Implementación de Security Data Lake con Splunk

Creación de reglas de correlación y modelos para la detección avanzada de amenazas

Tomás García Hidalgo

Ingeniería Informática
Seguridad Informática

Nombre Tutor/a de TF
Jorge Miguel Moneo
Profesor/a responsable de la asignatura
Helena Rifá Pous

Fecha Entrega
09/01/2023

Universitat Oberta de Catalunya

Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-SinObraDerivada 3.0 España de Creative Commons

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Implementación de entorno Security Data Lake con Splunk*
Nombre del autor:	*Tomás García Hidalgo*
Nombre del consultor/a:	*Jorge Miguel Moneo*
Nombre del PRA:	*Helena Rifá Pous*
Fecha de entrega:	*01/2023*
Titulación o programa:	Grado en Ingeniería Informática
Área del Trabajo Final:	*Seguridad Informática*
Idioma del trabajo:	*Castellano*
Palabras clave	*Anomaly detection, SIEM, Security Data Lake*

Resumen del Trabajo

La finalidad de este trabajo consiste en exponer las problemáticas y limitaciones actuales de los sistemas de recolección y análisis de eventos de seguridad (SIEM), y proponer una solución basada en el diseño de un sistema security data lake (SDL) como una forma de superar estas limitaciones.

El contexto de aplicación de este trabajo es el de las organizaciones que necesitan una plataforma para detectar tempranamente amenazas y eventos maliciosos en sus sistemas y dispositivos de red. Esto es especialmente importante debido al aumento exponencial de la cantidad de sistemas y dispositivos de red, la digitalización de procesos y la necesidad de cumplir con ciertas normativas de cumplimiento.

La metodología utilizada en este trabajo ha consistido en realizar una revisión bibliográfica sobre los principios de los sistemas de manejo de grandes cantidades de información, especialmente los SDL, y en la propuesta de un sistema SDL utilizando Splunk como tecnología base. Se han detallado los beneficios que ofrece Splunk para el desarrollo de un sistema SDL distribuido y capaz de gestionar grandes cantidades de datos, y se ha explicado cómo se pueden implementar tanto reglas de correlación como modelos de detección de anomalías utilizando técnicas de machine learning.

En cuanto a los resultados, se ha llevado a cabo una comparativa entre las diferentes técnicas de detección de patrones maliciosos a través de los datos

recogidos por el sistema, resaltando la flexibilidad de los modelos de detección de amenazas frente las reglas de correlación.

Abstract

The purpose of this work is to expose the current problems and limitations of security event collection and analysis systems (SIEM), and to propose a solution based on the design of a security data lake (SDL) system as a way to overcome these limitations.

The application context of this work is organizations that need a platform for early detection of threats and malicious events in their systems and network devices. This is especially important due to the exponential increase in the number of network systems and devices, the digitization of processes and the need to meet certain compliance regulations.

The methodology used in this work has consisted of a literature review on the principles of large data management systems, especially SDLs, and the proposal of an SDL system using Splunk as the base technology. The benefits offered by Splunk for the development of a distributed SDL system capable of managing large amounts of data have been detailed, and it has been explained how both correlation rules and anomaly detection models can be implemented using machine learning techniques.

As for the results, a comparison has been made between the different techniques for detecting malicious patterns through the data collected by the system, highlighting the flexibility of threat detection models versus correlation rules.

Índice

Lista de figuras

1. Introducción

1.1 Contexto y justificación del Trabajo

La implementación de sistemas de recolección y análisis de eventos que permitan una temprana detección de amenazas en las organizaciones es un aspecto clave en la postura de seguridad de las empresas que en el transcurso de los últimos años se ha tornado en una tarea de dificultad elevada, esto en parte es debido al aumento exponencial de la cantidad de sistemas, dispositivos de red, eventos de aplicaciones y en general la digitalización de la mayoría de las empresas [1].

La cantidad de eventos generados por estos sistemas se encuentran en constante aumento debido a digitalización de procesos otora analógicos y la adopción de nuevas tecnologías en los procedimientos de negocio, este hecho sumado la adhesión por parte de las organizaciones a diversas normativas de cumplimiento como PCI DSS [2] en el caso del comercio electrónico o GDPR que obligan al almacenamiento de logs durante largos periodos de tiempo, hacen imprescindible el dimensionamiento y la elección de una plataforma SIEM que habilite no solamente la ingesta de eventos y la aplicación de reglas de correlación si no que permita escalar horizontalmente a soluciones SDL [3].

En lo referente a los métodos de detección, en la mayor parte de los sistemas SIEM existen dos aproximaciones para llevar a cabo el descubrimiento de anomalías de seguridad o eventos maliciosos, la primera es mediate las reglas de correlación que destacan por ser una agrupación de uno o varios operadores lógicos sobre un conjunto de eventos siendo esta la aproximación más habitual, la figura uno muestra el ejemplo de la lógica inherente a una regla de correlación que detectaría el inicio de sesión de un administrador [4], el otro sistema de detección es mediante modelos de detección de anomalías en el comportamiento basados muchos de ellos en algoritmos de machine learning, este tipo de sistema de detección se hace imprescindible para detectar amenazas no conocidas o comportamientos extraños anómalos en las redes a monitorizar [5].

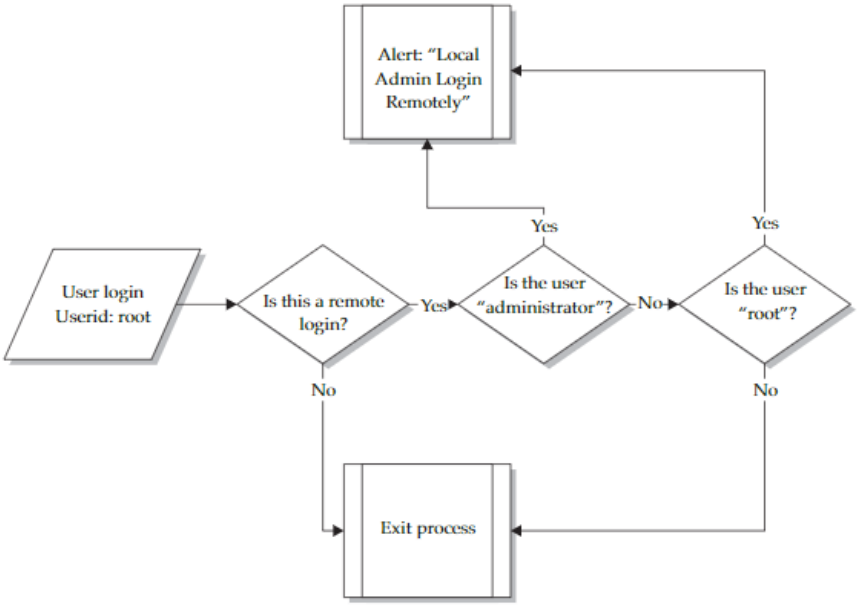


Figura 1: Lógica de detección de inicio de sesión por un administrador [4]

Uno de los principales desafíos a los que se deben de enfrentar los sistemas de monitorización es a una transición en donde se implementen métodos de análisis del comportamiento así como técnicas basadas en detección de anomalías [6], ya que las reglas de correlación como método para la detección de amenazas están perdiendo protagonismo debido a que esta aproximación requiere de una elevada interacción del componente humano, así como también conocimiento del patrón de ataque que se quiere detectar haciendo con ello que las amenazas desconocidas sean indetectables, este escenario en constante cambio y aumento en lo que cantidad de eventos y tipología se refiere hace que sea inviable el coste que supone para una organización en lo relativo a componentes humanos la creación constante de lógica para la detección de amenazas personalizada a cada tipo de evento o fuente recibida.

Igualmente, el aumento de la cantidad de eventos producidos en cualquier entorno real requiere de una aproximación distribuida para el manejo, análisis y correlación de los eventos de seguridad, así como la aplicación de técnicas de automatización en el tratamiento de las alertas generadas, es por ello por lo que cada vez más empresas se ven en la necesidad de implementar sistemas más aproximados a lo que serían los o data lakes utilizados en ciencia de datos separando con ello la capa de almacenamiento de la capa de procesamiento como sucede en algunas soluciones SIEM establecidas en la nube [7] permitiendo con ello poder agregar información contextual de múltiples sistemas sin apenas limitación.

Toda esta problemática inherente a los sistemas SIEM provoca que el grado de efectividad de este tipo de sistemas centralizados no alcance las cotas deseadas debido a no poder integrar toda la información proveniente de todos los sistemas de la empresa, así mismo las reglas de correlación utilizadas y el alto número de falsos positivos que estas generan crean una fatiga en los analistas de seguridad que se traduce un bajo aprovechamiento de los sistemas SIEM actuales, involucrando la automatización en el flujo del tratamiento de la alerta

se podrían reducir drásticamente los tiempos de respuesta y los falsos positivos mediante las plataformas SOAR (*Security Orchestration and Automation Response*) [8].

El objeto de este trabajo es abordar estas problemáticas anteriormente expuestas, e implementar una aproximación a un sistema de detección de eventos avanzados en una plataforma distribuida donde diferenciemos la capa de almacenamiento de la capa de procesamiento utilizando para ello el producto Splunk.

La herramienta Splunk provee una plataforma que permite realizar la ingesta de datos, así como su representación gráfica en la interfaz web disponiendo de todo ello en un único servidor, predeterminadamente Splunk en su configuración básica no se conforma para establecerse como una plataforma de datos distribuidos por lo que es necesario configurar todos los elementos de manera de cada uno ocupe un rol diferenciado en el flujo de ingesta y análisis de datos permitiendo con ello la creación de una arquitectura Data Lake, de igual manera es necesaria la configuración de los objetos de conocimiento y el mapeado de los datos a los mismos con motivo de convertirlos en datos estructurados y poder elaborar un análisis sobre estos.

Este trabajo aspira a diseñar e implementar una arquitectura distribuida de ingesta de grandes cantidades de eventos de seguridad aprovechando las capacidades de almacenamiento basado en índices que ofrece la herramienta Splunk, la creación y configuración de los objetos de conocimiento necesarios para óptimo análisis de las métricas de seguridad, así como también toda la lógica y mecanismos de análisis inherentes a ellos sobre esta tecnología. Análogamente se pretende configurar y replicar los elementos necesarios de una red empresarial

Igualmente se pretende abordar el desarrollo de varios modelos para la detección de eventos almacenados en el Data Lake a través de algoritmos de machine learning y el tratamiento automatizado de estas alertas generadas intentando conseguir con ello la detección avanzada de amenazas no conocidas.

1.2 Objetivos del Trabajo

Los principales objetivos que se pretenden conseguir con este trabajo de final de grado se encuentran los siguientes:

Objetivos académicos de investigación:

- Exponer problemáticas relativas a la monitorización de seguridad en entornos empresariales debido a la alta densidad de eventos y las limitaciones de los sistemas SIEM actuales.
- Realizar una revisión del estado del arte en los diversos métodos de detección de amenazas, así como las plataformas utilizadas para su implementación.
- Realizar un estudio de las técnicas y productos empleados para la implementación de SDL,

- Implementar modelos de amenazas mediante algoritmos de machine learning.

Objetivos específicos de implementación:

- Implementación de una plataforma SDL que separe la capa de almacenamiento y la capa de procesamiento de reglas y analítica.
- Configuración de los diferentes elementos generadores de eventos comunes en una empresa, e integración de estos en nuestra plataforma de monitorización para comprobar beneficios modelos de detección de amenazas.
- Mejora en la detección de anomalías y ataques de las reglas de correlación implementando para ello técnicas de modelado de amenazas usando algoritmos de machine learning.
- Realización de pruebas de funcionamiento, así como simulación de ataque y posterior detección por nuestro sistema.

1.3 Impacto en sostenibilidad, ético-social y de diversidad

Para el desarrollo de este trabajo se han tenido en cuenta la alineación de las dimensiones de sostenibilidad, responsabilidad ético y social y diversidad con los objetivos de desarrollo sostenible de la ONU.

De igual manera se han considerado todas las aportaciones hechas en el campo objeto del trabajo sin tener en cuenta el género de la autora o autor velando por que no se refuercen sesgos o estereotipos masculinos en el trabajo de fin de grado.

1.3.1 Sostenibilidad

Se ha pretendido abordar la dimensión de sostenibilidad, alineando el desarrollo de esta con el objetivo número nueve de los objetivos de desarrollo sostenible de la Organización de Naciones Unidas [9].

La innovación es un motor de cambio que permite afrontar todos los desafíos a través del progreso económico y tecnológico, grandes problemas como la deforestación han sido atajados a través de la innovación como la creación de las memorias USB.

Este trabajo tiene un impacto positivo en la dimensión de sostenibilidad, ya que promueve la mejora de las capacidades tecnológicas de todos los sectores tanto industriales como empresariales pudiéndose aplicar los resultados y conclusiones tratadas en el mismo para el desarrollo de los procesos de detección de anomalías y aumento de la seguridad en las plataformas de monitorización tecnológica.

1.3.2 Comportamiento ético y responsabilidad social

El compromiso ético y social de este trabajo tiene una conexión directa con el objetivo de desarrollo sostenible número dieciséis de la ONU que apela a la paz, justicia e instituciones sólidas.

Una de las metas que propone este ODS es la consagración a todos los niveles de instituciones eficaces y transparentes que rindan cuentas, el resultado de nuestro trabajo tiene un impacto positivo ya que a través de la monitorización de los sistemas se puede desincentivar a la perpetración de ilícitos digitales.

Mediante estas acciones se promoverían comportamientos más éticos, así como el fomento de la transparencia en todas las instituciones ya que mediante las herramientas de monitorización y data lakes de seguridad se centralizarían todos los datos en una única plataforma pudiendo utilizarse este repositorio en aras de ofrecer más transparencia en los diferentes procesos de las instituciones.

1.3.3 Diversidad, género y derechos humanos

El resultado de este trabajo no tiene un impacto directo positivo, así como tampoco negativo sobre los aspectos de género o diversidad, dado que el resultado de este aborda un enfoque meramente técnico de las problemáticas encontradas en los sistemas de monitorización actuales y los métodos para abordar los retos que en el futuro supondrá la detección de eventos maliciosos en entornos de alta densidad de eventos.

No obstante, este trabajo encuentra una intersección con el ODS número diez, ya que el resultado de este trabajo puede ayudar a mejorar la reglamentación y vigilancia de las instituciones fortaleciendo la aplicación de esos reglamentos.

Como se ha mencionado en anteriores puntos, muchos marcos regulatorios obligan a la custodia y mantenimiento de los datos durante periodos largos de tiempo, este trabajo aborda la problemática que suscita el almacenamiento de grandes volúmenes de datos y su custodia en un entorno centralizado con lo que conseguimos generar un impacto positivo en lo referente a datos de seguridad de las tecnologías de cualquier organización pública o privada.

1.4 Enfoque y método seguido

El enfoque seguido para el desarrollo de este trabajo está basado en una aproximación de solución en dos fases una primera fase eminentemente teórica y la siguiente fase práctica, con la fase teórica se pretende mostrar las limitaciones que se vienen encontrando en los SIEM [10] en los últimos años debido al aumento exponencial de la cantidad de sistemas y por consiguiente eventos que deben ingestar y analizar estas plataformas, con la fase práctica se busca analizar los métodos de detección en la actualidad y la utilización de modelos para la detección de amenazas desconocidas utilizando algoritmos de machine learning.

Primero se pretende llevar a cabo un estudio teórico de las problemáticas expuestas, así como de las soluciones existentes en el mercado, tanto en lo referente a las actuales infraestructuras de monitorización existentes en muchas

empresas, así como en los métodos de detección usados actualmente para encontrar anomalías o acciones maliciosas en los eventos.

Una vez realizado este estudio se intentará implementar una solución efectiva que permita escogiendo un grupo de herramientas que consigan solventar la problemática expuesta en el estudio llevado a cabo reduciendo la tasa de falsos positivos, así como la problemática de escalabilidad de las plataformas SIEM actuales.

La metodología que se ha seguido es una aproximación al diseño sistemático de sistemas para ingeniería [11] que si bien diferentes autores representan con una visión diferenciada siempre consta de los siguientes elementos:

- Planteamiento del problema
- Investigación
- Requisitos de la solución
- Evaluación preliminar de solución mínima viable
- Desarrollo del prototipo
- Validación y pruebas sobre la solución

En base a esta metodología se han planificado los trabajos a realizar consecuentemente.

1.5 Planificación del Trabajo

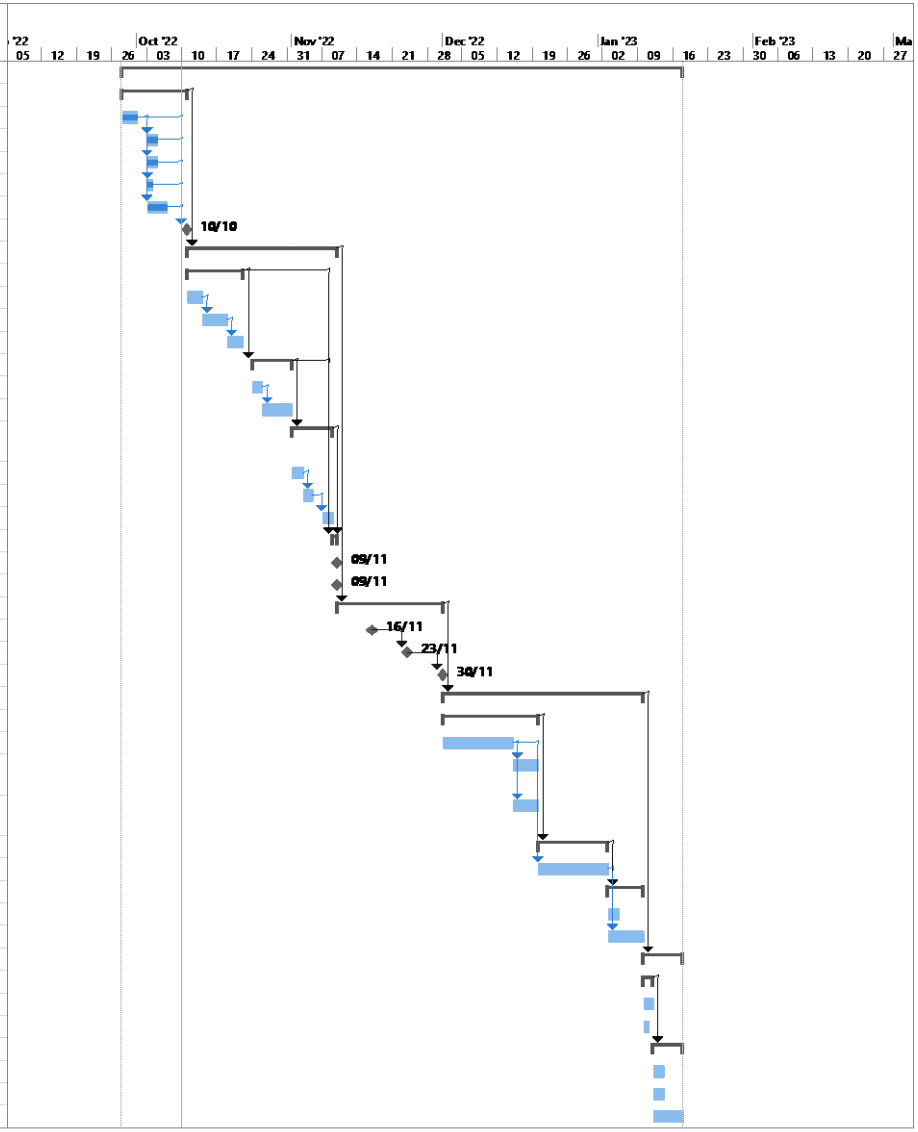
ID	Nombre de la tarea	Duración	Comienzo	Final	%
1	**Proyecto Final de Grado**	**80 days**	**Wed 28/09/22**	**Tue 17/01/23**	**15%**
2	**Fase 1 - Planificación**	**9 days**	**Wed 28/09/22**	**Mon 10/10/22**	**100%**
3	Definición de la problemática a resolver	3 days	**Wed 28/09/22**	Fri 30/09/22	100%
4	Enumeración de Objetivos	2 days	**Mon 03/10/22**	Tue 04/10/22	100%
5	Descripción Metodología	2 days	**Mon 03/10/22**	Tue 04/10/22	100%
6	Revisión de estado del arte	1 day	Mon 03/10/22	Mon 03/10/22	100%
7	Elaboración plan de trabajo	4 days	Mon 03/10/22	Thu 06/10/22	100%
8	Entrega PEC 1	2 days	Fri 07/10/22	Mon 10/10/22	100%
9	**Fase 2 - Estudio e Investigación**	**22 days**	**Tue 11/10/22**	**Wed 09/11/22**	**0%**
10	**Investigación plataformas Security Data Lake**	**9 days**	**Tue 11/10/22**	**Fri 21/10/22**	**0%**

11	Estudio Splunk	3 days	Tue 11/10/22	Thu 13/10/22	0%
12	Estudio Elastic Search	3 days	Fri 14/10/22	Tue 18/10/22	0%
13	Estudio HDFS	3 days	Wed 19/10/22	Fri 21/10/22	0%
14	**Investigación métodos de detección de amenazas**	**6 days**	**Mon 24/10/22**	**Mon 31/10/22**	**0%**
15	Estudio reglas de correlación	2 days	Mon 24/10/22	Tue 25/10/22	0%
16	Estudio modelos de detección a través de machine learning	4 days	Wed 26/10/22	Mon 31/10/22	0%
17	**Análisis de los elementos necesarios para entorno de laboratorio**	**6 days**	**Tue 01/11/22**	**Tue 08/11/22**	**0%**
18	Análisis elementos de computación necesarios	2 days	Tue 01/11/22	Wed 02/11/22	0%
19	Análisis elementos de seguridad de red necesarios	2 days	Thu 03/11/22	Fri 04/11/22	0%
20	Análisis elementos de generación de eventos necesarios	2 days	Mon 07/11/22	Tue 08/11/22	0%
21	**Elección de tecnologías en base a los requisitos**	**1 day**	**Wed 09/11/22**	**Wed 09/11/22**	**0%**
22	Elección de tecnologías candidatas SDL	1 day	Wed 09/11/22	Wed 09/11/22	0%
23	Elección de tecnologías candidatas entorno de laboratorio	1 day	Wed 09/11/22	Wed 09/11/22	0%
24	**Fase 3 - Diseño de la propuesta**	**15 days**	**Thu 10/11/22**	**Wed 30/11/22**	**0%**
25	Diseño plataforma de laboratorio	5 days	Thu 10/11/22	Wed 16/11/22	0%

26	Diseño plataforma de monitorización de eventos de seguridad	5 days	Thu 17/11/22	Wed 23/11/22	0%
27	Diseño de modelos para la detección de anomalías	5 days	Thu 24/11/22	Wed 30/11/22	0%
28	**Fase 4 - Implementación de la solución**	**28 days**	**Thu 01/12/22**	**Mon 09/01/23**	**0%**
29	**Implementación de la plataforma de laboratorio**	**13 days**	**Thu 01/12/22**	**Mon 19/12/22**	**0%**
30	Implementación de la infraestructura de virtualización	10 days	Thu 01/12/22	Wed 14/12/22	0%
31	Implementación de los diferentes elementos de computación en la plataforma de virtualización	3 days	Thu 15/12/22	Mon 19/12/22	0%
32	Implementación de los diferentes elementos de seguridad de red	3 days	Thu 15/12/22	Mon 19/12/22	0%
33	**Implementación de la plataforma SDL**	**10 days**	**Tue 20/12/22**	**Mon 02/01/23**	**0%**
34	Instalación y configuración de la plataforma SDL	10 days	Tue 20/12/22	Mon 02/01/23	0%
35	**Implementación reglas monitorización**	**5 days**	**Tue 03/01/23**	**Mon 09/01/23**	**0%**
36	Implementación y configuración reglas de correlación	2 days	Tue 03/01/23	Wed 04/01/23	0%
37	Implementación y configuración modelos de detección	5 days	Tue 03/01/23	Mon 09/01/23	0%

38	**Fase 5 - Pruebas de validación**	**6 days**	**Tue 10/01/23**	**Tue 17/01/23**	**0%**
39	**Validación entorno SDL**	**2 days**	**Tue 10/01/23**	**Wed 11/01/23**	**0%**
40	Subsanación errores en entorno SDL	2 days	Tue 10/01/23	Wed 11/01/23	0%
41	Validación final entorno SDL	1 day	Tue 10/01/23	Tue 10/01/23	0%
42	**Validación métodos de detección**	**4 days**	**Thu 12/01/23**	**Tue 17/01/23**	**0%**
43	Subsanación errores métodos de detección	2 days	Thu 12/01/23	Fri 13/01/23	0%
44	Validación final reglas de correlación	2 days	Thu 12/01/23	Fri 13/01/23	0%
45	Validación final modelos de detección	4 days	Thu 12/01/23	Tue 17/01/23	0%

ID	Task Name	Duration	Start	Finish
1	**Proyecto Final de Grado**	**80 days**	**Wed 28/09/22**	**Tue 17/01/23**
2	**Fase 1 - Planificación**	**9 days**	**Wed 28/09/22**	**Mon 10/10/22**
3	Definición de la problemática a resolver	3 days	**Wed 28/09/22**	Fri 30/09/22
4	Enumeración de Objetivos	2 days	**Mon 03/10/22**	Tue 04/10/22
5	Descripción Metodología	2 days	**Mon 03/10/22**	Tue 04/10/22
6	Revisión de estado del arte	1 day	Mon 03/10/22	Mon 03/10/22
7	Elaboración plan de trabajo	4 days	Mon 03/10/22	Thu 06/10/22
8	Entrega PEC 1	2 days	Fri 07/10/22	Mon 10/10/22
9	**Fase 2 - Estudio e Investigación**	**22 days**	**Tue 11/10/22**	**Wed 09/11/22**
10	**Investigación plataformas Security Data Lake**	**9 days**	**Tue 11/10/22**	**Fri 21/10/22**
11	Estudio Splunk	3 days	Tue 11/10/22	Thu 13/10/22
12	Estudio Elastic Search	3 days	Fri 14/10/22	Tue 18/10/22
13	Estudio HDFS	3 days	Wed 19/10/22	Fri 21/10/22
14	**Investigación métodos de detección de amenazas**	**6 days**	**Mon 24/10/22**	**Mon 31/10/22**
15	Estudio reglas de correlación	2 days	Mon 24/10/22	Tue 25/10/22
16	Estudio modelos de detección a través de machine learning	4 days	Wed 26/10/22	Mon 31/10/22
17	**Análisis de los elementos necesarios para entorno de laboratorio**	**6 days**	**Tue 01/11/22**	**Tue 08/11/22**
18	Análisis elementos de computación necesarios	2 days	Tue 01/11/22	Wed 02/11/22
19	Análisis elementos de seguridad de red necesarios	2 days	Thu 03/11/22	Fri 04/11/22
20	Análisis elementos de generación de eventos necesarios	2 days	Mon 07/11/22	Tue 08/11/22
21	**Elección de tecnologías en base a los requisitos**	**1 day**	**Wed 09/11/22**	**Wed 09/11/22**
22	Elección de tecnologías candidatas SDL	1 day	Wed 09/11/22	Wed 09/11/22
23	Elección de tecnologías candidatas entorno de laboratorio	1 day	Wed 09/11/22	Wed 09/11/22
24	**Fase 3 - Diseño de la propuesta**	**15 days**	**Thu 10/11/22**	**Wed 30/11/22**
25	Diseño plataforma de laboratorio	5 days	Thu 10/11/22	Wed 16/11/22
26	Diseño plataforma de monitorización de eventos de seguridad	5 days	Thu 17/11/22	Wed 23/11/22
27	Diseño de modelos para la detección de anomalías	5 days	Thu 24/11/22	Wed 30/11/22
28	**Fase 4 - Implementación de la solución**	**28 days**	**Thu 01/12/22**	**Mon 09/01/23**
29	**Implementación de la plataforma de laboratorio**	**13 days**	**Thu 01/12/22**	**Mon 19/12/22**
30	Implementación de la infraestructura de virtualización	10 days	Thu 01/12/22	Wed 14/12/22
31	Implementación de los diferentes elementos de computación en la plataforma de virtualización	3 days	Thu 15/12/22	Mon 19/12/22
32	Implementación de los diferentes elementos de seguridad de red	3 days	Thu 15/12/22	Mon 19/12/22
33	**Implementación de la plataforma SDL**	**10 days**	**Tue 20/12/22**	**Mon 02/01/23**
34	Instalación y configuración de la plataforma SDL	10 days	Tue 20/12/22	Mon 02/01/23
35	**Implementación reglas monitorización**	**5 days**	**Tue 03/01/23**	**Mon 09/01/23**
36	Implementación y configuración reglas de correlación	2 days	Tue 03/01/23	Wed 04/01/23
37	Implementación y configuración modelos de detección	5 days	Tue 03/01/23	Mon 09/01/23
38	**Fase 5 - Pruebas de validación**	**6 days**	**Tue 10/01/23**	**Tue 17/01/23**
39	**Validación entorno SDL**	**2 days**	**Tue 10/01/23**	**Wed 11/01/23**
40	Subsanación errores en entorno SDL	2 days	Tue 10/01/23	Wed 11/01/23
41	Validación final entorno SDL	1 day	Tue 10/01/23	Tue 10/01/23
42	**Validación métodos de detección**	**4 days**	**Thu 12/01/23**	**Tue 17/01/23**
43	Subsanación errores métodos de detección	2 days	Thu 12/01/23	Fri 13/01/23
44	Validación final reglas de correlación	2 days	Thu 12/01/23	Fri 13/01/23
45	Validación final modelos de detección	4 days	Thu 12/01/23	Tue 17/01/23



1.6 Estado del arte

En los últimos años las tecnologías empleadas en la detección de anomalías o ataques maliciosos se han visto sobrepasadas debido al aumento del número de eventos y tecnologías que es preciso recoger para detectar y responder eficientemente a los ataques que suceden [12].

Los sistemas SIEM tradicionales paulatinamente están pasando a un segundo plano dejando lugar a tecnologías que implementan técnicas de ciencia de datos para tratar la gran cantidad de eventos que es necesario analizar en tiempo real [13].

Esta tendencia ha hecho aparecer nuevos actores en el espacio del análisis de eventos de seguridad como es el caso de Splunk que siempre ha tenido un papel protagonista en el campo de la analítica de eventos en entornos de alta densidad de estos como puede ser monitorización de infraestructura IT.

Esta problemática también implica el desbordamiento de los analistas de seguridad, ya que debido a las aproximaciones de los SIEM tradicionales donde como método de detección se viene usando prominentemente las reglas de correlación estas causan muchos falsos positivos que obligan a los analistas a revisar constantemente.

En lo referente a técnicas de detección, actualmente se comienza a utilizar técnicas de análisis de comportamiento de los usuarios o modelos, este tipo de técnicas depende en gran medida de la habilidad de los analistas para definir los resultados de un comportamiento anormal.

Ejemplo de una regla de esta tipología podría ser la siguiente:

- Un usuario cambia de su cuenta normal a una cuenta de administrador y acto seguido realiza una trasferencia de datos excesivamente alta a sistemas fuera de la compañía.

Una vez expuesta la problemática a la que se enfrentan los SIEM se muestran algunas de las soluciones más conocidas de SDL que están sustituyendo en gran medida a los sistemas SIEM tradicionales:

1.6.1 Sistemas SDL

La base teórica sobre la cual se sustentan este tipo de tecnologías nace con el *big data* [14], esta nueva terminología define a los conjuntos grandes de datos que por su tamaño son imposibles de procesar individualmente por computadores como se venía haciendo hasta la fecha.

Es por esto por lo que la aproximación que se requiere para procesar estas grandes cantidades de datos pasa por arquitecturas distribuidas donde varios computadores puedan trabajar paralelamente en conjuntos de datos de un conjunto mayor.

Esta filosofía es la que define a los sistemas actuales de Security Data Lakes, ya que la mayoría de ellos implementan métodos para tratar los grandes conjuntos de eventos e información de manera desagregada, esto es separando las capas de almacenamiento, ingesta y procesamiento y permitiendo escalar de manera distribuida en función de la necesidad.

Un ejemplo claro de esta filosofía es Hadoop [15] que implementa un paradigma de procesamiento de datos caracterizado por estar distribuido entre todos sus nodos sumando la capacidad de procesamiento de todos ellos a la hora de realizar consultas sobre los datos del conjunto.

1.6.1.1 Azure Sentinel

La tecnología subyacente que utiliza Azure Sentinel como Data Lake es conocida como Azure Monitor [16] junto con Azure Log Analytics, estas dos tecnologías permiten a Azure Sentinel tanto el almacenamiento de una gran cantidad de eventos provenientes de diferentes fuentes de información tanto de eventos operacionales como de seguridad, así como su posterior consulta y análisis mediante el lenguaje Kusto Query Language.
A grandes rasgos Azure Monitor funciona como una base de datos columnar donde se almacenan todo tipo de datos en diferentes tablas que luego pueden ser consultados a través del servicio de Azure Log Analytics creando diferentes tipos de consultas utilizando para ello toda la lógica y elementos del lenguaje Kusto Query Language

```
SecurityEvent | where EventID == "4626" | summarize count() by Account | limit 10
```

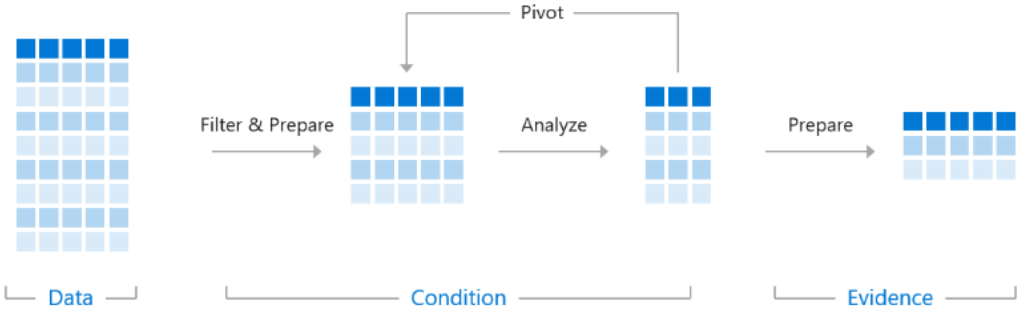


Figura 2: Estructura de una consulta Kusto Query Language [17]

1.6.1.2 Splunk

Splunk se conforma por una plataforma de ingesta y análisis de datos que permite almacenar la información en grupos de información indexada, una vez indexada esta información puede ser consultada a través del lenguaje *Search Language Processing*.

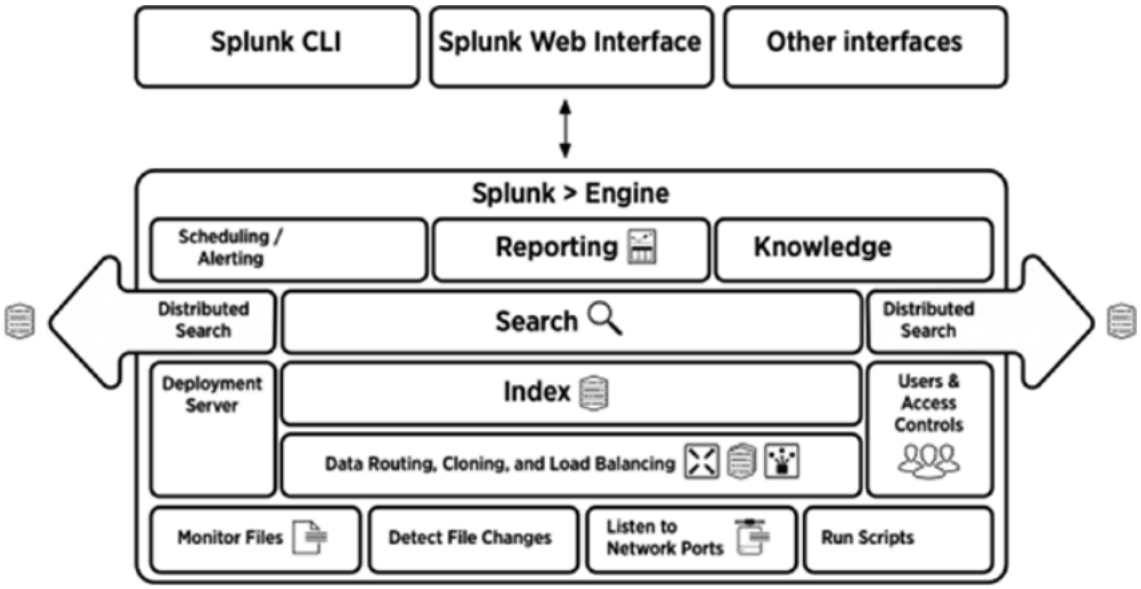


Figura 3: Arquitectura Splunk [18]

La arquitectura interna de Splunk permite tener todas sus funcionalidades distribuidas permitiendo con ello separar todas las capas que conforman el sistema, cabe destacar de esta solución la manera en la cual transforma Splunk los datos en eventos, este proceso lo realizan los servidores con rol de Indexer y para cada grupo de datos se encarga de comprimir los datos en bruto, crear índices apuntando a esos datos en bruto y ficheros de metadatos, estos índices son punteros de cada uno de los términos que aparecen y su lugar en los datos en bruto.

Esta implementación es la que hace de Splunk una gran solución ya que mediante el almacenamiento de estos índices junto con el grupo de datos permite realizar búsquedas extremadamente rápidas en grandes conjuntos de datos, ya que como cada índice se almacena para un grupo de datos en bruto al realizar la búsqueda la capacidad de proceso será repartida entre los diferentes elementos donde se encuentre la información.

Una de las grandes ventajas que se observa en la citada plataforma es la capacidad de desempeñar todos las capacidades ofrecidas en un único nodo con la posibilidad de configurar varios nodos para adoptar roles diferenciados en el caso de los despliegues distribuidos más complejos.

1.6.1.3 Exabeam Data Lake

Exabeam se conforma por varias tecnologías que permiten la ingesta de datos y posterior análisis de estos, la tecnología sobre la cual se cimenta Exabeam es el motor de búsqueda de Elastic Search donde se almacenan los datos indexados.

La principal diferencia de este sistema con otros es que los datos en la ingesta deben de ser modificados para adecuarlos al esquema del índice donde vayan a ser almacenados.

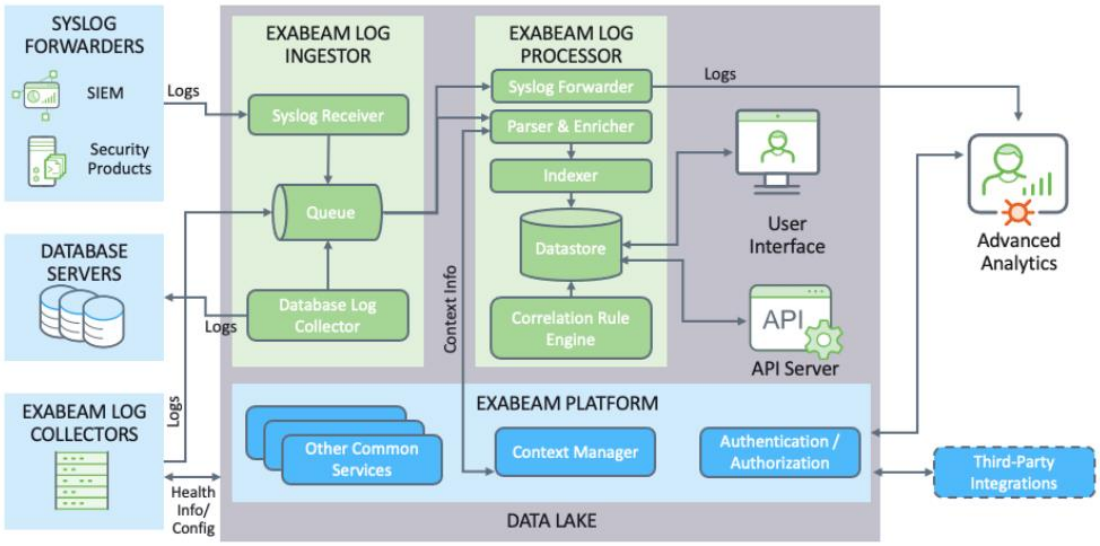


Figura 4: Arquitectura Exabeam Data Lake [19]

En lo referente a la detección, se realizan reglas de correlación a través de consultas Kibana Query Language directamente sobre el motor de búsqueda de Elastic Search que tiene embebida esta solución.

1.6.2 Reglas de detección de anomalías y patrones de ataque

En lo referente a las reglas de detección de anomalías cada fabricante ofrece su aproximación y entorno de desarrollo personalizado para ellas, ejemplos dispares de lo anteriormente mencionado son Azure Sentinel y Exabeam.

En Azure Sentinel existe una tipología de reglas conocida como alertas fusión [20] que identifican comportamientos anómalos mapeando estos con la cadena *cyber kill chain* sin embargo tanto la lógica como los modelos subyacentes que permiten la detección permanecen ocultos sin posibilidad de modificarlos o consultarlos.

El fabricante Exabeam sin embargo opta por otra aproximación, una vez implementada su herramienta, permite la modificación y consulta de las reglas por defecto configuradas en la plataforma para adecuarlas a nuestro entorno pudiendo observar los modelos subyacentes que originan las alertas.

1.7 Breve sumario de productos obtenidos

En la realización de este trabajo se han utilizado un amplio abanico de tecnologías con el objeto de simular una implementación de un sistema SDL, así como también los elementos generadores de eventos más comunes en una infraestructura empresarial.

Entre los productos obtenidos se pueden destacar un examen concienzudo de las mejoras que ofrecen los sistemas SDL respecto a otro a los sistemas de gestión y análisis de eventos tradicionales, así mismo se puede interpretar el trabajo como una guía práctica de implementación de un entorno SDL para la

detección y análisis de eventos de seguridad y la creación de modelos de detección mediante algoritmos de machine learning.

1.8 Breve descripción de los otros capítulos de la memoria

La distribución de este trabajo se ha estructurado en nueve capítulos.

En el primer capítulo se ha realizado una introducción donde se detalla los aspectos básicos del trabajo, así como son el contexto, justificación, impacto, planificación y estado del arte.

En el segundo capítulo se ha llevado a cabo una investigación sobre las plataformas y los diferentes métodos de detección de amenazas existentes en la actualidad.

En el tercer capítulo se diseña una propuesta de implementación en base al análisis realizado en el segundo capítulo que permita a través de la herramienta Splunk realizar la detección de amenazas avanzadas mediante técnicas y algoritmos de machine learning.

En el cuarto capítulo se realiza la implementación y evaluación de la solución implementada, realizando todos los test necesarios para comprobar el grado de certeza para la detección de determinadas amenazas de seguridad.

En el quinto capítulo se detallan las conclusiones extraídas del ejercicio de diseño e implementación de la herramienta, así como consideraciones a tener en cuenta en futuros trabajos.

En el sexto capítulo se detalla un glosario con toda la terminología usada en el presente trabajo

En el séptimo capítulo está reflejado todas las fuentes que han sido consultadas para la realización de este trabajo.

En el octavo capítulo se incluyen los anexos utilizados en el trabajo.

2. Investigación y análisis

2.1 Investigación de plataformas Security Data Lake

2.1.1 Splunk

Splunk es una plataforma de almacenamiento y visualización de datos que utiliza índices y no base de datos para almacenar y acceder a todos los datos que se almacenen en la plataforma.

El ciclo de vida de los datos en Splunk comienza por la fase de ingesta donde a través de diversos protocolos bien sean protocolos de red o lectura sobre ficheros en el mismo sistema operativo se recopilan los diferentes grupos de datos, en la siguiente fase dotaremos a los datos recibidos de una estructura

específica para cada tipología de eventos, realizando las diferentes operaciones sobre estos para normalizarlos (asignarles una hora), dividir los eventos en función de un patrón (salto de línea) o incluso enmascaramiento de algún campo.

La siguiente etapa consiste en el almacenamiento de estos datos en grupos de eventos indexados en el disco, existiendo diferentes tipos de grupos en función del tiempo de acceso a los diferentes grupos de información.

Por último, al final del ciclo se encuentra la fase de búsqueda donde se consultarán estos datos en base a las estructuras creadas de cada tipología de eventos.

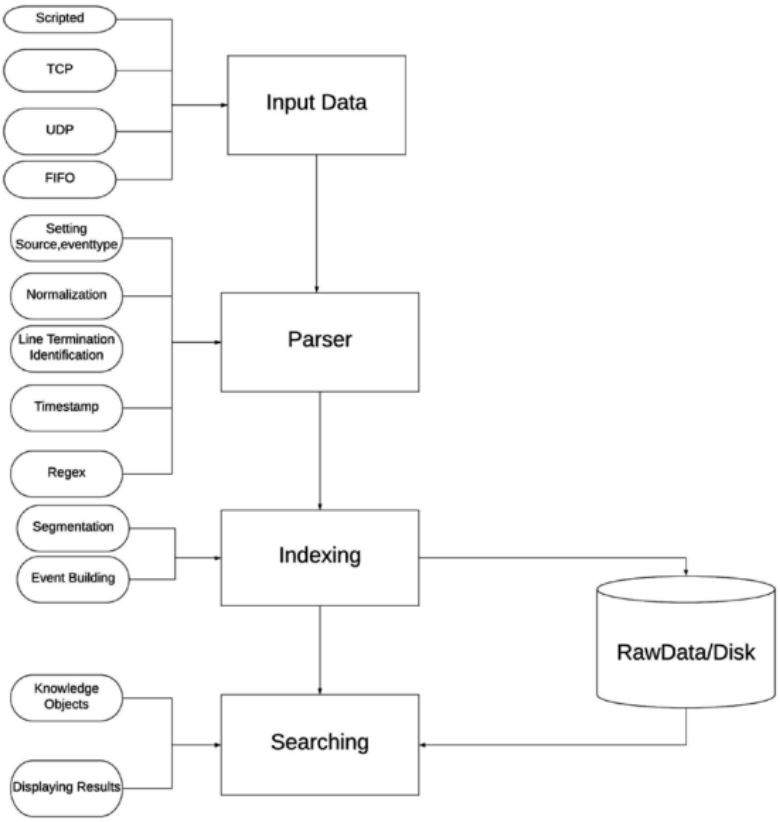


Figura 5: Ciclo de vida eventos en Splunk [18]

Cada servidor de Splunk puede desempeñar uno, varios o todos los roles necesarios para poder ejecutar las etapas del tratamiento de eventos de seguridad, los roles más destacados para los servidores de Splunk son los siguientes.

El rol de Forwarder, está encargado de la recolección e ingesta de todos los datos a través de diferentes métodos o protocolos, incluyendo entre los más destacados, syslog, tcp, udp, ficheros y registro de eventos de Windows.

El rol de Indexer, aplica las transformaciones y normalizaciones necesarias sobre los datos previamente a almacenarlos en los grupos de eventos definidos a tal efecto como índices.

El rol de Search Head, realiza todas las consultas sobre la información almacenad en los servidores cuyo rol es el de Indexer, este rol también es el encargado de la ejecución de todas las alertas tanto de recolección como de los modelos establecidos para la detección de anomalías.

2.1.2 Elastic Stack

Elastic Stack es un conjunto de tecnologías que basan su funcionalidad en la ingesta, almacenamiento y búsqueda sobre grandes cantidades de datos, los principales sistemas que componen Elastic Stack son los siguientes.

Logstash es un producto que se encarga de realizar la ingesta, transformación y normalización de todos los datos previamente a ser almacenados, entre sus capacidades se destaca el amplio abanico de integraciones con diferentes tecnologías.

Elastic Search es la tecnología principal del conjunto dado que su función principal es el almacenamiento de todo el conjunto de datos, así como su consulta, está basado en el proyecto de software libre Apache Lucene [21].

Kibana es la aplicación web que actúa como interfaz del conglomerado de tecnologías que componente el Elastic Stack actuando como punto de unión entre las mismas.

2.1.3 HDFS

Hadoop Distributed File System [15] es un sistema de ficheros distribuido que permite almacenar set masivos de datos tanto de logs normalizados como de logs sin normalizar.

Entre las ventajas que presenta este sistema de ficheros están la replicación de los datos y por tanto la tolerancia a fallos, así como la alta escalabilidad horizontal.

HDFS es una capa de abstracción que permitiría a cualquier aplicativo acceder a los datos en el lugar donde se encuentren almacenados, el funcionamiento de esta tecnología está basado en la división de la información en bloques de un mismo tamaño que son distribuidos a través de los nodos que conforman el clúster.

Uno de los principales beneficios que presenta HDFS a la hora de conformar un Data Lake, es MapReduce [22] que es un paradigma de procesamiento de datos que se caracteriza por la división en dos etapas cuyos subprocesos se ejecutan paralelamente consiguiendo con ello unos reducidos tiempo de acceso a los datos consultados.

La principal contrapartida de esta tecnología es la complejidad subyacente para la implementación de plataformas basadas en HDFS y su posterior mantenimiento, del mismo modo la implementación de consultas utilizando *map reduce* presenta complejidades que deben ser salvadas mediante otras capas

de abstracción como Hive para utilizar lenguajes similares a SQL en las consultas.

2.2 Investigación de métodos de detección de amenazas

2.2.1 Reglas de Correlación

Las reglas de correlación de manera general están formadas por expresiones lógicas que generan acciones o eventos específicos cada vez que un tipo de evento definido previamente ocurre, estas reglas de correlación en los sistemas SDL son llevadas a cabo por búsquedas mediante el lenguaje inherente a cada tecnología de patrones sobre diferentes conjuntos y tipologías de eventos que pueden dar lugar a una alerta de seguridad.

Un patrón de ataque está basado en una serie de pasos donde cada uno de ellos se basa en un paso anterior con la finalidad de llegar a un objetivo [23].

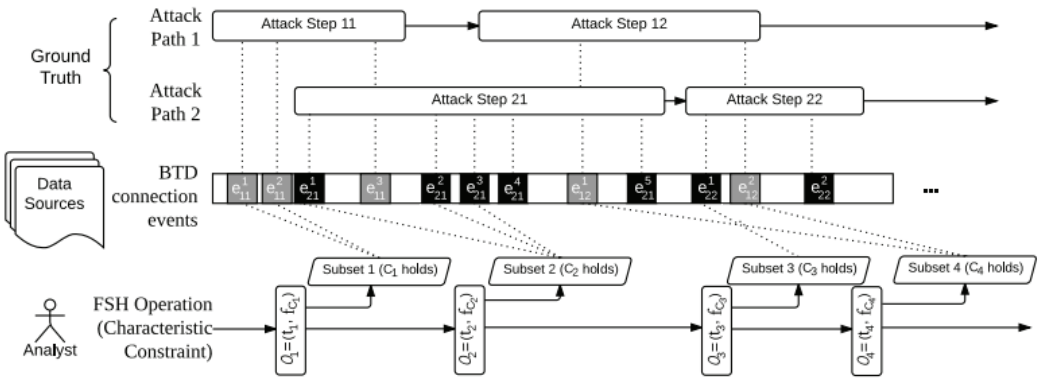


Figura 6: Regla de correlación sobre conjunto de eventos [23]

Una de las deficiencias de las reglas de correlación es que estas no evalúan el historial de los eventos objetivo, esto significa que este tipo de evaluaciones lógicas únicamente evalúa las condiciones para una determinada franja de tiempo.

Algunas implementaciones de este tipo de reglas de correlación permiten establecer reglas de correlación compuestas lo que permite generar alertas en el caso de que una o más reglas de correlación hayan sido activadas previamente, un ejemplo de esto sería una regla de correlación que se activara ante una sucesión de intentos de inicio de sesión fallidos.

En resumen, este tipo de reglas presenta claras ventajas en varios supuestos entre ellos la detección de amenazas ya conocidas, donde los patrones de eventos de estos ataques son conocidos y pueden ser trasladados a lógicas de detección por los analistas de seguridad.

2.2.2 Modelos de detección

Los modelos de detección al contrario que las reglas de correlación por norma general evalúan las desviaciones en el comportamiento identificando valores anormales y generando alertas en relación con estos comportamientos.

Para la creación de este tipo de reglas se usan algoritmos de Machine Learning que permite que los modelos aprendan sin la necesidad de intervención humana, el proceso de aprendizaje de las máquinas está dividido en cuatro fases [24], la primera de ellas es la fase de análisis donde se analizara el conjunto de datos de entrada para detectar patrones ocultos en los datos, la segunda fase los parámetros encontrados en la fase de análisis son usados para crear los modelos, la tercera fase consiste en la comprobación del rendimiento del modelo a través de matrices de confusión y otros métodos, por último en la cuarta fase se procede a poner en producción el modelo y realizar la ingesta de datos reales.

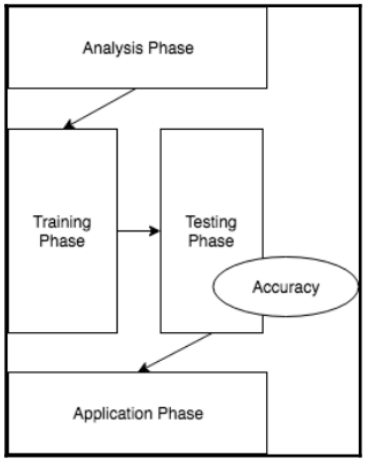


Figura 7: Fases proceso Machine Learning [24]

Entre los diferentes casos de uso que podríamos encontrar para este tipo de detección se encontraría, la detección de anomalías en red, predicción de ataques DDoS [25] y detección de correos maliciosos.

Para la aplicación de este tipo de modelos de detección de anomalías es necesario hacer una revisión de los sistemas de machine learning más utilizados en el campo de la ciberseguridad.

2.2.2.1 Algoritmos de aprendizaje supervisado

Un algoritmo se clasifica como de aprendizaje supervisado cuando el conjunto de datos utilizados para clasificar o predecir es conocido y consulta regularmente al usuario aprendiendo en el camino, este tipo de sistemas normalmente utilizan datos etiquetados para aprender y luego tomar decisiones sobre el conjunto de datos resultante.

Un ejemplo donde se aplican este tipo de algoritmos es en la detección de correos Spam, mediante esta técnica podemos distinguir correos legítimos de los considerados como Spam, el acto de enviar un correo a Spam ayuda a mejorar el algoritmo ya que utilizará ese movimiento para predecir futuros correos de Spam en base a los correos movidos con anterioridad.

2.2.2.2 Algoritmos de aprendizaje no supervisado

Esta clasificación de los algoritmos se refiere a cuando el conjunto de datos iniciales para el entrenamiento del modelo no está etiquetado, este tipo de algoritmos son más complejos ya que los sistemas aprenden sin ninguna intervención.

Un ejemplo práctico de algoritmo de aprendizaje no supervisado se puede ver en el análisis del comportamiento de usuario, en este ejemplo se utilizan varios parámetros acerca del comportamiento como podría ser por ejemplo la cantidad de inicios de sesión que realiza al día, si usa cuentas con permisos administrativos o no y utiliza esta información para clasificar a los usuarios en grupos que comparten patrones similares, pudiendo monitorizar si existe alguna desviación en el comportamiento normal de algún usuario concreto respecto a su grupo.

2.3 Análisis de los elementos necesarios para entorno de laboratorio

Para la puesta en práctica de este trabajo será necesario simular un entorno que provea de los elementos generadores de eventos básicos en cualquier red empresarial de trabajo, estos elementos consisten en elementos de red, elementos de autenticación y directorio que permitan analizar comportamientos anómalos en este dominio y generar el modelado de diferentes reglas para su detección.

Para la implementación de esta propuesta se ha optado por utilizar un hypervisor de cara a utilizar óptimamente los recursos computacionales disponibles.

Los medios utilizados han sido un servidor Fujitsu TX1320 M3 Intel Xeon E3-1220v6 con 56GB de memoria RAM con VMWare ESXI instalado donde se han provisionado las diferentes máquinas virtuales necesarias para la realización de este trabajo.

VMWare ESXI

Se ha optado por la elección de este hypervisor de tipo uno en su versión 7.0 U3 para la implementación de todas las máquinas virtuales necesarias de cara a generar un entorno base simulado sobre el cual poder realizar nuestra propuesta.

Security Data Lake

En el contexto de este trabajo se ha optado por utilizar Splunk como tecnología base para la implementación de la solución central SDL para la recopilación de eventos de seguridad y posterior consulta de los datos.

PFSense

Para la parte de red se ha optado por la elección de este dispositivo cortafuegos en su versión 2.6.0 debido a las capacidades que ofrece permitiendo generar

eventos de trazabilidad IDS mediante la instalación de paquetes adicionales como son SNORT y ZEEK dentro del propio cortafuegos.

Directorio Activo

Para la generación de eventos de directorio y de usuario se ha optado por la utilización de tres máquinas virtuales, la primera de ellas es un controlador de dominio donde se ha creado un directorio activo como elemento generador de todos estos eventos.

Así mismo se ha establecido como necesarios para la propuesta el provisionamiento de un cliente y un servidor adicional para la recolección de diferentes tipologías de eventos de cara a generar eventos maliciosos y probar diferentes tipologías de métodos de detección.

3. Diseño de la propuesta

3.1 Arquitectura general de la solución

La arquitectura general de la solución propuesta está compuesta de un grupo de servidores y elementos de red que serán los encargados de generar eventos y enviarlos al sistema SDL.

Se han utilizado una arquitectura de tres máquinas virtuales teniendo la posibilidad de escalar horizontalmente ya que todos los roles implementados tienen capacidad de escalado 1...N.

En el siguiente diagrama se puede apreciar el flujo que seguirán los datos en nuestra arquitectura de manera general, en primer lugar, el servidor con rol de Forwarder se encargará de recibir todos los eventos vía los diferentes protocolos utilizando syslog de manera predeterminada.

En el sistema SDL se almacenarán los eventos en el servidor con el rol de Indexer, estos eventos almacenados serán consultados a través del servidor con el rol de Search Head tanto para las reglas de correlación como para la ejecución de modelos de análisis basados en algoritmos de Machine Learning.

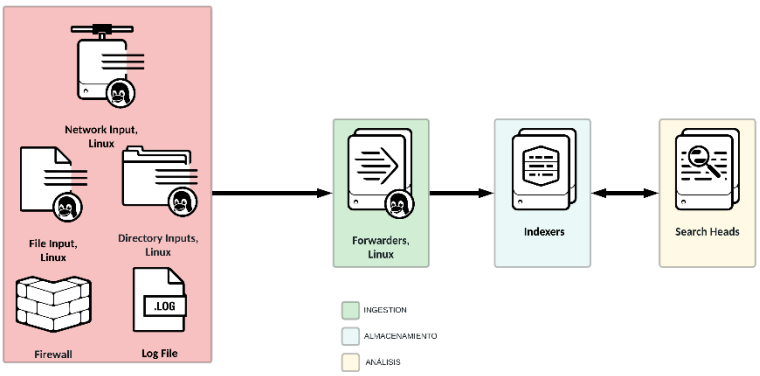


Figura 8: Propuesta de arquitectura SDL

En el servidor con el rol de Search Head se crearán las reglas, así como los modelos para la detección de anomalías y eventos de seguridad en el sistema SDL, estas reglas de correlación y los modelos se realizarán mediante consultas utilizando el lenguaje SPL a los conjuntos de datos almacenados en los servidores con el rol Indexer.

3.2 Arquitectura de detección

Entre los objetivos del presente trabajo se pretende abordar la mejora de los modelos de detección basados en reglas de correlación utilizando para ellos modelos de detección basados en algoritmos de machine learning.

Para acometer este objetivo es necesario la generación de grandes conjuntos de datos que permitan establecer patrones entre los mismos, para ello la siguiente arquitectura se ha establecido aprovechando las ventajas que ofrece el cortafuegos PFSense y el hypervisor ESXI, en este caso PFSense actúa como puerta de enlace para todos los elementos de nuestra red.

Esta red virtual que interconecta todos los elementos se genera mediante un conmutador virtual configurado en modo promiscuo de tal manera que a todos los efectos actúa como un concentrador permitiendo que configurando la interfaz interna de red en modo promiscuo de PFSENSE recibamos todos los paquetes generados en la red estableciendo con ello un buen punto de partida en lo que a capacidades de detección se refiere.

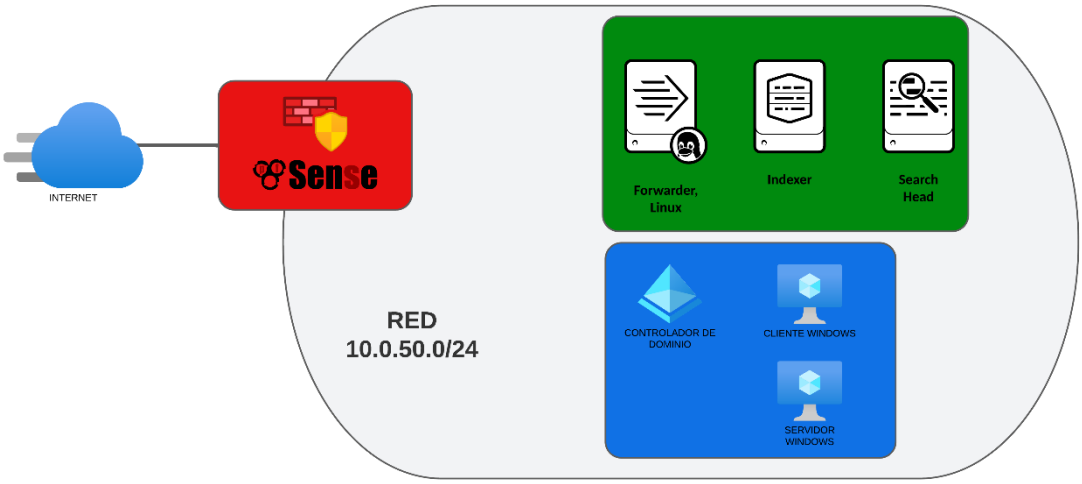


Figura 9: Propuesta arquitectura de detección

3.3 Reglas y modelos de detección

En este trabajo se pretende diseñar un conjunto de reglas y modelos que permitan evaluar el desempeño de ambos métodos de detección, para ello se han escogido dos dominios de eventos sobre los cuales se aplicarán las diferentes reglas y modelos observando en cada caso que solución prevalece sobre la otra en cuanto a efectividad se refiere.

El primer dominio de detección es el de red, para este dominio aplicaremos reglas y modelos que pretendan detectar ataques de reconocimiento, ataques de denegación de servicio y otro tipo de ataques de índole similar que puedan ser detectados a través de los eventos generados.

El segundo dominio es el de autenticación y directorio, para este dominio emplearemos reglas y modelos de detecten ataques de fuerza bruta sobre credenciales en el dominio empresarial, anomalías en la autenticación o credenciales del sistema.

4. Implementación de la solución

4.1 Infraestructura

La infraestructura propuesta para el presente trabajo se ha dividido en tres apartados, todos ellos se encargan de definir los elementos necesarios para la constitución del *security data lake* y los dominios sobre los cuales se realizará la monitorización, así como los flujos de información a alto nivel.

El primer apartado detalla la composición y elementos necesarios para el sistema encargado de realizar la ingesta de datos y posterior detección de anomalías de seguridad sobre los mismos.

El segundo apartado se encarga de detallar a grandes rasgos el diseño a alto nivel del flujo de información relativo a los eventos generados en el dominio de red y su posterior ingesta en el sistema.

El tercer apartado detalla de igual manera el diseño a alto nivel del flujo de información relativo a los eventos generados en el dominio de autenticación y directorio y su posterior ingesta en el sistema.

4.1.1 Sistema SDL

El sistema *security data lake* tiene como objetivo servir como herramienta base para la detección del laboratorio, para su implementación se ha optado por utilizar máquinas virtuales teniendo en cuenta la optimización y los requisitos establecidos por el fabricante [26] de recursos con las siguientes características.

HOST	CPU	RAM	ALMACENAMIENTO	SO
SPL_SH	4vCPUs	8GB	500GB (thin)	CentOS7
SPL_IDX	4vCPUs	8GB	500GB (thin)	CentOS7
SPL_FW	2vCPUs	2GB	250GB (thin)	CentOS7

La instalación de los diferentes aplicativos se ha llevado a cabo siguiendo las instrucciones del fabricante [27] y se ha detallado en el Anexo I.

Así mismo se ha procedido a la configuración del entorno de manera distribuida conformando con ello un sistema data lake que permitirá almacenar y analizar grandes cantidades de datos no estructurados separando con ello las capas de

recolección, ingestión y análisis consiguiendo así el poder escalar en función de las necesidades, la configuración llevada a cabo se ha detallado en el Anexo II

4.1.2 Entorno de detección en el dominio de red

Para el entorno de detección se ha implementado un elemento de red cortafuegos que será el encargado de proveernos de visibilidad y todos los eventos relacionados con este dominio, así mismo para este dominio se han configurado herramientas IDS como suricata [28] y de inspección de paquetes profunda como zeek [29].

Las características de las máquinas virtuales se han establecido teniendo en cuenta los recursos disponibles y se corresponden con la siguiente tabla.

HOST	CPU	RAM	ALMACENAMIENTO	SO
FW01	2vCPUs	512MB	500GB (thin)	FreeBSD

La configuración de los eventos se realiza mediante el protocolo SYSLOG, así mismo también se ha configurado el reenvío de todo el tráfico mediante una interfaz hacia los elementos IDS y de inspección de paquetes quedando como sigue en el siguiente esquema

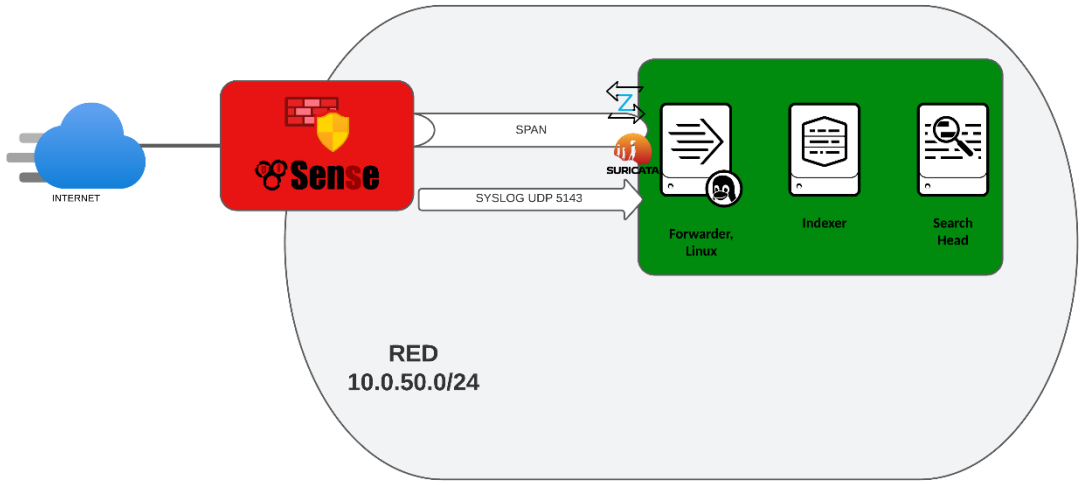


Figura 10: Esquema general de detección de red

Toda la configuración relativa a los diferentes elementos del dominio de red se encuentra detallada en el ANEXO III de este trabajo.

4.1.3 Entorno de detección autenticación y directorio

En el entorno de detección de autenticación y directorio se ha optado por la implementación de un dominio Windows con arquitectura de recolección de eventos unificada.

En total existen dos servidores y un cliente, en la arquitectura implementada uno de los servidores tiene el rol de controlador de dominio autenticando y

almacenando los eventos de autenticación de todos los ordenadores y servidores del dominio, el segundo servidor es un servidor que tiene el rol de recolector de eventos de eventos de Windows, al estar configurado de esta manera todos los eventos generados en el dominio se redirigen a esta máquina permitiendo la recolección desde un único punto unificado, en este servidor es donde se encuentra instalado el agente reenviador de Splunk.

Por último, el cliente tiene instalado un sistema operativo cliente donde se generarán los inicios de sesión y el resto de las actividades necesarias para llevar a cabo este trabajo.

Los recursos de las máquinas implementadas son las siguientes

HOST	CPU	RAM	ALMACENAMIENTO	SO
DC	4vCPUs	4GB	500GB (thin)	Windows Server 2016
WEF	4vCPUs	4GB	300GB (thin)	Windows Server 2016
W10	2vCPUs	2GB	150GB (thin)	Windows Server 2016

4.2 Reglas de detección

Las reglas y modelos de detección que se han propuesto para el trabajo tienen como objetivo evaluar las metodologías más comunes para identificar patrones anómalos en el conjunto de datos recibidos para los diferentes dominios, para estos dominios de red y autenticación se ha buscado crear reglas que monitoricen casos de uso como la exfiltración de datos o ataques de fuerza bruta respectivamente.

En el primer apartado se han expuesto el conjunto de instrucciones lógicas que definen las reglas de correlación de los dominios de red y autenticación.

En el segundo apartado se han definido las características y algoritmos usados para la creación de los modelos de detección en los dominios de red y autenticación.

4.2.1 Reglas de correlación

Las reglas de correlación permiten comparar diferentes eventos con relaciones ya predefinidas y conocidas, este tipo de reglas permiten establecer una serie de condiciones que funcionarán como una alerta activándose cuando se cumplan las condiciones.

En Splunk este tipo de reglas se definen a través de alertas, que se accionarán en base a una consulta SPL, el lenguaje Search Processing Language [30]es un lenguaje de consultas diseñado por Splunk para ser usado en su software, la

sintaxis está basada en el uso de tuberías del sistema operativo UNIX donde existen una serie de comandos que efectúan transformaciones sobre los datos y el resultado de estos comandos puede ser enrutado a los comandos posteriores a través de la sintaxis usada en UNIX para el mismo propósito.

4.2.1.1 Dominio de red

La lógica implementada para la regla de correlación en el dominio de red buscará crear una alerta cada vez que un equipo de usuario haya transmitido una cantidad mayor a 250MB en un periodo de tiempo dado, el diagrama de flujo de la alerta se encuentra detallado en la figura 11.

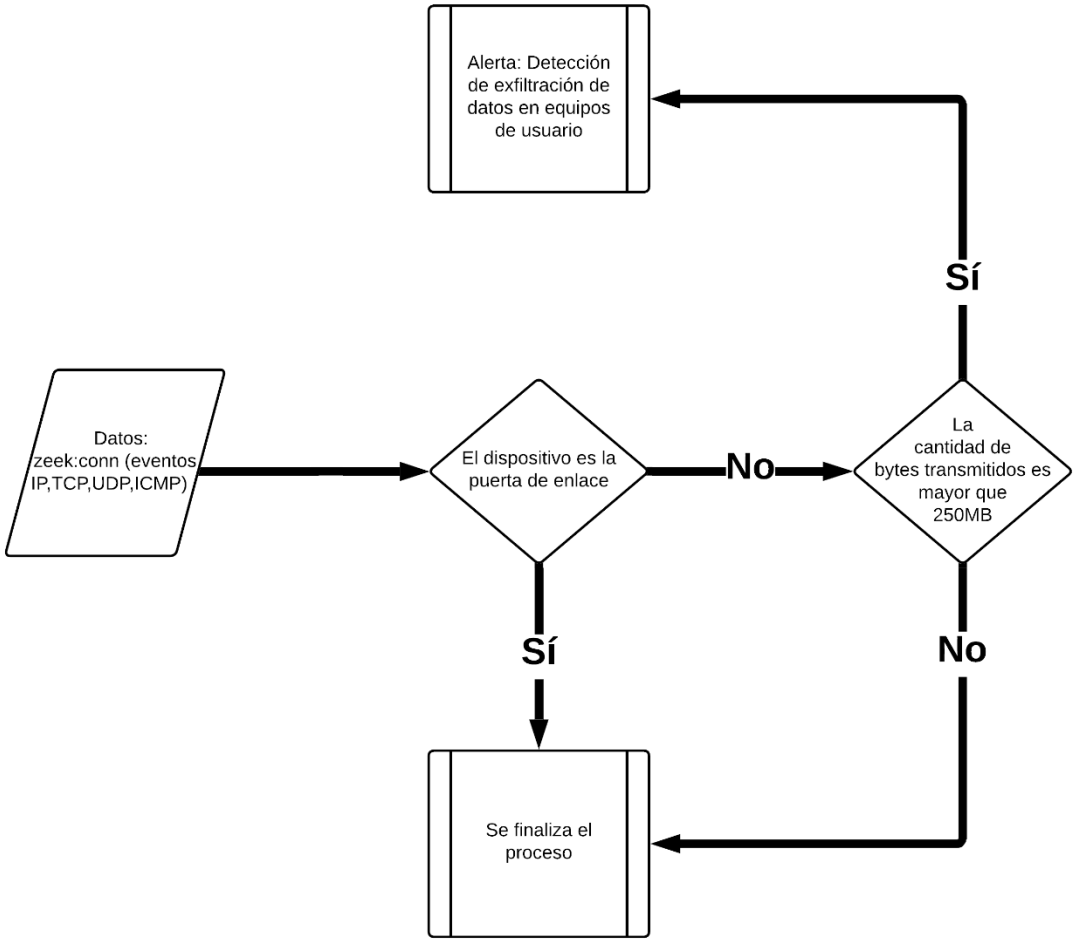


Figura 11: Diagrama regla de detección de exfiltración de datos

En este caso de uso se ha buscado crear una regla de correlación que genere una alerta cuando los bytes de salida del dominio superen un cierto umbral, para esta ocasión el umbral se ha establecido en 250MB cantidad suficiente para establecer una sospecha factible de exfiltración de datos.

El origen de información relativa a las conexiones se obtendrá del esquema de datos que proporcionan los ficheros de zeek [31], en concreto se utilizará las propiedades y valores que proporciona el fichero conn.log encargado de recopilar todas las conexiones TCP/UDP/ICMP y entre los datos que almacena

está el campo “orig_ip_bytes” que muestra el número de paquetes desde la dirección de origen a través del campo “total_length” de la cabecera de los paquetes IP.

La consulta se ha creado buscando trasladar la lógica inherente al caso de uso de búsqueda de exfiltración de la información en lenguaje SPL.

```
1. index=zeek sourcetype="bro:conn:json" id.orig_h!=192.168.1.254
2. | stats sum(orig_ip_bytes) as bytes_out by id.orig_h id.resp_h
3. | rename id.orig_h as src_ip, id.resp_h as dest_ip
4. | where bytes_out>250000000
5. | table src_ip dest_ip bytes_out
```

En la primera línea acotamos la búsqueda definiendo el índice donde está almacenada la información que buscamos y el tipo concreto de información a través del tipo de origen index y sourcetype respectivamente, después eliminamos de los resultados la IP del router para evitar obtener resultados duplicados ya que esta al actuar de puerta de enlace para todas las conexiones replicará la conexión realizada desde el host.

En la segunda línea mediante el comando “stats” se realiza una agrupación de los diferentes valores en función de los argumentos escogidos, en este caso estamos creando un conjunto mediante la cláusula “by” del par de valores dirección ip de origen y dirección ip de destino y sumando mediante la función “sum()” todos los bytes enviados para este conjunto.

En la tercera línea modificamos mediante el comando “rename” las cabeceras de las columnas para facilitar la lectura y no depender de la nomenclatura de los campos de la fuente de origen.

En la cuarta línea se aplica un filtro a los datos mediante el comando “where”, permitiendo con ello que únicamente se muestren aquellos conjuntos cuyo valor para el campo “bytes_out” calculado previamente sea mayor de 250MB.

Por último, el comando “table” permite mostrar los datos obtenidos en formato tabular representando los campos “src_ip” “dest_ip” “bytes_out” definidos previamente.

El resultado de ejecutar esta consulta aparece en la figura número 12, donde se puede observar las direcciones ip del flujo de comunicación, y la cantidad de datos enviados en bytes.

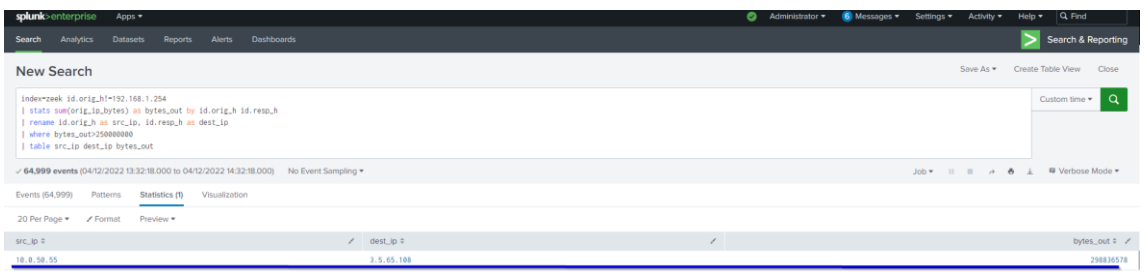


Figura 12: Resultados búsqueda de correlación

Una vez comprobada la correcta funcionalidad de nuestra consulta deberemos ajustar el rango de tiempo del cual obtendremos los datos ya que dependiendo del valor que tomemos en este parámetro la alerta podría tener o no, por ejemplificar esta afirmación un posible caso de exfiltración de información podría darse si la suma de bytes de salida desde una determinada ip a otra supera los 250MB en un rango de 60 minutos, sin embargo si el rango de tiempo se estableciera en dos meses la consulta dejaría de tener sentido ya que es bastante probable que la suma de los bytes de salida de una misma ip a una determinada dirección ip llegue a alcanzar cantidades mayores a 250MB.

Para crear la alerta se deberá ajustar el rango de tiempo acorde al caso de uso establecido, para el caso de exfiltración de datos configuraremos un rango de tiempo de 60 minutos.

La figura número 13 muestra el menú de configuración donde se especifica el rango de tiempo relativo sobre el cual se ejecutará la consulta en lenguaje SPL.

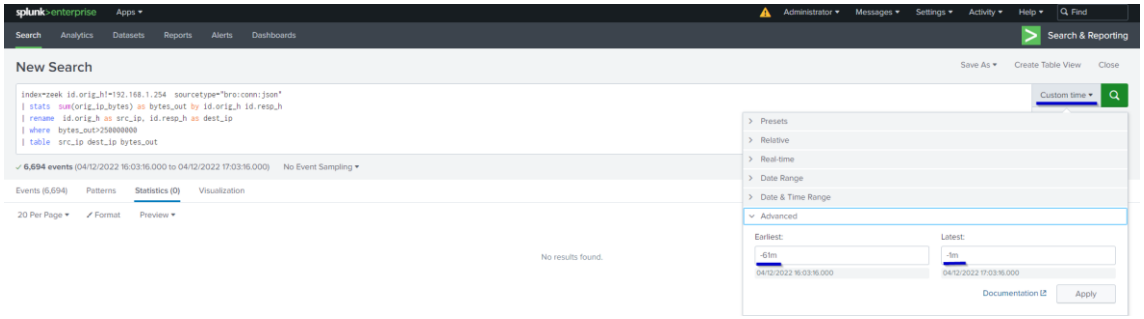


Figura 13: Menú de configuración rango de tiempo en Splunk

Hecho esto se guardará la búsqueda como una alerta a través del menú guardar como, después se deberá de configurar la alerta para que se ejecute en el sistema cada hora mediante una expresión de cron permitiendo así que la búsqueda consulte los 60 minutos anteriores.

La figura 14 muestra el menú de configuración de alertas en concreto la parte relativa a la programación de ejecución mediante una expresión de cron.

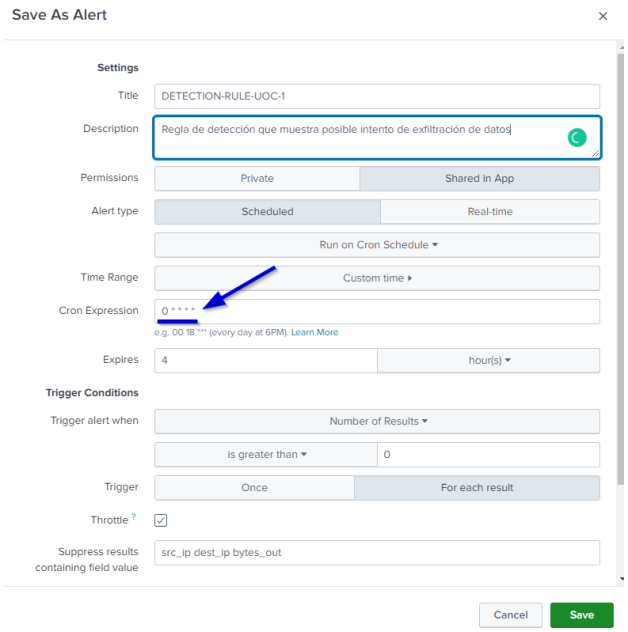


Figura 14: Creación de una alerta y configuración de ejecución en Splunk

4.2.1.2 Dominio de autenticación y directorio

La lógica implementada para la regla de correlación del dominio de autenticación y directorio buscará crear una alerta si detecta que ha existido un intento de fuerza bruta exitoso sobre una cuenta de usuario, los parámetros establecidos para activar esta alerta serán que el usuario haya fallado el inicio de sesión 100 o más veces y que la cuenta de usuario haya conseguido iniciar sesión exitosamente una o más veces, el diagrama de flujo de la alerta se encuentra detallado en la figura 15.

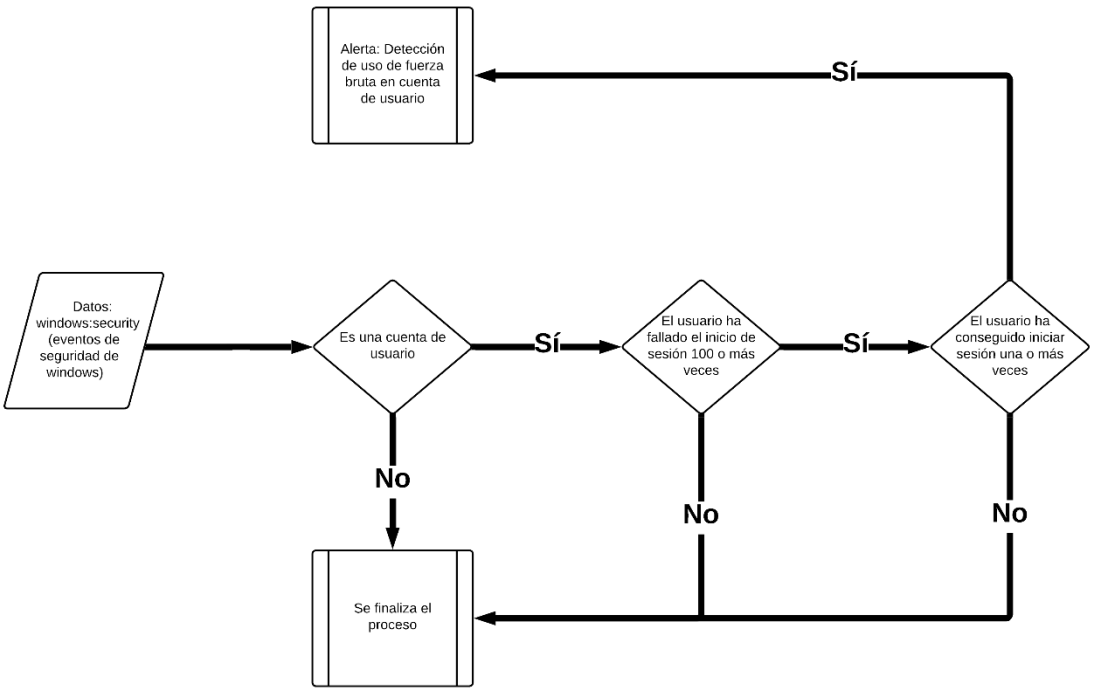


Figura 15: Diagrama regla de detección de uso de fuerza bruta en cuenta de usuario

La regla de correlación implementada para este dominio establecerá un umbral para alertar en caso de un gran número de autenticaciones fallidas desde un mismo origen.

Los eventos utilizados para alimentar a este caso de uso provendrán de los eventos generados por el sistema operativo Windows, los cuales proporcionarán datos como el origen, el destino y el estado de la autenticación.

La consulta realizada para obtener los datos en leguaje SPL acorde al diagrama de la figura 15 es la siguiente

```
1. index=wineventlog source="*WinEventLog:Security" user!="*$*"
2. | stats count(eval(action="success")) as correctos
count(eval(action="failure")) as fallos by src
3. | rename src as origen
4. | where correctos>0 AND fallos>100
5. | table src_ip dest_ip bytes_out
```

En la primera línea se definen los filtros que aplicarán a la búsqueda, en este caso escogeremos el índice relativo a los eventos Windows y el tipo de eventos relativos a los eventos de autenticación, también especificaremos un filtro para evitar los usuarios con el carácter $ ya que estos usuarios son usuarios de servicio o computador por norma general.

En la segunda línea realizamos una agrupación de los datos por origen a través del comando “stats”, creando en el proceso dos variables que se corresponderán al número de eventos cuyo campo “action” corresponda a autenticaciones exitosas y fallidas respectivamente.

En la tercera línea renombramos a través del comando “rename” la columna src para que se muestre como origen.

En la cuarta línea establecemos la condición para mostrar únicamente aquellos eventos que en el rango de tiempo seleccionado para la búsqueda hayan acumulado una o más autenticaciones exitosas y tengan más de 100 autenticaciones fallidas.

Figura 16: Resultado regla de correlación para dominio de autenticación

Al igual que en el caso anterior, se deberá guardar esta consulta como una alerta mediante el menú guardar como, estableciendo los mismos parámetros que en la consulta anterior.

4.2.2 Modelos de detección

En Splunk la creación de modelos de detección avanzada se realiza a través de la aplicación MLTK [32], esta aplicación añade una serie de comandos al lenguaje SPL que permiten interactuar y aplicar los diferentes algoritmos que contiene la aplicación a los datos ya contenidos en el despliegue de la solución pudiendo crear modelos y testear estos modelos contra datos en tiempo real a través de consultas en lenguaje SPL.

Para la definición de los modelos, la aplicación de Splunk MLTK provee de los comandos “fit” y “apply” [33], en concreto el comando “fit” se encarga de aplicar el algoritmo elegido al conjunto de datos devuelto por una búsqueda, a bajo nivel las acciones que realiza este comando para generar el modelo de datos son las siguientes.

- Los resultados de la búsqueda se almacenan en memoria.
- Se realizan las siguientes transformaciones sobre los datos.
    - Se descartan campos cuyo valor sea nulo de todos los eventos retraídos.
    - Se descartan los campos no numéricos con más de 100 valores distintos.
    - Se descartan eventos con cualquiera de los campos nulos.
    - Se transforman los campos no numéricos en variables utilizando el algoritmo “one-hot encoding”.
    - Se convierten los datos en una matriz de números y aplica el algoritmo elegido para crear un modelo.
- Se aplica el modelo a los datos preparados y se crean nuevas columnas que muestran la predicción.
- El modelo se codifica y se guarda en Splunk.

El comando “apply” se encarga de aplicar uno de los modelos previamente guardados por el comando “fit”, las acciones que realiza a bajo nivel son las siguientes.

- Se carga el modelo aprendido mediante el comando “fit”
- Se realizan las siguientes transformaciones en el conjunto de datos de la búsqueda
    - Se descartan cualquier campo que sea nulo
    - Se descartan los campos no numéricos con más de 100 valores distintos.
    - Se transforman los campos no numéricos en variables utilizando el algoritmo “*one-hot encoding*”.
    - Se descartan las variables que no estén presentes en el modelo a aplicar
    - Se reemplazan las variables no existentes con ceros
    - Se convierte estos datos en una matriz numérica
- Se aplica el modelo a los datos que se han preparado en los pasos anteriores y se producen nuevas columnas que muestran las predicciones.

4.2.2.1 Dominio de red

El modelo implementado para la detección de anomalías en el dominio de red buscará analizar el tamaño de los datos enviados y recibidos por un único dispositivo en los diferentes tramos horarios del día con respecto los mismos valores en periodos de tiempo anteriores, en el caso de encontrar una desviación notable del resultado esperado utilizando como algoritmo la función de densidad se generará una alerta, el diagrama de flujo de comportamiento se puede encontrar en la figura 17.

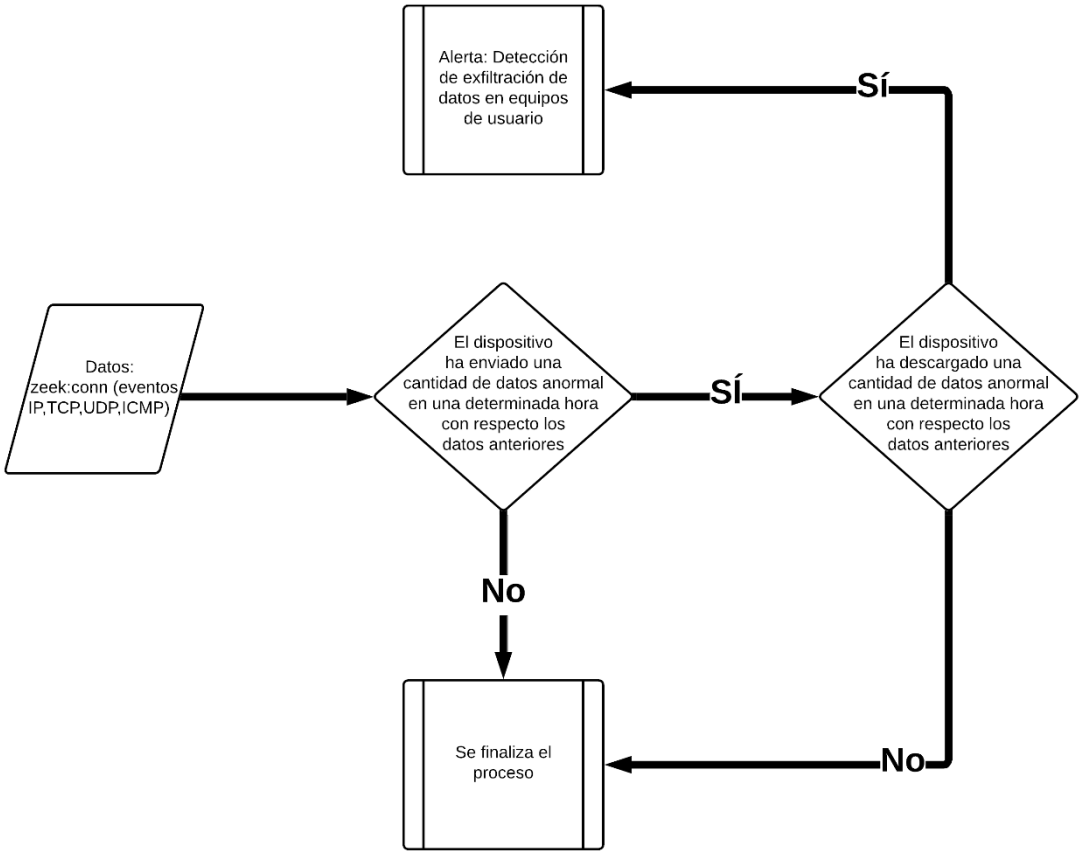


Figura 17: Diagrama modelo de detección de exfiltración de datos

El modelo creado permite detectar anomalías en el tráfico de red, en concreto este se ha desarrollado intentando buscar periodos de tiempo donde hayan existido patrones anómalos de transferencia de datos, al igual que en el caso de uso de la regla de correlación mediante este modelo se debería poder detectar una exfiltración de datos sin necesidad de establecer un umbral de datos enviados.

Para crear el modelo de detección utilizaremos la fuente de datos del fichero conn.log proveniente de zeek ya que esta fuente de información proporciona datos relevantes a todas las conexiones que se han producido en la red y desde donde podremos sacar parámetros como la cantidad de bytes tanto enviados como recibidos por cada conexión realizada.

El algoritmo usado para la detección de anomalías sobre el conjunto de datos será la función probabilística de densidad, esta función permite inferir como de probable es que una variable aleatoria continua obtenga un determinado rango de valores, esta probabilidad se obtiene calculando la integral de la variable sobre ese rango de valores entre un punto y otro.

A grandes rasgos, para las personas no versadas en la estadística esta función opera observando el pasado para crear un modelo que permita establecer umbrales dinámicos, a partir de aquí y mediante estos umbrales definidos es posible determinar que es normal o que no es normal.

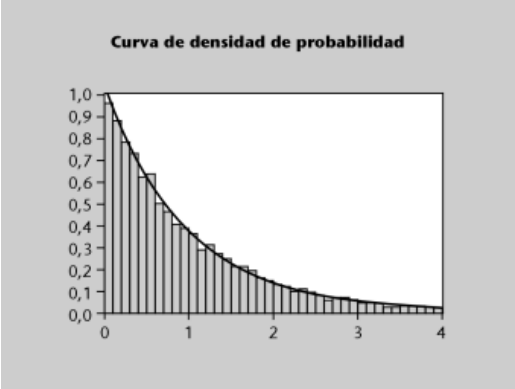


Figura 18: Curva de densidad de probabilidad [34]

Entre las ventajas asociadas a este algoritmo es la facilidad de uso que ofrece ofreciendo grandes resultados incluso a aquellos con limitados conocimientos estadísticos, por el contrario uno de los mayores inconvenientes es que puede producir falsos positivos.

La consulta realizada para la obtención de los datos y creación del modelo ha sido la siguiente

```
1. index=zeek sourcetype="bro:conn:json"
2. | eval src_dest='id.orig_h'." | ".'id.resp_h'
3. | bin _time span=5m
4. | stats sum(resp_ip_bytes) as bytes_in sum(orig_ip_bytes) as bytes_out
by _time src_dest
5. | eval horaDelDia=strftime(_time,"%H")
6. | fit DensityFunction bytes_in by horaDelDia as outlier_bytes_in into
bytes_in_anomalia_modeloDeDeteccion
7. | fit DensityFunction bytes_out by horaDelDia as outlier_bytes_out into
bytes_out_anomalia_modeloDeDeteccion
```

En la primera línea se establecen los filtros necesarios para acotar la búsqueda y optimizar así los recursos que se dedicarán a la obtención de la información, el filtro se corresponde con los datos almacenados en el índice de los eventos recopilados por zeek y en concreto a los del tipo conn.log ya que proporcionarán la información deseada.

En la segunda línea con el comando “eval” generamos un nuevo campo con el nombre de src_dest, este campo se alimentará de la concatenación de los valores de la dirección IP de origen y la dirección IP de destino.

En la tercera línea a través del comando “bin” agruparemos los eventos recibidos por sus valores numéricos en el campo _time que es el campo que especifica el momento en el que ese evento fue generado, con la cláusula “span” definiremos el tiempo entre conjuntos, en este caso estableceremos conjuntos de valores para cada 5 minutos en el total de la búsqueda.

En la cuarta línea con el comando “span” se crea una conjunto de eventos agrupados por el campo _time y el campo creado anteriormente src_dest donde se realizan dos operaciones de suma sobre los bytes recibidos y enviados por el conjunto respectivamente.

En la quinta línea creamos el campo horaDelDia el cual aprovisionaremos utilizando el campo _time de todos los eventos y dándole el formato necesario a través de la función strftime para retraer la hora.

En la sexta y séptima línea utilizaremos el comando “fit” para generar el modelo, también se utiliza la cláusula “DensityFunction” para designar el algoritmo que queremos utilizar, sendos modelos se crean para cada conjunto de horas especificando la cláusula “by” horaDelDia y finalmente con la cláusula “into” designamos el nombre con el que se guardará el modelo.

El resultado del comando es el siguiente.

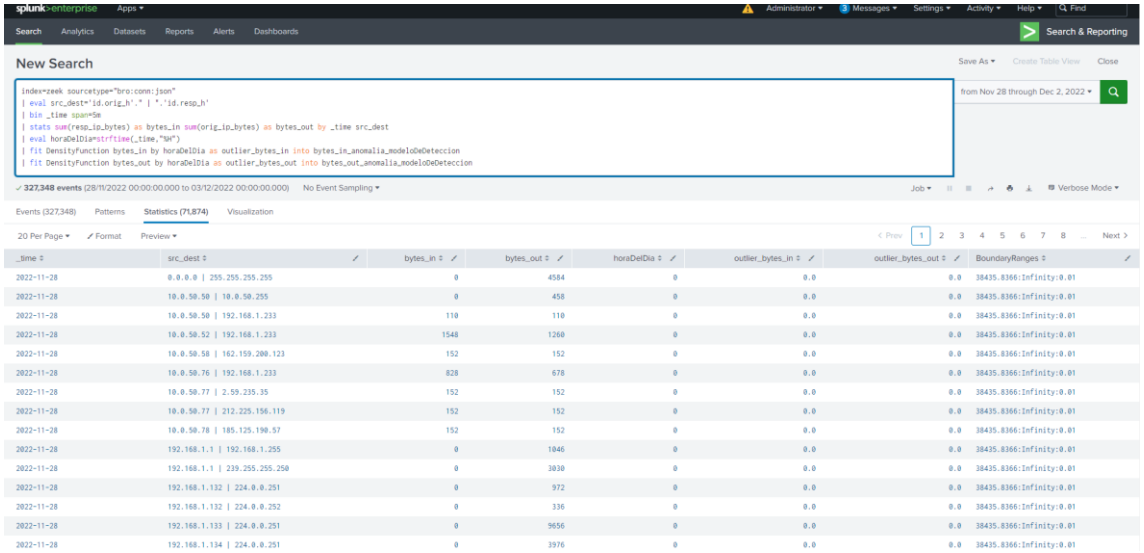


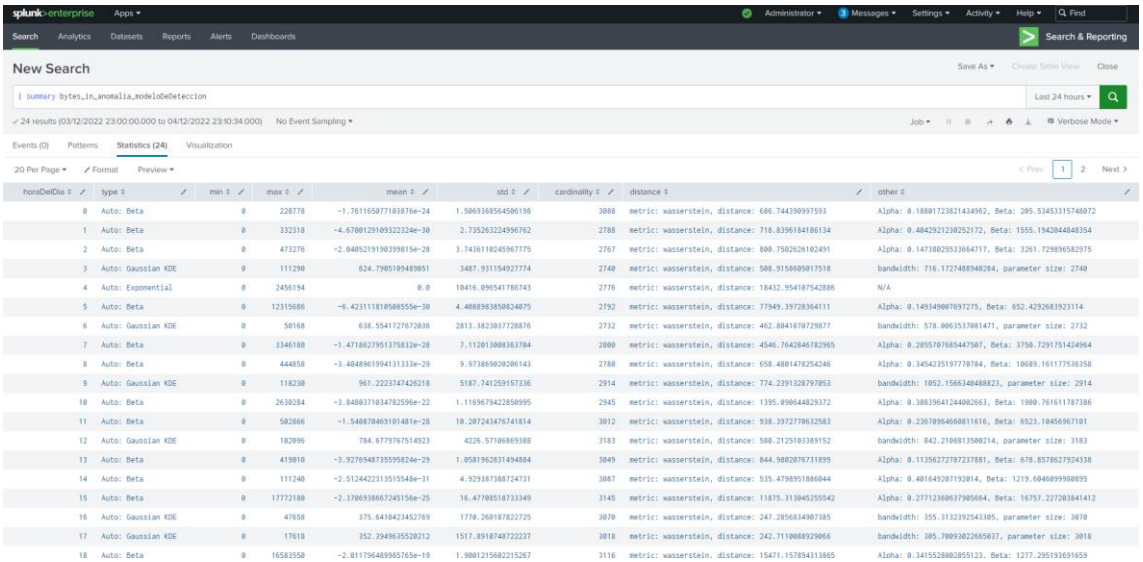
Figura 19: Creación de modelo de detección

Para evaluar los parámetros del modelo se usará el comando “summary” este comando proporcionará toda la información necesaria relativa al modelo que ha sido creado.

La información proporcionada por el comando facilita entre otras cosas los valores para la media y la desviación estándar de los datos, también facilita la

cardinalidad que son el número de puntos de datos utilizados para ajustar la función de densidad.

En la figura 20 se aprecia como mediante el uso del comando “summary” seguido del nombre del modelo creado, Splunk devuelve una tabla de resultados donde se puede observar en función de la hora del día las características usadas para la función de densidad, para cada hora del día se puede observar como Splunk automáticamente ha elegido una función de densidad de probabilidad continua entre Normal, Exponencial, Gaussiana KDE y distribución Beta.



horaDelDia	type	min	max	mean	std	cardinality	distance	other
0	Auto: Beta	0	228778	-1.761165077103876e-24	1.5069368564506198	3088	metric: wasserstein, distance: 686.744390997593	Alpha: 0.18801723821434962, Beta: 205.53453315748072
1	Auto: Beta	0	332318	-4.670012910932232 4e-30	2.735263224996762	2788	metric: wasserstein, distance: 718.8396184186134	Alpha: 0.40429212302521 72, Beta: 1555.1942044848354
2	Auto: Beta	0	473276	-2.040521919039398 15e-28	3.7436118245967775	2767	metric: wasserstein, distance: 800.7502626102491	Alpha: 0.14738025533664717, Beta: 3261.729896582975
3	Auto: Gaussian KDE	0	111290	624.7905109489051	3487.931154927774	2740	metric: wasserstein, distance: 508.9158605017518	bandwidth: 716.1727488940284, parameter size: 2740
4	Auto: Exponential	0	2456194	0.0	10416.096541786743	2776	metric: wasserstein, distance: 18432.954107542886	N/A
5	Auto: Beta	0	12315686	-6.423111810508555e-30	4.408898385082407 5	2792	metric: wasserstein, distance: 77549.35728364111	Alpha: 0.1493490007697275, Beta: 652.4292683923114
6	Auto: Gaussian KDE	0	50168	638.5541727672036	2813.3823037728876	2732	metric: wasserstein, distance: 462.8041070729877	bandwidth: 578.0063537081471, parameter size: 2732
7	Auto: Beta	0	3146180	-1.471862795137583 2e-28	7.1120130008363704	2800	metric: wasserstein, distance: 4546.7642846782965	Alpha: 0.2855707685447507, Beta: 3750.7291751424964
8	Auto: Beta	0	444858	-3.404896199413133 3e-29	9.973869020206143	2780	metric: wasserstein, distance: 658.4801478254246	Alpha: 0.3454235197770784, Beta: 10689.161177536358
9	Auto: Gaussian KDE	0	118230	961.2223747426218	5187.741259157336	2914	metric: wasserstein, distance: 774.2391328797053	bandwidth: 1052.1566340488823, parameter size: 2914
10	Auto: Beta	0	2630284	-3.848037103478259 6e-22	1.1169679422850995	2945	metric: wasserstein, distance: 1395.090644829372	Alpha: 0.3083964124400266 3, Beta: 1900.761611787386
11	Auto: Beta	0	502866	-1.540870469101481e-28	10.207243476741814	3012	metric: wasserstein, distance: 938.3972770632583	Alpha: 0.2367096466081161 6, Beta: 6523.10456967101
12	Auto: Gaussian KDE	0	102096	784.6779767514923	4226.57106865388	3183	metric: wasserstein, distance: 580.2125103389152	bandwidth: 842.2106813500214, parameter size: 3183
13	Auto: Beta	0	415010	-3.927654873559582 4e-29	1.0581962831494884	3049	metric: wasserstein, distance: 844.9802076731899	Alpha: 0.1135627278723788 1, Beta: 678.8578627924338
14	Auto: Beta	0	111240	-2.512442231351554 8e-31	4.929387388724731	3087	metric: wasserstein, distance: 535.4798951886044	Alpha: 0.4016492071920 14, Beta: 1219.6046099980895
15	Auto: Beta	0	17772180	-2.370693866724515 6e-25	16.47708518733349	3145	metric: wasserstein, distance: 11875.313045255542	Alpha: 0.2771236063790566 4, Beta: 16757.227203841412
16	Auto: Gaussian KDE	0	47658	375.6410423452769	1770.2681878222725	3070	metric: wasserstein, distance: 247.2856834907385	bandwidth: 355.3132392543305, parameter size: 3070
17	Auto: Gaussian KDE	0	17618	352.3949635520212	1517.8910748722237	3018	metric: wasserstein, distance: 242.7110088929066	bandwidth: 305.7009302266503 7, parameter size: 3018
18	Auto: Beta	0	16583558	-2.011796489985765e-19	1.9001215602215267	3116	metric: wasserstein, distance: 15471.157894313865	Alpha: 0.3415528002855123, Beta: 1277.2951936916 59

Figura 20: Características del modelo de detección creado

4.2.2.2 Dominio de autenticación y directorio

El modelo establecido para el dominio de autenticación y directorio buscará patrones anormales comparando la cantidad de veces que un usuario ha fallado el inicio de sesión en una hora determinada con respecto el histórico de datos anteriores usando para ello el algoritmo de la función de densidad, el diagrama de flujo se puede observar en la figura 21

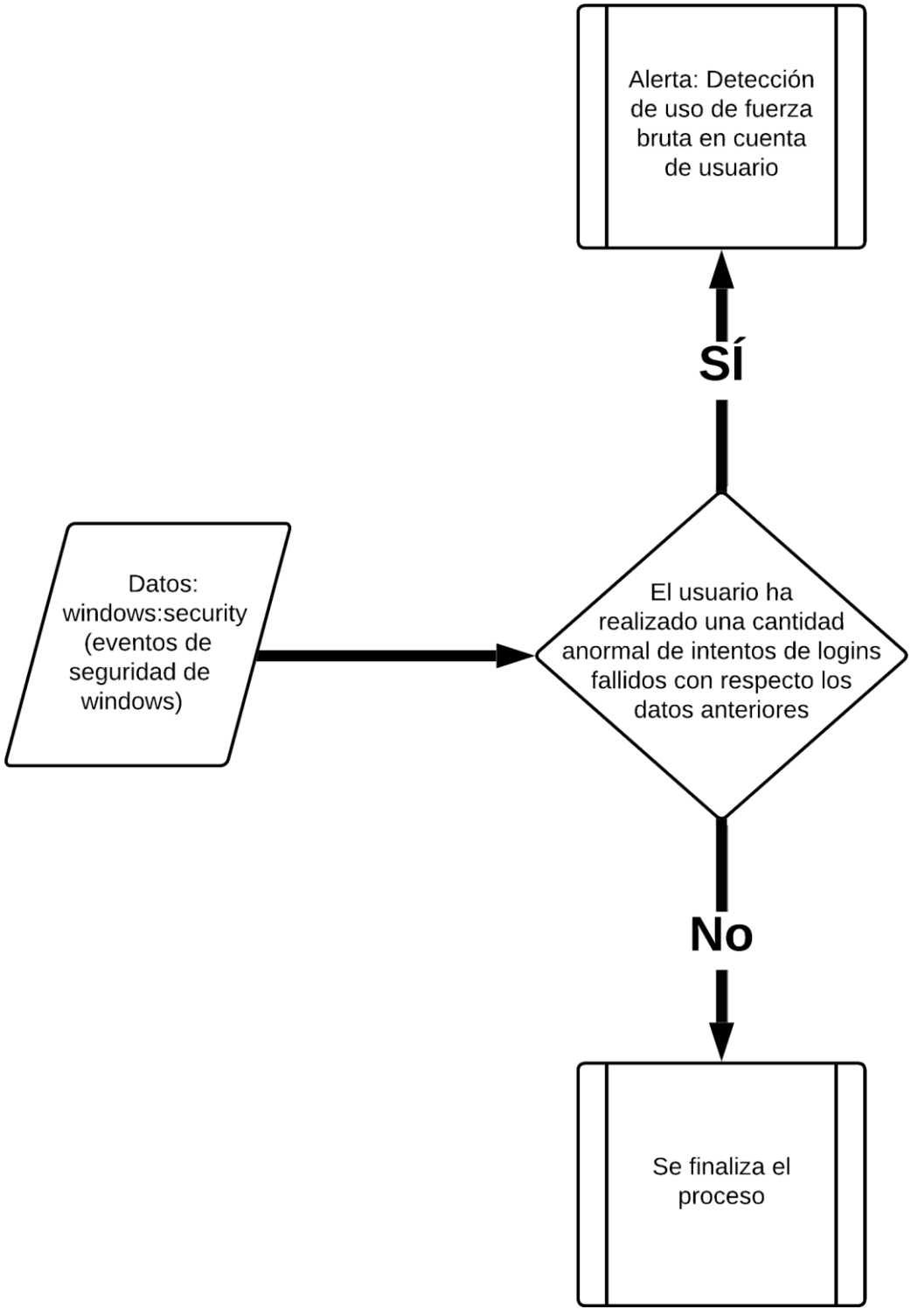


Figura 21: Diagrama de modelo de detección de uso de fuerza bruta

Este modelo se ha implementado utilizando la función de densidad, lo que permite que se encargue de registrar todos los intentos de autenticación fallidos en periodos de 10 minutos enriqueciendo el data set con la característica de la hora del día donde se produzca.

La implementación de este modelo a través del lenguaje SPL se puede observar en el siguiente fragmento de código.

```
1. | tstats count as logins_f WHERE index=wineventlog
source="wineventlog:security" TERM(EventCode=4625) BY _time
span=10m host
2. | eval horaDelDia=strftime(_time,"%H")
3. | fit DensityFunction logins_f by horaDelDia as anomalia_login into
app:fallos_de_login_modelo
```

En la primera línea mediante el comando “tstats” se crea un conjunto de datos agregados sobre los filtros establecidos con la cláusula “WHERE”, en este caso se ha filtrado por el índice donde se encuentran los eventos Windows y por el evento cuyo código es relativo a los intentos de autenticación fallidos en los sistemas operativos Windows, esta agrupación se ha realizado en conjuntos de 10 minutos y se han agrupado por la hora y el equipo en cuestión.

En la segunda línea se ha creado una variable que establece un formato mediante la función “strftime” al campo “_time” de cada agrupación.

En la tercera línea mediante el comando “fit” se procede a la creación del modelo utilizando para ello el algoritmo de la función de densidad.

4.3 Evaluación de los resultados

La evaluación de los resultados se ha realizado tomando en cuenta diversos parámetros relativos a la efectividad del resultado, para ello se han utilizado técnicas de simulación de ataques de fuerza bruta mediante programas especializados en este campo como *Infection Monkey* [35].

En la evaluación de las reglas y modelos se ha diseñado un marco de referencia para poder establecer un método de valoración cuantitativo, en este marco se han tenido en cuenta los siguientes parámetros:

- Precisión – El modelo o la regla ha sido capaz de detectar la amenaza,
- Falso positivo – El modelo o regla ha identificado eventos normales como maliciosos
- Falso negativo – El modelo o regla ha identificado eventos maliciosos como normales

En función de los diferentes parámetros se han otorgado diferentes puntuaciones acorde a los resultados obtenidos, este rango estará definido desde -3 hasta +3, el proceso de evaluación y asignación de puntuaciones se encuentra detallado en el siguiente diagrama.

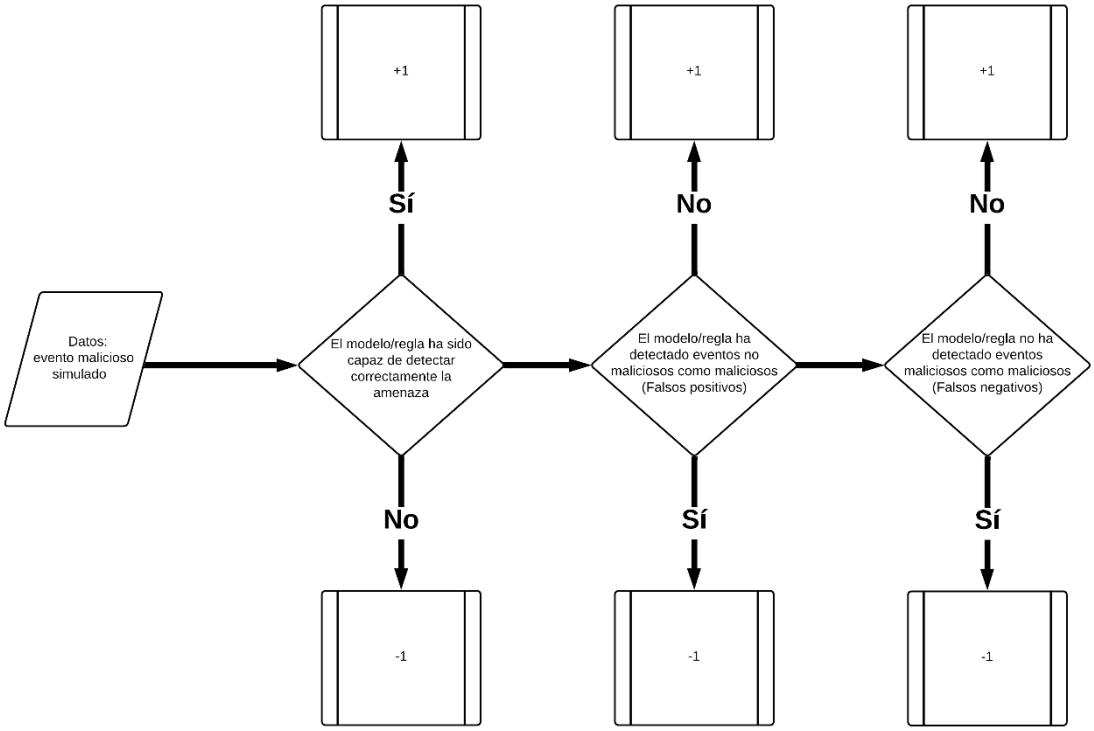


Figura 22: Diagrama de evaluación de modelos y reglas

4.3.1 Análisis detección de red

Para la evaluación de la regla de correlación, se ha procedido a la subida de un archivo mayor de 250MB a una web de intercambio de ficheros mediante el navegador web.

La consulta SPL relativa a la regla de correlación es la siguiente.

```
1. index=zeek sourcetype="bro:conn:json" id.orig_h!=192.168.1.254
2. | stats sum(orig_ip_bytes) as bytes_out by id.orig_h id.resp_h
3. | rename id.orig_h as src_ip, id.resp_h as dest_ip
4. | where bytes_out>250000000
5. | table src_ip dest_ip bytes_out
```

Una vez lanzada se puede comprobar que efectivamente aparece un único resultado correspondiente a la acción llevada a cabo por el host con dirección IP 10.0.50.55 tal y como se muestra en la figura 12.

La limitación relativa a este tipo de reglas es que solamente pueden proporcionar detección para un serie de casos de uso ya conocidos y acotados, sería plausible plantearse la siguiente cuestión un actor malicioso que conociera la lógica inherente a este tipo de reglas podría saltárselas fácilmente simplemente dividiendo el archivo en partes no mayores de 249MB, como beneficio se observa la optimización de las regla no produciendo apenas falsos positivos para actividades maliciosas acotadas y marcadas como tales.

Para la evaluación del modelo de detección se utilizará la siguiente consulta SPL.

```
1. index=zeek sourcetype="bro:conn:json" id.orig_h=10.0.50.55
2. | eval src_dest='id.orig_h'." | ".'id.resp_h'
3. | bin _time span=5m
4. | stats sum(resp_ip_bytes) as bytes_in sum(orig_ip_bytes) as bytes_out
by _time src_dest
5. | eval horaDelDia=strftime(_time,"%H")
6. | apply bytes_in_anomalia_modeloDeDeteccion
7. | apply bytes_out_anomalia_modeloDeDeteccion
8. | eval anomalyScore=outlier_bytes_in+outlier_bytes_out
9. | where anomalyScore>1
```

Como se puede observar a diferencia del comando SPL para la creación del modelo se ha substituido el comando “fit” por el comando “apply” ya que este último comando permitirá aplicar los modelos creados con anterioridad a los datos traídos por nuestra búsqueda, también se ha incluido el filtro inicial de la dirección IP de origen relativa al host desde el cual se realizó la subida del fichero para acotar la búsqueda.

En la línea seis y siete con el comando “apply” se aplica el modelo de detección creado con anterioridad a los datos obtenidos en la búsqueda, estos datos vendrán con dos campos adicionales llamados outlier o atípico, estos campos indicarán si los valores analizados son anómalos o no marcándose como uno o cero en cada caso respectivamente.

En la línea ocho creamos una variable llamada “anomalyScore” esta variable busca establecer el grado de anomalía que tienen los datos, ya que en este caso tenemos dos posibles variables “bytes_in” y “bytes_out” comparadas con dos modelos cada una de ellas pudiendo ser anómala o no. La variable “anomalyScore” se compondrá de la suma de los valores outlier o atípicos de las variables analizadas, lo que hace que el rango de valores que pueda tomar vaya de cero a dos, dependiendo de cada caso.

En la línea nueve con el comando “where” establecemos la lógica para ajustar el funcionamiento de nuestras alertas estableciendo que únicamente genere alertas cuando ambos valores tanto los bytes de entrada como de salida sean anómalos.

El resultado de la consulta devuelve 12 resultados constitutivos de posibles anomalías como se puede observar en la figura 23.

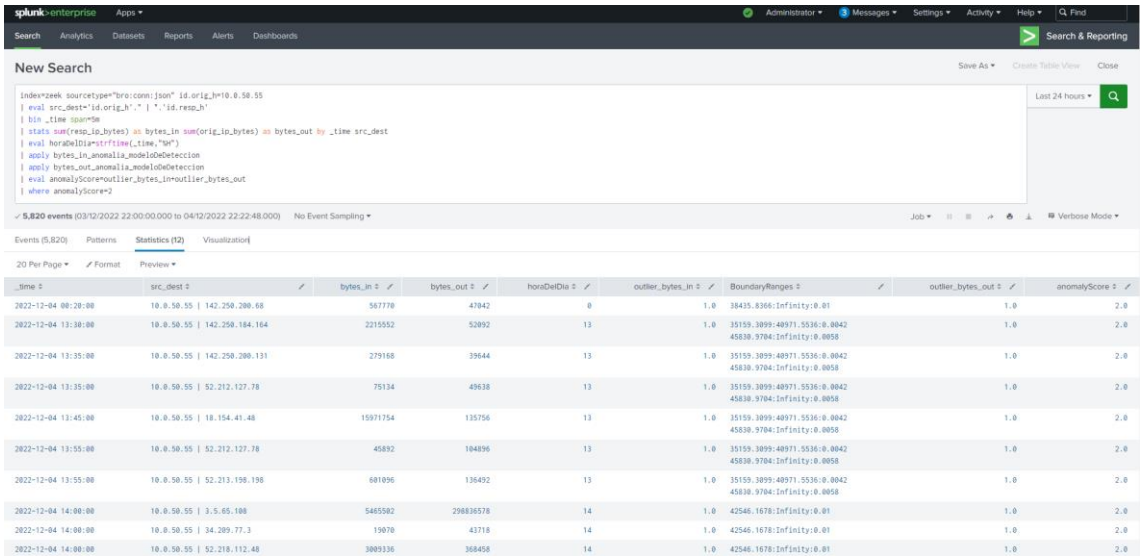


Figura 23: Resultado modelo de detección

Dado que la búsqueda devuelve más resultados de los esperados añadiremos la cláusula de “threshold” al comando “apply”, esta cláusula permite establecer el mínimo de datos que el algoritmo definirá como anomalías, por regla general está configurado con un valor de 0.01 lo que conlleva que al menos un 1% de los datos estarán marcados como anomalía, para ajustar el modelo reduciremos esta cantidad para evitar en la mayor medida de lo posible falsos positivos.

```
1. index=zeek sourcetype="bro:conn:json" id.orig_h=10.0.50.55
2. | eval src_dest='id.orig_h'." | ".'id.resp_h'
3. | bin _time span=5m
4. | stats sum(resp_ip_bytes) as bytes_in sum(orig_ip_bytes) as bytes_out
by _time src_dest
5. | eval horaDelDia=strftime(_time,"%H")
6. | apply bytes_in_anomalia_modeloDeDeteccion threshold=0.001
7. | apply bytes_out_anomalia_modeloDeDeteccion threshold=0.001
8. | eval anomalyScore=outlier_bytes_in+outlier_bytes_out
9. | where anomalyScore>1
```

Realizado el cambio y tras ejecutar de nuevo el comando se observa una notable disminución de los elementos catalogados como anómalos en función de nuestro modelo obteniendo solamente cinco comunicaciones desde el host el cual hemos realizado la subida de información a la web anómalas entre las que se encuentra la comunicación realizada manualmente a modo de evaluación.

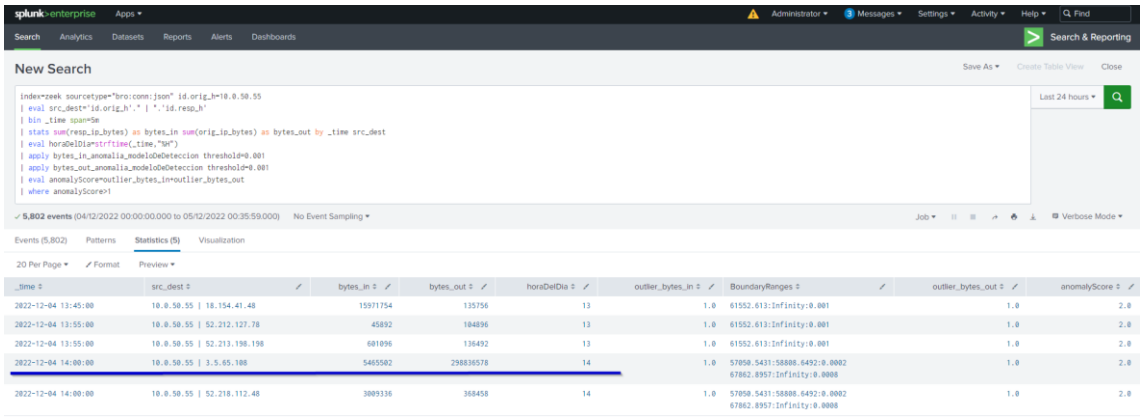


Figura 24: Resultado modelo de detección optimizado

Como se puede observar el modelo de detección ha sido capaz de detectar la actividad de evaluación llevada a cabo y catalogarla como anómala, todo ello sin necesidad de conocer previamente patrones de exfiltración de datos como por ejemplo la cantidad de información que se considera necesaria para catalogar la acción como maliciosa, pudiendo detectar amenazas no conocidas hasta el momento. El principal inconveniente que se encuentran son los falsos positivos que puede llegar a generar este tipo de reglas, ya que los 4 resultados obtenidos adicionales al conocido habría que estudiarlos individualmente para establecer si son maliciosos o no.

La puntuación final obtenida en nuestro marco de referencia para el dominio de red se muestra en la siguiente tabla.

	Precisión	Falsos positivos	Falsos Negativos	Total
Regla de correlación	+1	+1	+1	+3
Modelo de detección	+1	-1	+1	+2

Como se puede observar en nuestro marco de referencia obtendría mayor puntuación la regla de correlación respecto el modelo de detección.

4.3.2 Análisis detección autenticación y directorio

Para el análisis del dominio de autenticación y directorio se utilizado la herramienta Infection Monkey [35] esta herramienta proporciona un mecanismo automatizado que simula intrusiones en la red generando una gran cantidad de tráfico e intentos de autenticación contra las diferentes máquinas de nuestra red.

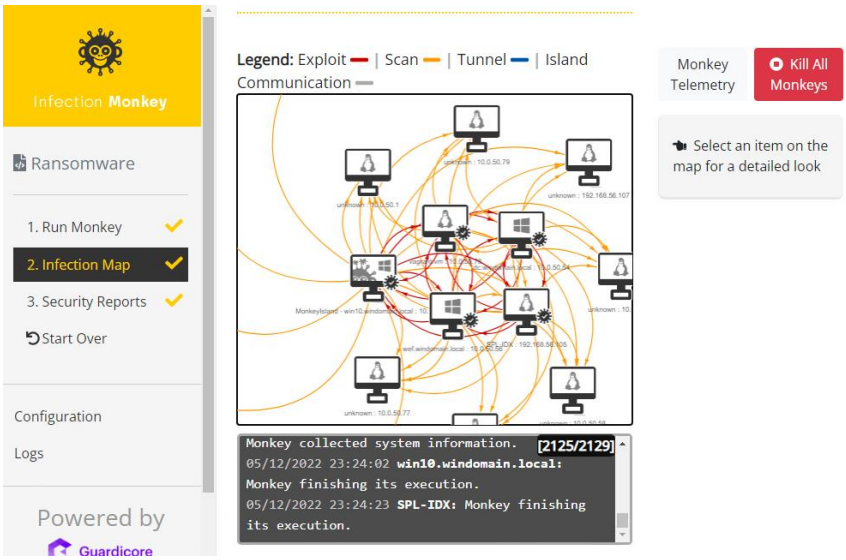


Figura 25: Menú herramienta Infection Monkey

Una vez finalizada la ejecución de la herramienta se procede a realizar la misma consulta relativa a la regla de correlación creada con anterioridad ya que mediante esta búsqueda devolverá todos aquellos accesos que hayan cumplido la lógica especificada en la regla.

```
1. index=wineventlog source="*WinEventLog:Security" user!="*$*"
2. | stats count(eval(action="success")) as correctos
count(eval(action="failure")) as fallos by src
3. | rename src as origen
4. | where correctos>0 AND fallos>100
5. | table src_ip dest_ip bytes_out
```

Una vez ejecutada la consulta para un rango de tiempo determinado devuelve los computadores que donde existen más de 100 fallos y autenticación y al menos un inicio de sesión correcto lo que podría ser consecutivo de actividad maliciosa.

El resultado de la consulta anterior se puede observar en la figura 16 donde muestra que se ha realizado un intento de fuerza bruta exitoso sobre el equipo DC.

Para validar el modelo de detección para este dominio lo primero que realizaremos será la aplicación de este sobre el conjunto de datos donde se ha utilizado la herramienta.

La búsqueda comando utilizado para aplicar el modelo y poder encontrar los elementos anómalos es el siguiente.

```
1. | tstats count as logins_f WHERE index=wineventlog
source="wineventlog:security" TERM(EventCode=4625) BY _time
span=10m host
2. | eval horaDelDia=strftime(_time,"%H")
3. | apply fallos_de_login_modelo
4. | search anomalia_login > 0
```

En la tercera línea se utiliza el comando “apply” para aplicar el modelo creado con anterioridad

En la cuarta línea se utiliza el comando “Search” para detectar únicamente aquellos elementos que hayan sido identificados como anómalos por el comando, pudiendo esta consulta ser ejecutada como una alerta al igual que en el caso de las reglas de correlación.

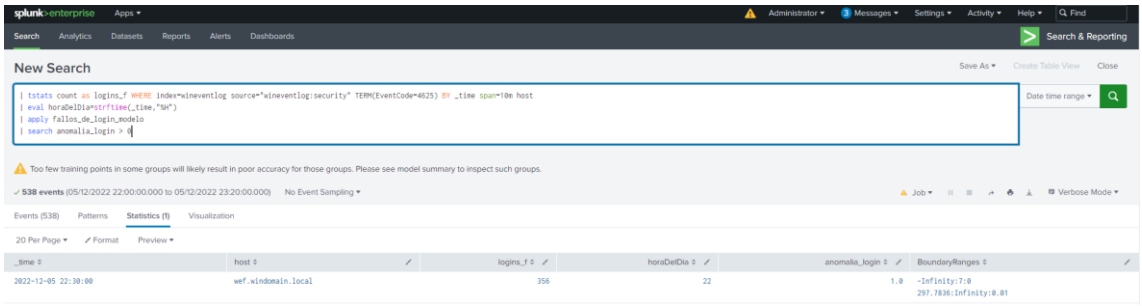


Figura 26: Resultado modelo de detección autenticación y directorio

En este dominio se ha podido comprobar el potencial que ofrecen los modelos de detección frente a las reglas de correlación tradicionales, mientras que estas solamente advierten ante hechos conocidos y bajo unos umbrales muy específicos, los modelos de detección de eventos posibilitan el análisis de gran cantidad de eventos y la detección de amenazas no conocidas, al igual que para el dominio anterior la mayor problemática que se presenta es la posibilidad de falsos negativos por lo que es crucial ajustar los valores del modelo de cara a obtener los mejores resultados posibles.

La puntuación final obtenida en nuestro marco de referencia para el dominio de autenticación y directorio se muestra en la siguiente tabla.

	Precisión	Falsos positivos	Falsos Negativos	Total
Regla de correlación	+1	+1	+1	+3
Modelo de detección	+1	+1	+1	+3

Como se puede observar en nuestro marco de referencia tanto la regla de correlación como el modelo han obtenido la misma puntuación.

5. Conclusiones y trabajos futuros

5.1 Conclusiones

En este trabajo se han expuesto las problemáticas y limitaciones actuales que sufren los sistemas de recolección y análisis de eventos, así mismo se ha realizado una introducción teórica a los principios de los sistemas de manejo de grandes cantidades de información haciendo especial mención a los sistemas *security data lake* como posible solución a esta problemática anteriormente mencionada y realizado una propuesta de solución basada en esta tecnología.

Para llevar a término el sistema propuesto se ha realizado un análisis del estado del arte respecto diferentes soluciones del mercado habiendo elegido Splunk como tecnología base para la implementación de la propuesta, se han expuesto los beneficios que ofrece Splunk para el desarrollo de un sistema distribuido de almacenamiento de datos de seguridad y detección de anomalías sobre estos, permitiendo gestionar una gran cantidad de datos debido a su naturaleza distribuida e implementar tanto reglas de correlación como modelos de detección de anomalías utilizando técnicas de *machine learning*.

La propuesta de sistema implementada en este trabajo ha permitido verificar la viabilidad de los sistemas *security data lake* para el tratamiento de grandes cantidades de datos y eventos relacionados con la seguridad informática de los entornos modernos y su capacidad de crecimiento virtualmente infinita gracias a los componentes distribuidos de la misma que permiten adaptarse a las necesidades actuales.

Así mismo, se han podido analizar e implementar varios métodos para la detección de patrones maliciosos, se ha explicado la diferencia entre reglas de correlación y modelos de detección usando algoritmos de *machine learning* como es el caso de la función de densidad de probabilidad, destacando los puntos fuertes y débiles de cada método propuesto.

Para la realización del análisis de los diferentes métodos se ha propuesto un escenario de prueba sobre el sistema *security data lake* implementado en el presente trabajo, este escenario se ha compuesto de dos dominios uno de red y otro de autenticación compuesto por los diferentes elementos generadores de eventos para cada uno de ellos. En la batería de pruebas realizadas mediante simulación de ataques en los diferentes dominios se ha puesto a prueba la efectividad de las reglas de correlación y los modelos de detección de anomalías valorando cuantitativamente el desempeño para cada uno de los dominios a través de unos parámetros establecidos.

No obstante, durante el desarrollo del trabajo se han podido llegar a las siguientes conclusiones en relación con los beneficios y desventajas de sendos modelos de detección.

Las reglas de correlación son más efectivas identificando patrones de amenazas conocidas, debido a su naturaleza son fáciles de entender debido a que son un conjunto de reglas lógicas por lo que los analistas pueden entender a simple vista su funcionamiento y comportamiento, entre las desventajas que presenta este

tipo de método encontramos la poca flexibilidad que presentan estas reglas debido a que están basadas en condiciones predefinidas lo que provoca que sean menos flexibles que los modelos de detección, también requieren perfiles expertos que definan, creen y mantengan estas reglas ya que las condiciones lógicas plasman el conocimiento experto de estos perfiles.

Los modelos de detección, por el contrario presentan beneficios tales como la flexibilidad dado que estos están condicionados por los datos específicos del dominio de información sobre el cual se ha creado el modelo permitiendo que se adapten mejor que las reglas de correlación, otra ventaja reseñable pasa por la sencillez de implementar detecciones para eventos maliciosos complicados ya que los modelos detectan patrones anómalos y no simplemente condiciones, como desventaja cabe remarcar la dificultad de entendimiento que pueden presentar a los analistas ya que las alertas producidas por estos sistemas están basadas en complejas relaciones matemáticas que no pueden ser fácilmente interpretadas por los humanos.

Teniendo en cuenta todo lo expuesto anteriormente se puede aseverar que se han cumplido tanto los objetivos académicos de investigación como los específicos de implementación de la solución propuesta.

5.2 Trabajos futuros

En lo relativo a líneas de trabajo futuras cabe la posibilidad de ahondar en los diferentes métodos y algoritmos de machine learning para el desarrollo de modelos y enfocar el trabajo a la mejora de resultados y la creación de modelos específicos para cada dominio de monitorización.

En el presente trabajo únicamente se ha explorado la viabilidad y uso de modelos de detección de anomalías utilizando el algoritmo de la función de densidad de probabilidad categorizado dentro de los algoritmos de detección de anomalías, no obstante, existen otras muchas categorías de algoritmos como el grupo de algoritmos de *clustering*, el grupo de algoritmos de regresiones y otros muchos más cada uno de ellos con beneficios para solventar determinados tipos de problemas.

De igual modo en líneas de trabajo futuras se podría explorar la conexión de esta propuesta con sistemas SOAR para la automatización y la orquestación de la respuesta ante determinados eventos de seguridad, pudiendo aplicar lógica adicional más allá de la generación de alertas a cada patrón malicioso detectado.

6. Glosario

GDPR:. General Data Protection Regulation
HDFS:. Hadoop Distributed File System
MLTK:. Machine Learning Toolkit
ONU:. Organización de las Naciones Unidas
SDL:. Security Data Lake
SIEM:. Security Information and Event Magnagement
SOAR:. Security Orchestration and Automation Response
SPL:. Splunk Processing Language

7. Bibliografía

1. Rawat DB, Doku R, Garuba M. Cybersecurity in Big Data Era: From Securing Big Data to Data-Driven Security. IEEE Trans Serv Comput. 2021;14(6):2055–72.
2. PCI Security Standars Council. PCI DSS Effective Daily Log Monitoring [Internet]. 2016 [cited 2022 Oct 2]. Available from: https://listings.pcisecuritystandards.org/documents/Effective-Daily-Log-Monitoring-Guidance.pdf
3. Marty R. The Security Data Lake [Internet]. 2015 [cited 2022 Oct 2]. Available from: https://iotiran.com/wp-content/uploads/2021/04/security-data-lake.pdf
4. Miller D, Harris S, Harper A, VanDyke S, Blask C. Security Information and Event Management (SIEM) Implementation. 2010.
5. Exabeam. Rules vs Models in your SIEM [Internet]. 2018 [cited 2022 Oct 8]. Available from: https://www.exabeam.com/siem/siem-threat-detection-rules-or-models/
6. Cinque M, Cotroneo D, Pecchia A. Challenges and Directions in Security Information and Event Management (SIEM). In: Proceedings - 29th IEEE International Symposium on Software Reliability Engineering Workshops, ISSREW 2018. Institute of Electrical and Electronics Engineers Inc.; 2018. p. 95–9.
7. Panther Labs. Reducing Costs by Moving to a Cloud-Native SIEM [Internet]. 2022 [cited 2022 Oct 9]. Available from: https://panther.com/resources/whitepapers/reducing-cost-by-moving-to-a-cloud-native-siem/
8. Λάλος Δ, Lalos D. Analysis on Security Orchestration Automation and Response (SOAR) platforms for Security Operation Centers. 2022 Aug 29 [cited 2022 Oct 9]; Available from: https://dione.lib.unipi.gr/xmlui/handle/unipi/14560
9. Organización de las Naciones Unidas. Infraestructura - Desarrollo Sostenible [Internet]. 2022 [cited 2022 Oct 9]. Available from: https://www.un.org/sustainabledevelopment/es/infrastructure/
10. Baum D. Deploying a Modern Security Data Lake [Internet]. 2022. Available from: http://oreilly.com
11. Yousef Haik, Tamer M. Shahin. Engineering Design Process Second Edition. Stamford; 2010.
12. Kotenko IV, Saenko IB. Creating new-generation cybersecurity monitoring and management systems. Her Russ Acad Sci. 2014;84(6):424–31.
13. Ullah F, Ali Babar M. Architectural Tactics for Big Data Cybersecurity Analytics Systems: A Review. Journal of Systems and Software. 2019 May 1;151:81–118.
14. Oracle. The Evolution of Big Data [Internet]. 2022 [cited 2023 Jan 6]. Available from: https://www.oracle.com/es/a/ocom/docs/big-data/big-data-evolution.pdf
15. Hadoop. Apache Hadoop [Internet]. 2022 [cited 2022 Dec 6]. Available from: https://hadoop.apache.org/
16. Microsoft. Azure Monitor overview - Azure Monitor | Microsoft Learn [Internet]. 2022 [cited 2022 Nov 6]. Available from: https://learn.microsoft.com/en-us/azure/azure-monitor/overview

17. Microsoft. Understand the Kusto Query Language statement structure - Training | Microsoft Learn [Internet]. 2022 [cited 2022 Nov 6]. Available from: https://learn.microsoft.com/en-us/training/modules/construct-kusto-query-language-statements/2-understand-kusto-query-language-statement-structure
18. Mehta D. Splunk Certified Study Guide. Splunk Certified Study Guide. Apress; 2021.
19. Exabeam. Data Lake Administration Guide [Internet]. 2020 [cited 2022 Nov 6]. Available from: www.exabeam.com/.
20. Microsoft. Detección de un ataque avanzado de varias fases en Microsoft Sentinel | Microsoft Learn [Internet]. 2022 [cited 2022 Oct 10]. Available from: https://learn.microsoft.com/es-es/azure/sentinel/fusion
21. Sachdeva GS. Practical ELK Stack. Practical ELK Stack. Apress; 2017.
22. Patel AB, Birla M, Nair U. Addressing big data problem using Hadoop and Map Reduce. In: 3rd Nirma University International Conference on Engineering, NUiCONE 2012. 2012.
23. Chen Zhong, John Yen, Peng Liu, Robert F. Erbacher. Automate Cybersecurity Data Triage by Leveraging Human Analysts' Cognitive Process [Internet]. 2016 [cited 2022 Nov 8]. Available from: https://ieeexplore.ieee.org/document/7502316
24. Halder S, Ozdemir S. Hands-on machine learning for cybersecurity : safeguard your system by making your machines intelligent using the Python ecosystem.
25. Amazon. Qué es un ataque DDOS [Internet]. 2022 [cited 2023 Jan 6]. Available from: https://aws.amazon.com/es/shield/ddos-attack-protection/
26. Splunk. System requirements for use of Splunk Enterprise on-premises - Splunk Documentation [Internet]. 2022 [cited 2022 Dec 3]. Available from: https://docs.splunk.com/Documentation/Splunk/latest/Installation/Systemrequirements
27. Splunk. Install on Linux - Splunk Documentation [Internet]. 2022 [cited 2022 Dec 3]. Available from: https://docs.splunk.com/Documentation/Splunk/9.0.2/Installation/InstallonLinux
28. Suricata. Suricata - IDS [Internet]. 2022 [cited 2022 Dec 3]. Available from: https://suricata.io/
29. Zeek. About Zeek — Book of Zeek [Internet]. 2022 [cited 2022 Dec 3]. Available from: https://docs.zeek.org/en/master/about.html
30. Splunk. Understanding SPL syntax - Splunk Documentation [Internet]. 2022 [cited 2022 Dec 4]. Available from: https://docs.splunk.com/Documentation/Splunk/9.0.2/SearchReference/UnderstandingSPLsyntax
31. Zeek. Log Files — Book of Zeek [Internet]. 2022 [cited 2022 Dec 4]. Available from: https://docs.zeek.org/en/master/script-reference/log-files.html
32. Splunk. Algorithms in the Machine Learning Toolkit - Splunk Documentation [Internet]. 2022 [cited 2022 Dec 4]. Available from: https://docs.splunk.com/Documentation/MLApp/5.3.3/User/Algorithms#DensityFunction
33. Splunk. Search commands for machine learning - Splunk Documentation [Internet]. 2022 [cited 2022 Dec 4]. Available from:

https://docs.splunk.com/Documentation/MLApp/5.3.3/User/Customsearchcommands#fit_command
34. Besalú M, Àngel M, Gil J, Carles E, Escofet R. Probabilidad y variables aleatorias. 2020.
35. Akamai. Infection Monkey | Akamai [Internet]. 2022 [cited 2022 Dec 6]. Available from: https://www.akamai.com/infectionmonkey
36. Carnegie Mellon University. CERT LiFTeR - The Linux Forensics Tools Repository [Internet]. 2022 [cited 2022 Dec 4]. Available from: https://forensics.cert.org/

8. Anexos

8.1 ANEXO I – Instalación de Splunk

La instalación de la herramienta Splunk se realiza a través de línea de comandos en los diferentes sistemas Linux, el propio fabricante facilita un enlace de descarga mediante el cual podemos obtener la versión adecuada para el entorno.

Lo primero que realizaremos será la descarga en todos los host

```
wget -O splunk-9.0.1-82c987350fde-Linux-x86_64.tgz
"https://download.splunk.com/products/splunk/releases/9.0.1/linux/splunk-
9.0.1-82c987350fde-Linux-x86_64.tgz"
```

Descomprimiremos el contenido del fichero en la carpeta /opt

```
tar xvzf splunk-9.0.1-82c987350fde-Linux-x86_64.tgz -C /opt
```

Configuramos el usuario administrador

```
cd /opt/splunk/bin/
./splunk start

This appears to be your first time running this version of Splunk.

Splunk software must create an administrator account during startup.
Otherwise, you cannot log in.
Create credentials for the administrator account.
Characters do not appear on the screen when you type in credentials.

Please enter an administrator username: splunk

Password must contain at least:
* 8 total printable ASCII character(s).
Please enter a new password: password
```

Configuración de Splunk como servicio para que inicie automáticamente en el arranque de la máquina

```
cd /opt/splunk/bin/
./splunk enable boot-start
```

8.2 ANEXO II – Configuración Security Data Lake

Para establecer la configuración del security data lake deberemos realizar la conexión entre los diferentes servidores, lo primero que realizaremos será la conexión del elemento Search Head con el elemento Indexer, para ello accederemos al menú de búsqueda distribuida en el servidor con rol Search Head y dentro de el al menú de *Search peers* que es otra designación para los servidores con rol de Indexer, en esta pantalla cumplimentaremos los datos del host que actuará de Indexer almacenando todos los datos del security data lake.

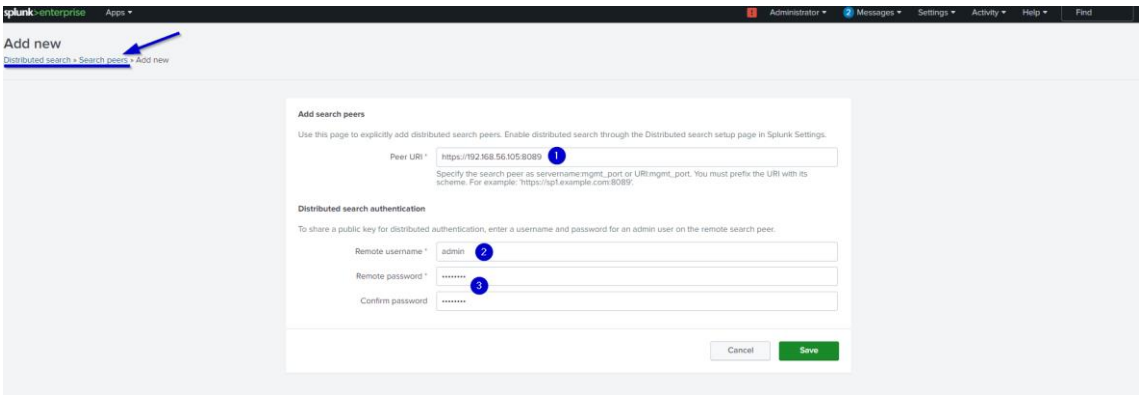


Figura 27: Menú de Distributed search/Search peers

Una vez terminada la configuración podremos realizar búsquedas desde el servidor con el rol de Search head en todo el dominio de los datos almacenados en los servidores con rol de Indexers.

Continuaremos con la configuración del servidor Indexer para que este pueda recibir datos desde el servidor con rol Forwarder, por lo que deberemos acceder al menú de configuración de reenvío y recibimiento y entrar en el submenú de recibir datos, allí configuraremos un puerto donde recibiremos eventos de las instancias configuradas para ello.

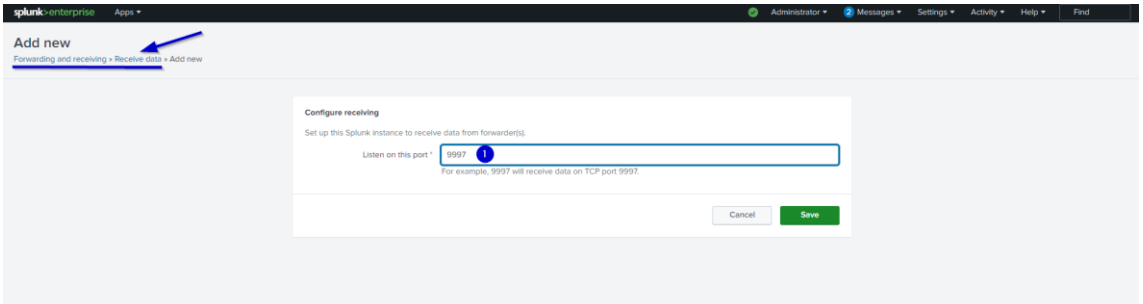


Figura 28: Menú Forwarding and receiving/Receive data

Por último, faltaría realizar la configuración del servidor con el rol de Forwarder, este host del sistema security data lake será el encargado de recibir y reenviar los eventos, para configurar el reenvío deberemos acceder al menú de reenvío y recibimiento y en el submenú de reenvío de datos configurando la dirección IP del servidor con el rol de Indexer.

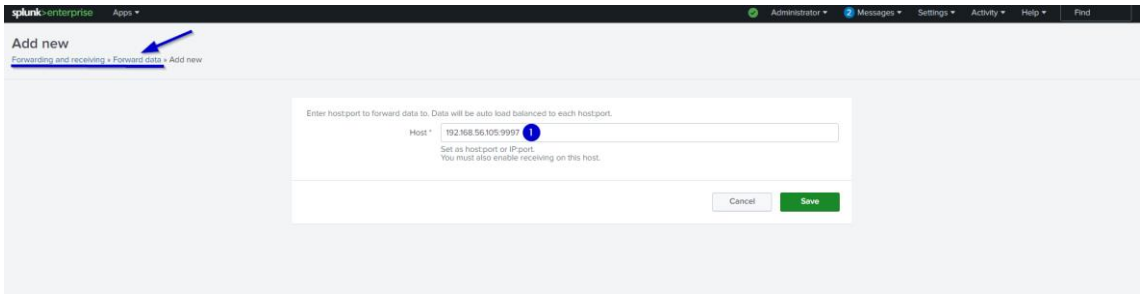


Figura 29: Menú de Forwarding and receiving/Forward data

Por último, es necesario realizar la configuración de los índices para cada dominio de datos que ingresen a el sistema, esta configuración se realizará en el servidor con el rol de Indexer ya que allí es donde reside la información. A través del menú de configuración en el submenú de índices tendremos la oportunidad de crear los índices necesarios para segregar la información del despliegue.

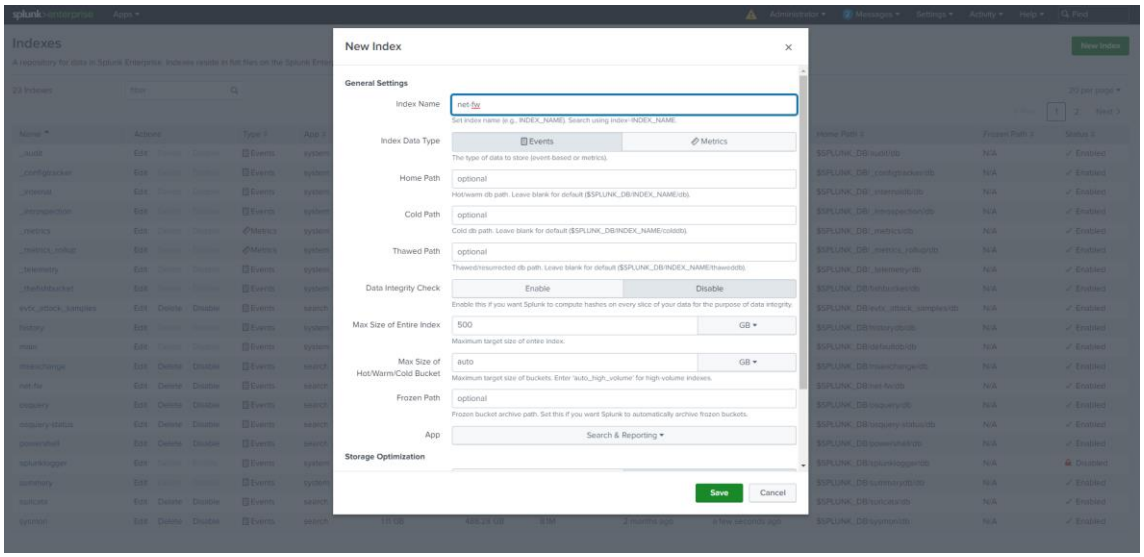


Figura 30: Menú de configuración de índices

8.3 ANEXO III – Configuración detección de red

La configuración llevada a cabo para la detección en el dominio de red ha sido la siguiente, primero se ha establecido la configuración en el sistema de recolección, Splunk dispone de la posibilidad de configurar puertos TCP o UDP para recibir eventos sin mayor problema, esta configuración se realiza desde el menú de entradas de datos.

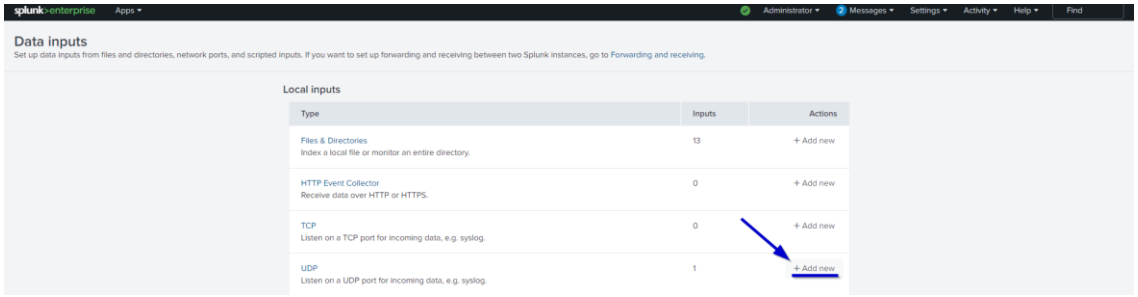


Figura 31: Menú Data inputs

Aquí estableceremos el puerto donde queremos recibir la información, así como los diversos parámetros que ofrece Splunk para almacenar los datos, una vez configurada la parte de recepción en el security data lake se procederá a la configuración necesaria en el elemento de red para realizar el envío de los eventos.

La primera actividad de configuración que llevaremos a cabo dentro del cortafuegos será establecer un puerto como SPAN, este tipo de puertos reenvían todo el tráfico que recibe el cortafuegos por un único puerto, para reproducir este comportamiento crearemos una interfaz tipo Bridge en el menú de interfaces donde designaremos las interfaces desde las cuales replicará el tráfico y la interfaz por la cual saldrá ese tráfico

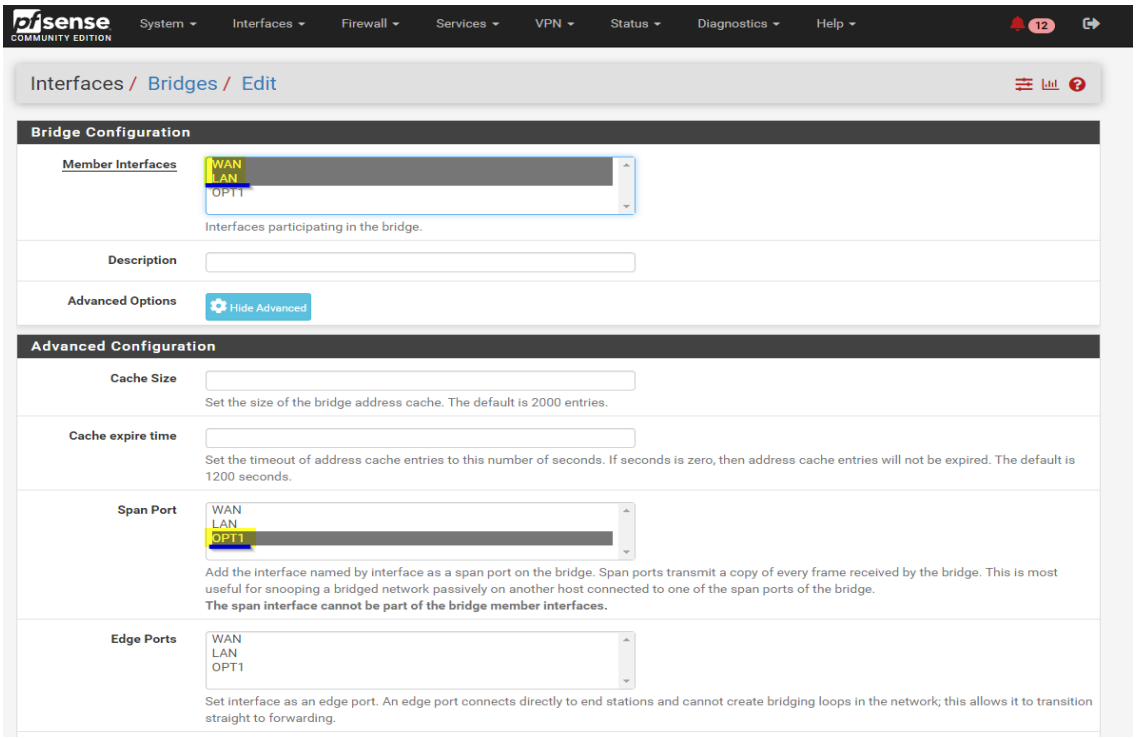


Figura 32: Menú de interfaces en Pfsense

Configurada esta parte, deberemos ir al menú de configuración de los eventos del sistema dentro del menú de estado, aquí seleccionaremos la opción de enviar todos los mensajes de log y estableceremos la dirección IP y el puerto configurado previamente en el receptor.

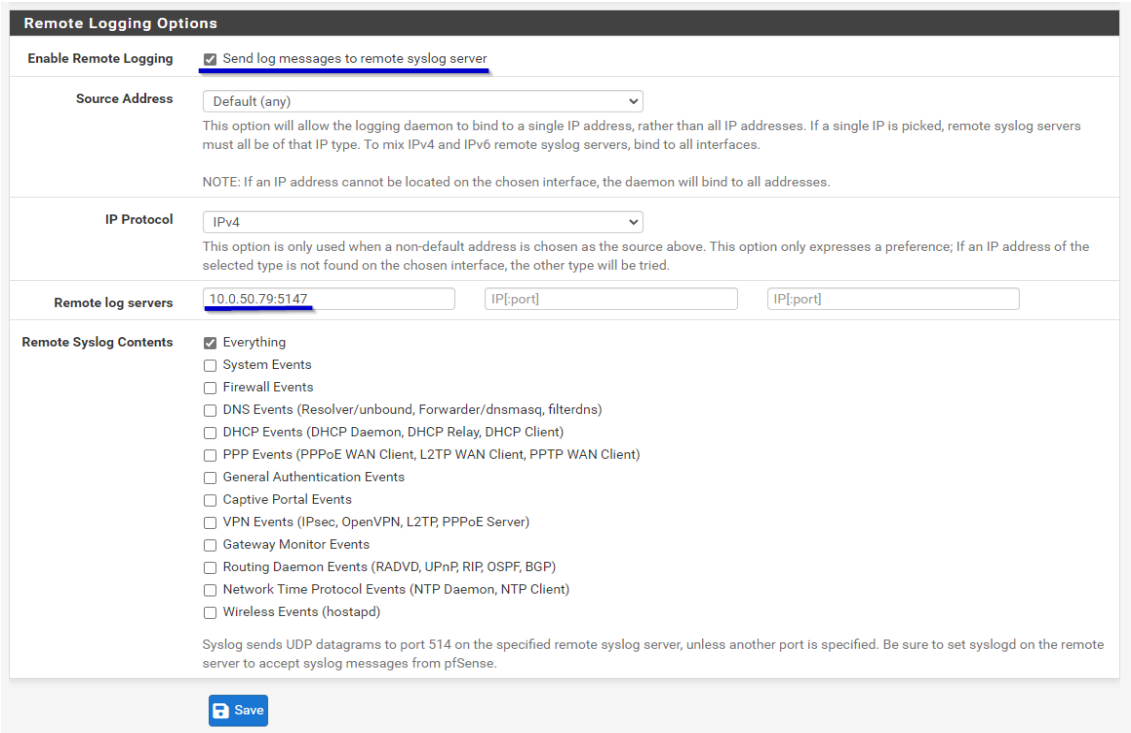


Figura 33: Menú de configuración de eventos del sistema

Continuaremos con la instalación de Zeek y Suricata en el host SPL-FW, estas aplicaciones serán las encargadas de recolectar y analizar el tráfico de red proveniente de la interfaz de SPAN.

Lo primero que se llevará a cabo será la instalación de las dependencias necesarias para facilitar el trabajo obtendremos los paquetes de instalación de zeek para el sistema operativo CentOS 7 del repositorio de herramientas forenses de Carneige Mellon University [36].

```
yum -y install epel-release
yum -y install centos-release-scl-rh
yum -y install zeek
yum -y install suricata
```

Antes de dar comienzo con la configuración de las herramientas se debe proceder a la activación de la interfaz encargada de recolectar el tráfico de red en modo promiscuo, para ello se debe de ejecutar el siguiente comando.

```
ip link set dev ens224 promisc on
```

Después se procederá a la configuración de los ficheros, en el caso de zeek se modificará el archivo node.cfg para establecer la interfaz definida anteriormente como promiscua y se pondrá en marcha la herramienta mediante la herramienta de línea de comandos zeekctl.