para simular un ataque de fuerza bruta se ha creado un script en bash que lanza un ataque de fuerza bruta contra el **usuarioVictima** y equipo **ELAM.PERSIA.UVA.ES** utilizando el protocolo SMB.

```
#!/bin/bash

TARGET="192.168.1.11"
USERNAME="usuarioVictima"
DOMAIN="PERSIA.UVA.ES"
PASSFILE="100-passwords.txt"
SHARE="C\$"

echo "[*] Iniciando intentos de login SMB contra $TARGET..."
```

```
while IFS= read -r PASSWORD
do
        echo "[!] Intentando $DOMAIN\\$USERNAME:$PASSWORD"
        smbclient "//$TARGET/$SHARE" -U "$DOMAIN\\$USERNAME%$PASSWORD" -c 'exit' > /dev/
            null 2>&1
done < "$PASSFILE"

echo "[+] Ataque de fuerza bruta completado."
```

Este tipo de ataque genera numerosos eventos de autenticación fallida en poco tiempo, por lo que un aumento significativo de intentos fallidos puede ser indicativo de un posible ataque.

Para detectar este comportamiento se creó la regla **Más de 100 eventos de login fallido en el equipo ELAM** que generará una alerta tras ejecutar el script como podemos ver en la Figura 8.1.




Figura 8.1: Dashboard alertas - Alerta ataque fuerza bruta

Una vez generada la alerta desde el dashboard de Directorio Activo podremos ver como a las 22:21 hay un pico de unos 100 eventos de login fallido en el host **ELAM**, del dominio **PERSIA.UVA.ES**, para autenticarse con el usuario **usuarioVictima**. También podemos ver que estos intentos de login se realizan desde la IP **192.168.1.77**, la cual no pertenece a nuestra red y que todos los intentos de autenticación fallan y generan el subcódigo de error **0x000006a**, que quiere decir “Inicio de sesión de usuario con contraseña incorrecta o mal escrita”.

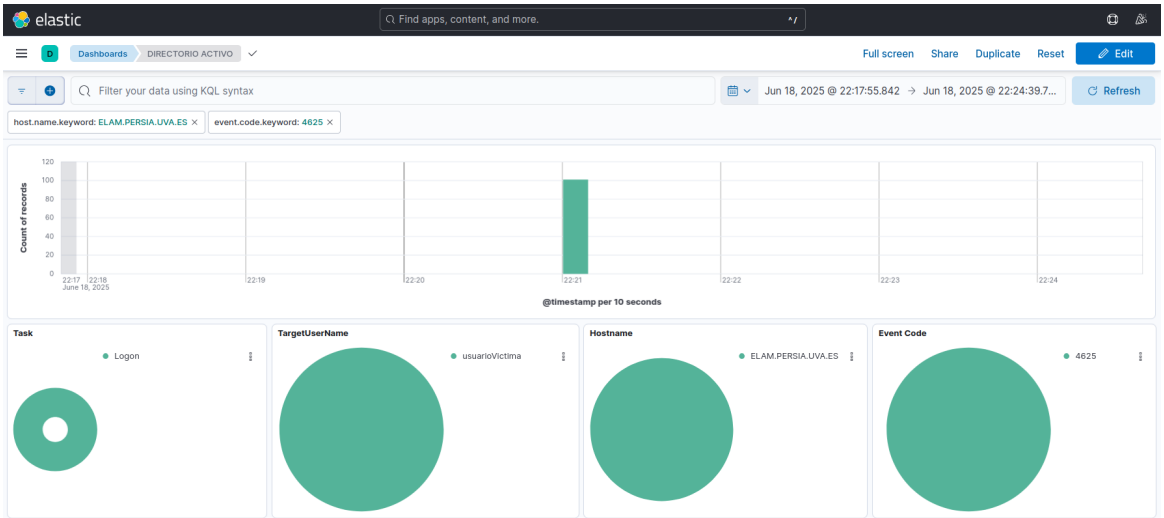


Figura 8.2: Dashboard Winlogbeat filtrado por host, evento y hora

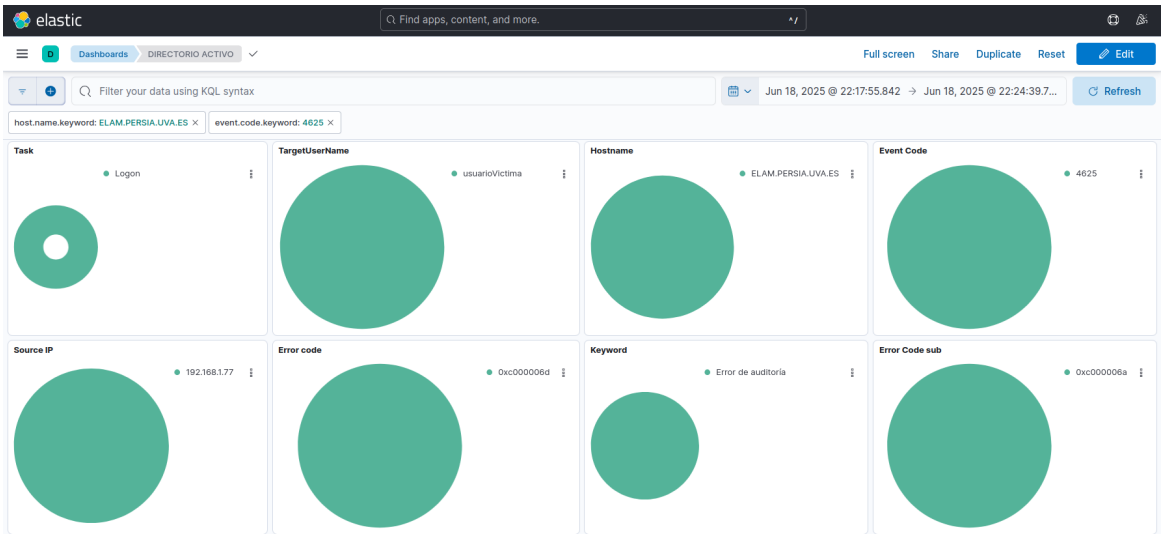


Figura 8.3: Dashboard Winlogbeat filtrado por host, evento y hora

Una vez detectado un ataque en tiempo real, es muy importante tomar acciones lo antes posible para mitigar su impacto antes de que se convierta en un incidente mayor o derive en una intrusión exitosa. En este caso sólo se han detectado eventos de login fallido, por lo que podemos descartar la intrusión, aun así podemos aplicar una serie de medidas para prevenir o reducir el impacto de ataques futuros:

- Bloquear la dirección IP de origen en el firewall para impedir futuros intentos de autenticación desde ese origen.
- Bloquear temporalmente la cuenta tras detectar un alto número de login fallidos.
- Aplicar políticas de retardo entre intentos de autenticación.

## 8.2 Segundo escenario - Escaneo de puertos interno

Un escaneo de puertos interno no es un ataque que tenga un impacto directo, pero es un comportamiento muy sospechoso. Mediante un escaneo de puertos se pueden identificar los servicios expuestos de la máquina objetivo, lo que permite descubrir vectores de ataque. Detectar este tipo de comportamiento es muy importante ya que puede ser indicio de una intrusión y de un intento de movimiento lateral.

El script creado para simular este comportamiento es un script en PowerShell que se ejecuta desde la VM01 contra la VM00. Recorre del puerto 20 al 150 intentando establecer conexión TCP con cada uno de ellos:

```
$target = "192.168.1.10"
$ports = 20..150

foreach ($port in $ports) {
    Write-Output "Probando puerto $port en $target..."

    try {
        $tcp = New-Object System.Net.Sockets.TcpClient
        $iar = $tcp.BeginConnect($target, $port, $null, $null)
        $success = $iar.AsyncWaitHandle.WaitOne(100)  # timeout en ms

        if ($success -and $tcp.Connected) {
            Write-Host " Puerto $port abierto"
            $tcp.Close()
```

```
        } else {
            Write-Host " Puerto $port cerrado o filtrado"
        }
    } catch {
        Write-Host " Error al probar puerto $port"
    }

    Start-Sleep -Milliseconds 50  # ralentiza para que Zeek pueda registrar
}
```

Para detectar este comportamiento se ha creado la regla de **Escaneo de puertos interno desde ELAM**, la cual se activa cuando se registran más de **50 conexiones** contra distintos puertos de el **Server-AD** desde el equipo **ELAM** en 2 minutos.

Al ejecutar este script, Zeek detectará todos los intentos de conexión TCP hacia distintos puertos de la VM00 y la alerta creada saltará, como se puede observar en Figura 8.4

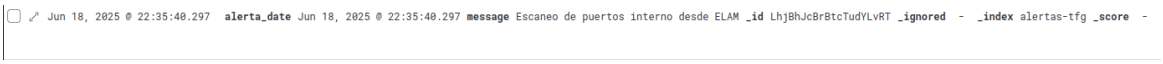


Figura 8.4: Dashboard alertas - Alerta escaneo de puertos

En la propia alerta podremos ver la hora exacta a la que se detectó el comportamiento sospechoso y los equipos implicados. Filtrando en Kibana por la fecha del 18 de junio a las 22:35 y por IP origen **192.168.1.11**, correspondiente al equipo **ELAM**, podremos ver numerosas peticiones hacia **192.168.1.10**, que corresponde con el **Server-AD**. A su vez, también podremos ver el estado de las conexiones, que en este caso es **REJ** para la gran mayoría, lo que indica que las conexiones fueron rechazadas debido a que los puertos destino estaban cerrados.

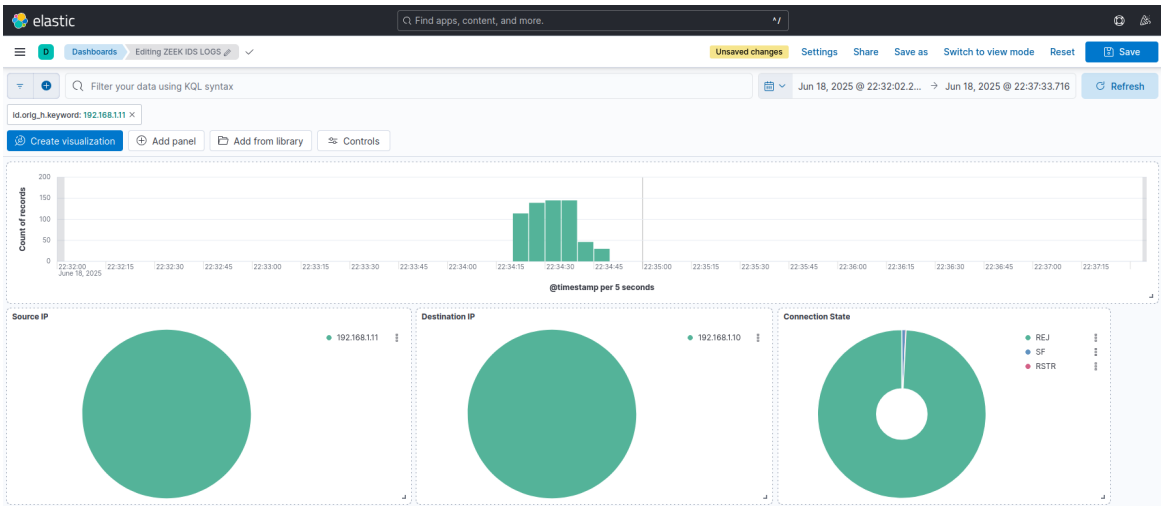


Figura 8.5: Dashboard Zeek filtrado por IP origen y hora

Si accedemos al Data View de **Zeek-Logs** y aplicamos los filtros de IP origen, IP destino y el momento en el que se generó la alerta, encontraremos todos los logs en bruto que provocaron su activación. En estos logs, entre otras cosas, se puede ver que cada petición estaba dirigida hacia un puerto destino distinto:

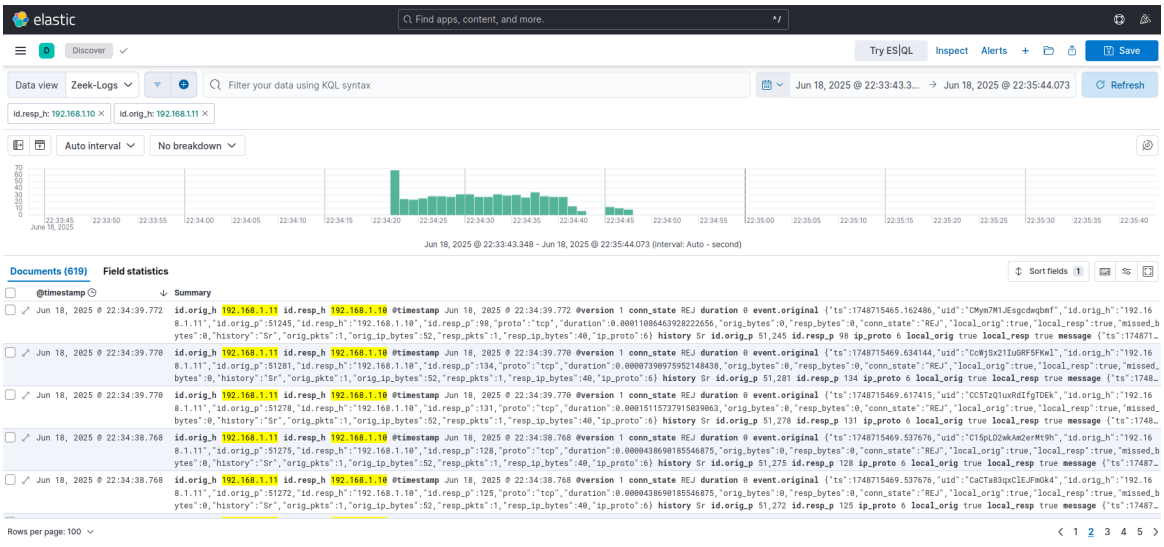

Figura 8.6: Data view Zeek-Logs filtrado por IP origen, IP destino y hora

Figura 8.7: Puertos destino observados en los logs de Zeek

La detección de este comportamiento nos permite actuar con rapidez ante una posible intrusión, dado que la regla está creada con la finalidad de detectar escaneos dentro de la red interna.

El hecho de haber detectado este comportamiento en tiempo real nos permite reaccionar de forma inmediata. Una vez detectado este comportamiento se pueden tomar diversas medidas como:

- Contener el equipo para aislarlo de la red y evitar que continúe realizando conexiones hacia otros equipos.
- Revisar la actividad del usuario asociado al equipo en el que se detectó el comportamiento sospechoso.

## 8.3 Tercer escenario - Peticiones hacia el exterior

Monitorizar las peticiones hacia el exterior es una buena práctica, sobre todo si los destinos son desconocidos, ya que esto puede ser un indicio de una intrusión o incluso una posible exfiltración de datos. Este tipo de tráfico puede indicar que un equipo ha sido comprometido.

Para simular este comportamiento se ha creado el siguiente script en PowerShell que realiza peticiones desde el **Server-AD (192.168.1.10)** hacia la IP **192.168.1.77**, dirección que no pertenece a nuestra red.

```
$target = "http://192.168.1.77:8888"
$requestCount = 10
$body = @{ data = "informacion confidencial ejemplo" }

for ($i = 1; $i -le $requestCount; $i++) {
    try {
        Invoke-WebRequest -Uri $target -Method POST -Body $body -TimeoutSec 3 | Out-Null
        Write-Host "[$i] POST enviada a $target"
    } catch {
        Write-Warning "[$i] Fallo al conectar con $target (puede no estar escuchando)"
    }
    Start-Sleep -Milliseconds 200
}
```

El script genera peticiones HTTP con un contenido de datos simulado hacia una IP externa.

Para detectar esto se ha creado la regla **Peticiones desde Server-AD hacia una IP externa**, que salta en caso de detectar peticiones hacia una IP externa.

Una vez ejecutado el script podemos ver desde Kibana como se ha activado la regla.Figura 8.8


Figura 8.8: Alerta por tráfico hacia el exterior

En la propia alerta podemos observar el nombre del equipo origen **Server-AD** y la hora a la que se produjo la actividad. Con esta información, filtramos desde Kibana por IP origen y fecha, lo que nos permite ver peticiones desde **192.168.1.10** hacia **192.168.1.77** contra el puerto **8888**.

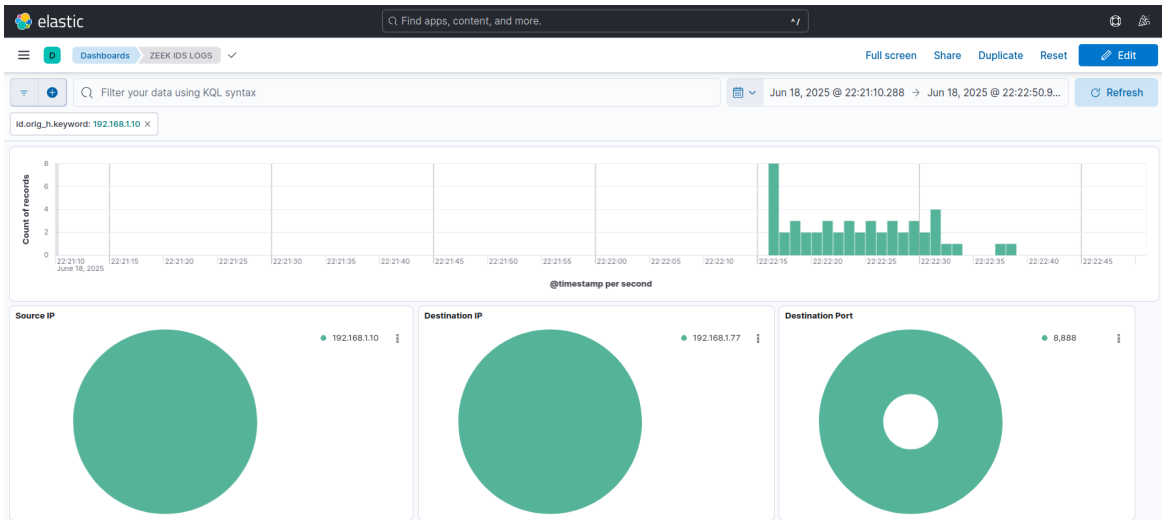


Figura 8.9: Dashboard Zeek filtrado por IP origen

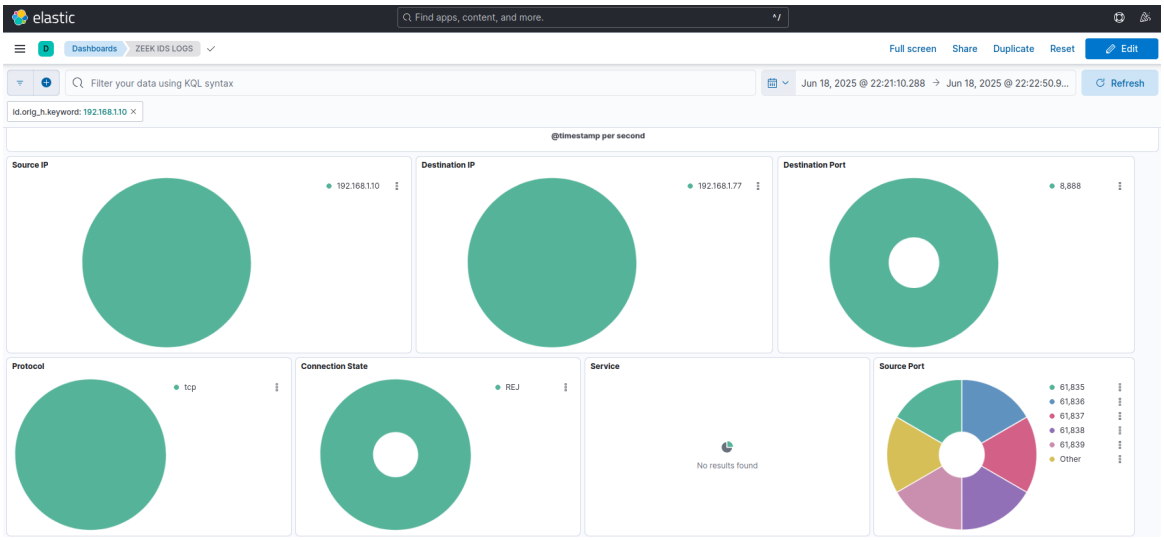


Figura 8.10: Dashboard Zeek filtrado por IP origen

Como se puede ver en el estado de las conexiones, todas son **REJ**, porque el puerto destino está cerrado. A pesar de esto, esta actividad es muy sospechosa sobre todo si el destino es desconocido o estas peticiones no se tratan de una actividad controlada.

Si tras realizar una investigación se determina que la IP de destino es maliciosa o está relacionada con actividades sospechosas, podremos aplicar en tiempo real medidas de contención y prevención para evitar una posible exfiltración de datos, la comunicación con un servidor Command and Control o la propagación del ataque dentro de nuestra red. Algunas de las medidas son:

- Aislar el equipo para impedir cualquier comunicación.
- Bloquear la dirección IP de destino en el firewall para impedir futuras conexiones.

A continuación se muestra el dashboard de **ALERTAS** tras la activación de las reglas:

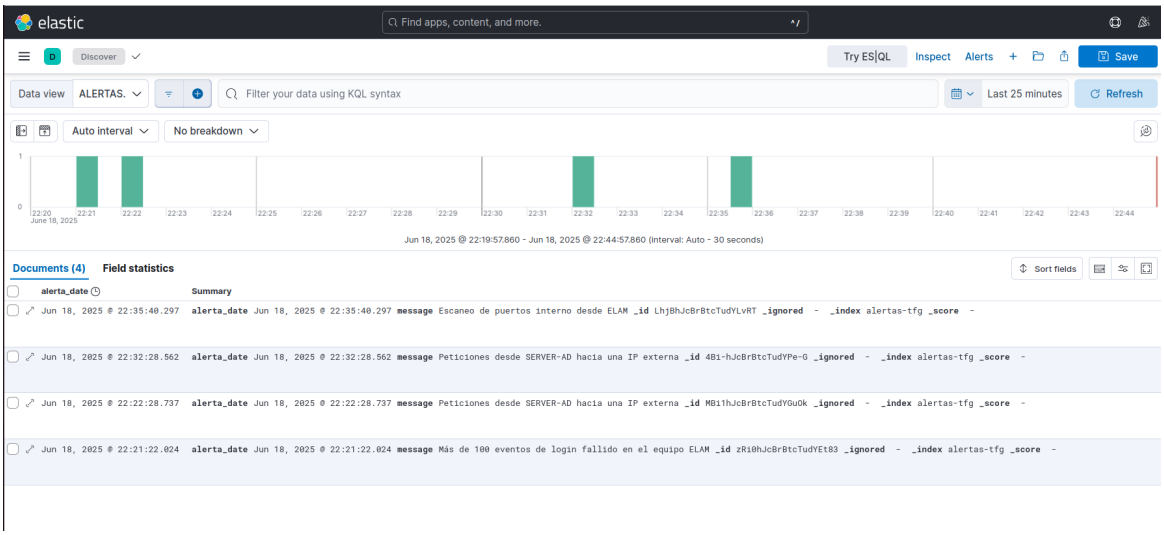


Figura 8.11: Dashboard Alertas

Esta vista centralizada permite a los analistas de seguridad monitorizar en tiempo real todos los comportamientos sospechosos detectados por las reglas creadas, permitiendo responder rápidamente ante posibles incidentes de seguridad.

# Capítulo 9

# Conclusiones

En este proyecto se ha llevado a cabo la implementación y configuración de una infraestructura SIEM desde cero, por lo que una gran parte del tiempo invertido en este trabajo ha sido para comprender cómo funciona y cómo desplegar una infraestructura de este tipo.

A lo largo del proyecto se han cumplido la mayoría de los objetivos propuestos:

- Se ha creado un entorno con Active Directory, formado por un controlador de dominio, un equipo y un usuario miembros de ese dominio, simulando una situación lo más real posible.
- Se ha instalado y configurado Winlogbeat en los equipos Windows del entorno de pruebas con el objetivo de recolectar los eventos generados por el sistema operativo y enviarlos al SIEM.
- Se ha instalado y configurado Zeek como sistema de detección de intrusiones con el objetivo de analizar el tráfico de red en el entorno de pruebas y generar logs que se envían al SIEM.
- El SIEM implementado con Kafka y ELK es capaz de recolectar, procesar y mostrar toda la actividad generada por Winlogbeat y Zeek, lo cual permite tener una visión centralizada de lo que sucede en nuestra red. Además, para facilitar la implementación, gestión y mantenimiento de los componentes del SIEM se ha utilizado Docker desplegando cada herramienta en contenedores individuales.
- Se han creado dos dashboards en Kibana, uno para la visualización de los logs de Zeek y otro para la visualización de los logs de Winlogbeat. En estos dashboards se muestra la información más relevante de cada fuente de logs, permitiendo realizar investigaciones y analizar comportamientos sospechosos.
- También se han creado tres reglas de detección, configuradas para generar alertas en tiempo real en el momento que detectan una condición preestablecida, la cual podría indicar un ataque o una intrusión.
- Se ha puesto a prueba la infraestructura implementada mediante la simulación de comportamientos maliciosos, lo que ha permitido validar el correcto funcionamiento de las reglas de detección creadas, y verificar que los dashboards creados en Kibana proporcionan una interfaz útil para investigar actividades sospechosas.

Uno de los objetivos iniciales que no se han cumplido es la simulación de ataques utilizando el framework Caldera. En su lugar, se han creado tres scripts que simulan comportamientos sospechosos, con la finalidad de verificar el correcto funcionamiento de las reglas de detección creadas. Estos scripts ponen a prueba la infraestructura creada, lo cual era uno de los objetivos principales del proyecto.

Por otro lado, la implementación de políticas ILM en Elasticsearch no se ha podido llevar a cabo por falta de tiempo. Sin embargo, se ha incluido una explicación teórica sobre su funcionamiento y beneficios en un SIEM.

## 9.1 Trabajo futuro

Para continuar con la mejora de la infraestructura implementada en este proyecto, se proponen una serie de puntos que permitirían mejorar sus capacidades:

- **Integración de más fuentes:** Actualmente el sistema solo recibe eventos generados por el IDS y Winlogbeat. La incorporación de más fuentes, como firewalls o EDR, ayudaría a tener una visión más completa de la actividad en toda la red. Esto también permitiría crear reglas más sofisticadas que añadirían más capas de seguridad.
- **Creación de reglas más generales:** Las reglas creadas en este proyecto son para casos específicos en equipos concretos. Una gran mejora sería la creación de reglas genéricas que puedan detectar comportamientos sospechosos en cualquier equipo de la red y generar un breve resumen de la actividad detectada.
- **Implementación de políticas ILM:** La implementación de estas políticas permitiría gestionar el ciclo de vida de los índices creados en Elasticserch. Esto ayudaría a optimizar el uso de los recursos del sistema y permitiría controlar de forma automatizada el almacenamiento de datos.
- **Correlación de eventos de diferentes fuentes:** Crear un sistema capaz de correlacionar datos procedentes de diversas fuentes mejoraría la detección de ataques más complejos, que de haber sido analizados por separado podrían pasar desapercibidos.
- **Mejora de la disponibilidad y rendimiento de Elasticsearch:** Aumentar el factor de replicación establecido en Elasticsearch mejoraría la tolerancia a fallos, ya que esto garantiza la duplicidad de los datos en distintos nodos. Además, esto mejoraría el rendimiento de las búsquedas al dividir la carga de las consultas entre los nodos que contienen réplicas.

La implementación de todas estas mejoras daría como resultado una infraestructura más robusta y eficaz en la detección de amenazas. La incorporación de nuevas fuentes, la creación de reglas más generales y la optimización del rendimiento son aspectos clave para crear un sistema de monitorización completo, capaz de responder ante escenarios reales de ataque.

# Apéndice A

# Glosario

**Active Directory**

Servicio de directorio de Microsoft utilizado para la gestión centralizada de usuarios, equipos y políticas en redes Windows.

**Clúster**

Grupo de nodos que trabajan de forma coordinada como si fueran un solo sistema.

**Data View**

Vista en Kibana que permite consultar y visualizar datos indexados en Elasticsearch.

**Dashboard**

Panel que representa datos mediante gráficos y tablas en Kibana.

**Endpoint**

Dispositivo conectado a una red, como ordenadores, servidores o móviles.

**Evento**

Acción registrada por un sistema, equivalente a un log.

**Firewall**

Sistema que controla y filtra el tráfico de una red actuando como barrera entre la red interna y redes externas.

**Fuentes**

Herramientas encargadas de generar eventos o logs que alimentan al SIEM con información.

**Nodo**

Instancia individual dentro de un clúster de Elasticsearch que procesa y almacena datos.

**Registro**

Sinónimo de log. Refleja un evento producido en un sistema.

**Reglas**

Condiciones definidas para detectar patrones o comportamientos anómalos en los logs generados por las fuentes.

**Topic**

Canal en Kafka que permite a los productores enviar mensajes y a los consumidores suscribirse para recibirlos.

# Apéndice B

# Lista de abreviaturas

**AD:** *Active Directory*

**AMQP:** *Advanced Message Queuing Protocol*

**C&C:** *Command and Control*

**CPU:** *Central Processing Unit*

**DDoS:** *Distributed Denial-of-Service*

**DNS:** *Domain Name System*

**EDR:** *Endpoint Detection and Response*

**ELK:** *Elasticsearch, Logstash, Kibana*

**HTTP:** *Hypertext Transfer Protocol*

**ICMP:** *Internet Control Message Protocol*

**IDS:** *Intrusion Detection System*

**ILM:** *Index Lifecycle Management*

**IP:** *Internet Protocol*

**IPS:** *Intrusion Prevention System*

**JSON:** *JavaScript Object Notation*

**OSS:** *Open Source Software*

**RDP:** *Remote Desktop Protocol*

**RAM:** *Random Access Memory*

**SIEM:** *Security Information and Event Management*

**SMB:** *Server Message Block*

**SO:** *Sistema Operativo*

**SSH:** *Secure Shell*

**TCP:** *Transmission Control Protocol*

**UDP:** *User Datagram Protocol*

**VM:** *Virtual Machine*

# Apéndice C

# Modificación Plan de Trabajo

El proyecto se ha retrasado 16 semanas. Este retraso ha sido causado por una dificultad técnica inesperada a la hora de implementar la infraestructura y, sobre todo, por falta de tiempo para compaginar el desarrollo del TFG con otras tareas. Estos retrasos ya se habían considerado en los riesgos R05 y R07, cuyo impacto en el desarrollo del proyecto estaba establecido en Alto. Como consecuencia, algunas tareas planificadas para las fases finales se han visto afectadas, como la integración del framework Caldera, que finalmente ha sido sustituido por scripts encargados de simular comportamientos sospechosos. A pesar de esto se han cumplido los objetivos principales del proyecto y se ha logrado implementar una infraestructura SIEM funcional, validando su funcionamiento mediante pruebas controladas.

El sprint 7 titulado “Instalación de CALDERA y Simulación de ataques” no pudo llevarse a cabo debido al retraso acumulado. En su lugar, se diseñó una nueva planificación que permitiera cumplir con los objetivos de validación del sistema implementado.

## SPRINT 7 (modificado): Simulación de ataques mediante scripts personalizados

| **Historia 12: Desarrollo de scripts para la simulación de comportamientos maliciosos** | |
|---|---|
| **Descripción** | Desarrollar 3 scripts con los que poner a prueba las reglas de detección creadas. |
| **Criterios de aceptación** | Los 3 scripts creados funcionan adecuadamente. |
| **Tareas** | Desarrollar un script que simule un ataque de fuerza bruta en el equipo ELAM.<br>Desarrollar un script que simule un escaneo de puertos desde ELAM hacia Server-AD.<br>Desarrollar un script que simule peticiones desde Server-AD hacia una IP externa desconocida. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 4 Horas |
| **Dependencias** | Depende de la Historia 8 |

| **Historia 13: Simulación de ataques con los scripts creados** | |
|---|---|
| **Descripción** | Ejecutar los scripts desarrollados para simular ataques y verificar que la infraestructura detecta correctamente los comportamientos maliciosos. |
| **Criterios de aceptación** | La infraestructura detecta los ataques simulados por los scripts y las reglas creadas generan alertas que se pueden ver desde los dashboards creados en Kibana. |
| **Tareas** | Ejecutar los scripts creados para la simulación de ataques y verificar en Kibana que la infraestructura implementada detecta los ataques y genera alertas. |
| **Prioridad** | Alta |
| **Estimación de esfuerzo** | 10 Horas |
| **Dependencias** | Depende de la Historia 12 |

# Bibliografía


[1] Infobae. *Récord histórico de ciberataques en todo el mundo y la pérdida de billones de dólares de las empresas en 2025*. 2024. URL: https://www.infobae.com/tecno/2024/11/22/record-historico-de-ciberataques-en-todo-el-mundo-y-las-perdida-de-billones-de-dolares-de-las-empresas-en-2025/ (visitado el 01-03-2025).

[2] Amazon Web Services. *¿Qué es Scrum?* 2025. URL: https://aws.amazon.com/es/what-is/scrum/ (visitado el 01-03-2025).

[3] Atlassian. *Scrum*. 2025. URL: https://www.atlassian.com/es/agile/scrum (visitado el 01-03-2025).

[4] Pirani Risk. *¿Qué es la ciberseguridad, cómo funciona y cuál es su importancia?* 2025. URL: https://www.piranirisk.com/es/academia/especiales/ciberseguridad-que-es-como-funciona-y-su-importancia?utm_source=chatgpt.com (visitado el 08-03-2025).

[5] Amazon Web Services. *¿Qué es la ciberseguridad?* 2025. URL: https://aws.amazon.com/es/what-is/cybersecurity/#:~:text=La%20ciberseguridad%20es%20la%20pr%C3%A1ctica,cliente%20y%20cumplir%20la%20normativa. (visitado el 08-03-2025).

[6] Age2. *Ciberseguridad: qué es y cuál es su importancia*. 2025. URL: https://www.age2.es/noticias/ciberseguridad-que-es-y-cual-es-su-importancia/ (visitado el 08-03-2025).

[7] Tranxfer. *Equipos de ciberseguridad: Red Team, Blue Team y Purple Team*. 2025. URL: https://www.tranxfer.com/equipos-ciberseguridad-red-team-blue-team-y-purple-team/ (visitado el 08-03-2025).

[8] Microsoft. *¿Qué es SIEM?* 2025. URL: https://www.microsoft.com/es-es/security/business/security-101/what-is-siem (visitado el 08-03-2025).

[9] Fortinet. *¿Qué es SIEM?* 2025. URL: https://www.fortinet.com/resources/cyberglossary/what-is-siem (visitado el 08-03-2025).

[10] Kaboom Eventos. *Security Event Management*. 2025. URL: https://kaboomeventos.com.ar/security-event-management/ (visitado el 08-03-2025).

[11] Anomali. *La evolución y el futuro de SIEM*. 2025. URL: https://www.anomali.com/es/resources/la-evolucion-y-el-futuro-de-siem (visitado el 08-03-2025).

[12] Stellar Cyber. *¿Qué es SIEM?* 2025. URL: https://stellarcyber.ai/es/learn/what-is-siem/ (visitado el 08-03-2025).

[13] Fortinet. *Firewall*. 2025. URL: https://www.fortinet.com/resources/cyberglossary/firewall (visitado el 08-03-2025).

[14] IBM. *Intrusion Detection System*. 2025. URL: https://www.ibm.com/es-es/topics/intrusion-detection-system (visitado el 08-03-2025).

[15] Versa Networks. *IDS vs IPS*. 2025. URL: https://versa-networks.com/es/sd-wan/ids-ips/ (visitado el 08-03-2025).

[16] Check Point. *¿Qué es Endpoint Detection and Response?* 2025. URL: https://www.checkpoint.com/es/cyber-hub/threat-prevention/what-is-endpoint-detection-and-response/ (visitado el 08-03-2025).
[17] Fortinet. *Herramientas de ciberseguridad para PYMEs*. 2025. URL: https://www.fortinet.com/lat/resources/cyberglossary/smb-cybersecurity-tools (visitado el 08-03-2025).
[18] S2 Grupo. *Blue Team en ciberseguridad: definición, funciones y herramientas*. 2025. URL: https://s2grupo.es/blue-team-en-ciberseguridad-definicion-funciones-y-herramientas/ (visitado el 08-03-2025).
[19] Fortinet. *Tipos de ciberataques*. 2025. URL: https://www.fortinet.com/lat/resources/cyberglossary/types-of-cyber-attacks (visitado el 08-03-2025).
[20] Check Point. *Tipos de ciberataques*. 2025. URL: https://www.checkpoint.com/es/cyber-hub/cyber-security/what-is-cyber-attack/types-of-cyber-attacks/ (visitado el 08-03-2025).
[21] Akamai. *¿Qué es un ataque DDoS?* 2025. URL: https://www.akamai.com/es/glossary/what-is-ddos (visitado el 08-03-2025).
[22] Check Point. *Ransomware*. 2025. URL: https://www.checkpoint.com/es/cyber-hub/threat-prevention/ransomware/ (visitado el 08-03-2025).
[23] Check Point. *¿Qué es un escaneo de puertos?* 2025. URL: https://www.checkpoint.com/es/cyber-hub/network-security/what-is-a-port-scan/ (visitado el 08-03-2025).
[24] Cloudflare. *Lateral Movement*. 2025. URL: https://www.cloudflare.com/es-es/learning/security/glossary/what-is-lateral-movement/ (visitado el 08-03-2025).
[25] Elastic. *Una introducción práctica a Logstash*. 2024. URL: https://www.elastic.co/es/blog/a-practical-introduction-to-logstash (visitado el 14-12-2024).
[26] Fluentd. *Architecture*. 2025. URL: https://www.fluentd.org/architecture (visitado el 17-05-2025).
[27] Splunk. *About the universal forwarder*. 2025. URL: https://help.splunk.com/en/splunk-enterprise/forward-and-process-data/universal-forwarder-manual/9.4/about-the-universal-forwarder/about-the-universal-forwarder (visitado el 17-05-2025).
[28] Formadores IT. *Qué es Apache Kafka y para qué sirve*. 2024. URL: https://formadoresit.es/que-es-apache-kafka-para-que-sirve/ (visitado el 14-12-2024).
[29] Arsys. *RabbitMQ: qué es y para qué sirve*. 2025. URL: https://www.arsys.es/blog/rabbitmq-mensajeria (visitado el 17-05-2025).
[30] Microsoft Azure. *Azure Event Hubs*. 2025. URL: https://azure.microsoft.com/es-es/products/event-hubs (visitado el 17-05-2025).
[31] Elastic. *¿Qué es Elasticsearch?* 2024. URL: https://www.elastic.co/guide/en/elasticsearch/reference/8.18/elasticsearch-intro-what-is-es.html (visitado el 14-12-2024).
[32] Splunk. *About Splunk Enterprise*. 2025. URL: https://docs.splunk.com/Documentation/Splunk/9.4.2/Overview/AboutSplunkEnterprise (visitado el 18-05-2025).
[33] MongoDB. *MongoDB is fantastic for logging*. 2025. URL: https://www.mongodb.com/blog/post/mongodb-is-fantastic-for-logging#:~:text=I%20would%20encourage%20everyone%20to,inserts%20can%20be%20done%20asynchronously. (visitado el 18-05-2025).
[34] Elastic. *Kibana*. 2024. URL: https://www.elastic.co/kibana (visitado el 14-12-2024).
[35] MetricFire. *What is Grafana?* 2025. URL: https://medium.com/@MetricFire/what-is-grafana-8de44d241765 (visitado el 18-05-2025).
[36] Splunk. *About dashboards*. 2025. URL: https://docs.splunk.com/Documentation/Splunk/9.4.2/SearchTutorial/Aboutdashboards (visitado el 18-05-2025).
[37] Fortinet. *Snort*. 2025. URL: https://www.fortinet.com/lat/resources/cyberglossary/snort (visitado el 18-05-2025).

[38] KeepCoding. *¿Qué es Suricata en ciberseguridad?* 2025. URL: https://keepcoding.io/blog/que-es-suricata-en-ciberseguridad/ (visitado el 18-05-2025).
[39] KeepCoding. *¿Qué es Zeek?* 2024. URL: https://keepcoding.io/blog/que-es-zeek/ (visitado el 21-12-2024).
[40] Tolu Michael. *Snort vs Suricata vs Zeek*. 2024. URL: https://tolumichael.com/snort-vs-suricata-vs-zeek/ (visitado el 21-12-2024).
[41] Zeek Project. *conn.zeek — Zeek Documentation*. 2024. URL: https://docs.zeek.org/en/current/scripts/base/protocols/conn/main.zeek.html (visitado el 21-12-2024).
[42] Logit.io. *Integración con Winlogbeat*. 2024. URL: https://logit.io/docs/integrations/winlogbeat/ (visitado el 22-12-2024).
[43] Elastic. *Configuración de Winlogbeat*. 2024. URL: https://www.elastic.co/docs/reference/beats/winlogbeat/configuration-winlogbeat-options (visitado el 22-12-2024).
[44] STR Sistemas. *Centralizado de eventos de Windows con Nxlog*. 2025. URL: https://www.strsistemas.com/blog/centralizado-de-eventos-de-windows-con-nxlog (visitado el 18-05-2025).
[45] Wazuh. *Instalación del agente Wazuh en Windows*. 2025. URL: https://documentation.wazuh.com/current/installation-guide/wazuh-agent/wazuh-agent-package-windows.html (visitado el 18-05-2025).
[46] Oracle. *VirtualBox Manual - Chapter 8: USB Support*. 2024. URL: https://www.virtualbox.org/manual/ch08.html (visitado el 07-12-2024).
[47] The Linux Foundation. *How to use the Netplan network configuration tool on Linux*. 2024. URL: https://www.linux.com/topic/distributions/how-use-netplan-network-configuration-tool-linux/ (visitado el 08-12-2024).
[48] Linux.com. *How to Use the Netplan Network Configuration Tool on Linux*. 2024. URL: https://www.linux.com/topic/distributions/how-use-netplan-network-configuration-tool-linux/ (visitado el 08-12-2024).
[49] Microsoft Learn. *New-NetIPAddress (PowerShell)*. 2024. URL: https://learn.microsoft.com/en-us/powershell/module/nettcpip/new-netipaddress?view=windowsserver2025-ps (visitado el 08-12-2024).
[50] Microsoft Learn. *Set-DnsClientServerAddress (PowerShell)*. 2024. URL: https://learn.microsoft.com/en-us/powershell/module/dnsclient/set-dnsclientserveraddress?view=windowsserver2025-ps (visitado el 08-12-2024).
[51] Microsoft Learn. *Install Active Directory Domain Services*. 2024. URL: https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/deploy/install-active-directory-domain-services--level-100- (visitado el 08-12-2024).
[52] Microsoft Learn. *New-ADUser (PowerShell)*. 2024. URL: https://learn.microsoft.com/en-us/powershell/module/activedirectory/new-aduser?view=windowsserver2025-ps (visitado el 08-12-2024).
[53] Gabriel Tanner. *Docker Compose Guide*. 2025. URL: https://gabrieltanner.org/blog/docker-compose/ (visitado el 01-03-2025).
[54] Docker. *Docker Network Drivers*. 2025. URL: https://docs.docker.com/engine/network/drivers/ (visitado el 01-03-2025).
[55] Docker. *Dockerfile reference - HEALTHCHECK*. 2025. URL: https://docs.docker.com/reference/dockerfile/#healthcheck (visitado el 01-03-2025).