

# ***Desplegar la herramienta "Zeek IDS"*** **y su posterior explotación para el análisis de actividades sospechosas en la red**

**Adelino Manuel Paulo Carvalho**

Máster Universitario en Seguridad de las Tecnologías de la Información  
y de las Comunicaciones

TFM - Análisis de Datos

**Borja Guaita Perez**

**Helena Rifà Pous**

01/06/2020



Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-SinObraDerivada [3.0 España de Creative Commons](https://creativecommons.org/licenses/by-nc-nd/3.0/es/)

## FICHA DEL TRABAJO FINAL

<b>Título del trabajo:</b>	<i>Desplegar la herramienta "Zeek IDS" y su posterior explotación para el análisis de actividades sospechosas en la red</i>
<b>Nombre del autor:</b>	<i>Adelino Paulo Carvalho</i>
<b>Nombre del consultor/a:</b>	<i>Borja Guaita Perez</i>
<b>Nombre del PRA:</b>	<i>Helena Rifà Pous</i>
<b>Fecha de entrega (mm/aaaa):</b>	06/2020
<b>Titulación::</b>	<i>Máster Universitario en Seguridad de las Tecnologías de la Información y de las Comunicaciones</i>
<b>Área del Trabajo Final:</b>	<i>Análisis de Datos</i>
<b>Idioma del trabajo:</b>	<i>Castellano</i>
<b>Palabras clave</b>	<i>Zeek-ELK, SIEM, Cyber-Operations</i>
<p><b>Resumen del Trabajo (máximo 250 palabras):</b> <i>Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.</i></p>	
<p>La finalidad de este trabajo es desplegar un sistema de detección de intrusiones y otros eventos anómalos en la red. El objetivo es detectar anomalías en tiempo real pero también disponer de una base de datos de información histórica para la investigación de ataques, intrusiones y sus consecuencias y gravedad.</p> <p>Se despliega Zeek-IDS como sensor de captura de trafico de red y Elasticsearch como base de datos de información y esta combinación se ha mostrado de enorme potencial y versatilidad.</p> <p>Por un lado, se consigue la captura pasiva de la información de los protocolos de red y, de esta forma, sin impacto en las operaciones y, por otro, se asegura su almacenamiento en un sistema escalable y flexible que permite acceso a la información en tiempo real, su procesamiento y análisis, pero también, el acceso a algoritmos de Machine Learning para detección de anomalías de forma desatendida.</p> <p>Hoy, la gestión continua de eventos ha sustituido el análisis de respuesta a incidentes, de ayer, y la solución desplegada, mediante Kibana y Elastic SIEM proporcionan el interfaz, herramientas y funcionalidades necesarios para la monitorización de la actividad de red y gestión de incidentes.</p> <p>La configuración de reglas de detección y también de ElastAlert, como sistema de notificación, ha creado un entorno óptimo para supervisión de la red de forma autónoma y con gestión integrada de la investigación de incidentes.</p>	

**Abstract (in English, 250 words or less):**

The purpose of this work is to deploy an intrusion detection system and other anomalous events on the network. The objective is to detect anomalies in real-time but also to have a database of historical information for the investigation of attacks, intrusions, and their consequences and severity.

Zeek-IDS is deployed as a network traffic capture sensor and Elasticsearch as an information database and this combination has shown enormous potential and versatility.

On the one hand, passive capture of information from network protocols is achieved and, thus, without impact on operations and, on the other, its storage is ensured in a scalable and flexible system that allows access to information in real-time, its processing and analysis, but also, access to machine learning algorithms for detecting anomalies unattended.

Today, continuous event management has replaced yesterday's incident response analysis, and the solution deployed, using Kibana and Elastic SIEM, provides the interface, tools and functionalities necessary for monitoring network activity and incident management.

The configuration of detection rules and also of ElastAlert, as a notification system, has created an optimal environment for autonomous network monitoring and with integrated management of incident investigation.

# Índice

<b>1. Introducción.....</b>	<b>1</b>
<b>1.1 Contexto y justificación del Trabajo .....</b>	<b>1</b>
<b>1.2 Objetivos del Trabajo .....</b>	<b>2</b>
<b>1.3 Enfoque y método seguido .....</b>	<b>3</b>
<b>1.4 Planificación del Trabajo .....</b>	<b>5</b>
<b>1.5 Análisis de Riesgos.....</b>	<b>8</b>
<b>1.6 Estado del Arte .....</b>	<b>10</b>
<b>1.7 Breve resumen de productos obtenidos.....</b>	<b>14</b>
<b>1.8 Breve descripción de los otros capítulos de la memoria .....</b>	<b>14</b>
<b>2. Diseño y Construcción .....</b>	<b>16</b>
<b>2.1 - Definición de Arquitectura de Red.....</b>	<b>18</b>
<b>2.1.1 – Diagrama de red .....</b>	<b>21</b>
<b>2.1.2 – Configuración de mirroring de tráfico de red .....</b>	<b>22</b>
<b>2.2 - Instalación y Configuración Debian .....</b>	<b>24</b>
<b>2.3 - Instalación y Configuración Zeek .....</b>	<b>26</b>
<b>2.4 - Instalación y Configuración ELK .....</b>	<b>32</b>
<b>2.5 - Pruebas de Monitorización con Zeek .....</b>	<b>33</b>
<b>2.6 – Configuración de Filebeat con modulo Zeek .....</b>	<b>35</b>
<b>2.7 - Pruebas de Visualización en ELK.....</b>	<b>37</b>
<b>2.8 – Análisis de Listas de Reputación.....</b>	<b>38</b>
<b>2.9 – Configuración de listas de reputación - IntelStack .....</b>	<b>41</b>
<b>3. Implementación SIEM .....</b>	<b>45</b>
<b>3.1 – Elastic SIEM .....</b>	<b>51</b>
<b>3.2 – Ingesta del log de pfSense en Elasticsearch .....</b>	<b>57</b>
<b>3.3 – Yara, Intelligence-Driven Incident Response .....</b>	<b>58</b>
<b>3.4 – Elastalert, procesamiento de alertas en tiempo real .....</b>	<b>65</b>
<b>4. Pruebas de validación.....</b>	<b>69</b>
<b>5. Conclusión.....</b>	<b>72</b>
<b>6. Bibliografía.....</b>	<b>76</b>
<b>7. Anexos 1 – Instalación ELK.....</b>	<b>77</b>

<b>7.1 - Instalación y configuración de Elasticsearch.....</b>	<b>77</b>
<b>7.2 – Configuración de Seguridad TLS.....</b>	<b>78</b>
<b>7.3 – Configuración de claves de usuarios de sistema.....</b>	<b>80</b>
<b>7.4 – Instalación y Configuración de Kibana .....</b>	<b>81</b>
<b>7.5 – Instalación y Configuración de Logstash .....</b>	<b>83</b>
<b>7.6 – Instalación y Configuración de Filebeat System .....</b>	<b>85</b>
<b>7.7 – Instalación y Configuración de Metricbeat .....</b>	<b>86</b>
<b>7.8 – Instalación y Configuración de Packetbeat y HTTP endpoint .....</b>	<b>90</b>
<b>7.9 – Instalación ELK – proceso alternativo y actualización de versión</b>	<b>93</b>
<b>8. Anexos 2 – Instalación YARA.....</b>	<b>94</b>
<b>8.1 - Instalación y configuración de Yara .....</b>	<b>94</b>
<b>8.2 – Yara, configuración de integración con Zeek .....</b>	<b>95</b>
<b>8.3 – Yara, automatización mediante script de Python.....</b>	<b>97</b>
<b>8.4 – Yara, compilación de reglas .....</b>	<b>98</b>

## Lista de figuras

Ilustración 1 - Planificación del Proyecto	5
Ilustración 2 - Planificación: Fase 1	6
Ilustración 3 - Planificación: Fase 2	6
Ilustración 4 - Despliegue de prueba de ELK en VM	7
Ilustración 5 - Planificación: Fase 3	7
Ilustración 6 - Planificación: Fase 4	8
Ilustración 7 - Soluciones IDS	11
Ilustración 8 - Elastic Stack	13
Ilustración 9 - Laboratorio de Seguridad VirtualBox	16
Ilustración 10 - Aplicaciones de la solución	17
Ilustración 11 - Diagrama de red	19
Ilustración 12 - Red Laboratorio Seguridad	19
Ilustración 13 - Diagrama de red	21
Ilustración 14 - Configuración de red VM pfSense	21
Ilustración 15 - Configuración de red VM zeek	22
Ilustración 16 - pfSense configuración de Interfaces - antes	22
Ilustración 17 - pfSense añadir Bridge	23
Ilustración 18 - pfSense configuración de Interfaces - después	23
Ilustración 19 - Cockpit interface	24
Ilustración 20 - /etc/nftables.conf	25
Ilustración 21 - características de las VMs	26
Ilustración 22 - /etc/network/interfaces	27
Ilustración 23 - /opt/zeek/etc/node.cfg	29
Ilustración 24 - /opt/zeek/etc/networks.cfg	29
Ilustración 25 - validación funcionalidad GeolIP	31
Ilustración 26 - prueba de instalación con query JQ	31
Ilustración 27 - ELK diagrama de flujo	32
Ilustración 28 - ZeekControl, verificación de estado de proceso	34
Ilustración 29 - Logs Zeek	34
Ilustración 30 - Logs Zeek ejemplo	35
Ilustración 31 - /etc/filebeat/modules.d/zeek.yml (parcial)	36
Ilustración 32 - /etc/filebeat/filebeat.yml	36
Ilustración 33 - Kibana, Zeek dashboard	37
Ilustración 34 - OTX AlienVault	38
Ilustración 35 - IntelStack, suscripción	39
Ilustración 36 - criterios de ordenación de listas	39
Ilustración 37 - IntelStack, comparativo de metricas	40
Ilustración 38 - IntelStack, mantenimiento de listas	40
Ilustración 39 - IntelStack master-public.dat	43
Ilustración 40 - IntelStack prueba de detección	43
Ilustración 41 - 2019 Data Breach Investigations Report	46
Ilustración 42 - The Cyber Kill Chain, example	49
Ilustración 43 - SIEM Process	50
Ilustración 44 - Security Information and Event Management Capabilities	50
Ilustración 45 - Elastic SIEM dashboard	51
Ilustración 46 - Elastic SIEM, Signal Detection Rules	52

Ilustración 47 - Regla de detección: VNC from Internet	53
Ilustración 48 - Regla de detección, MITRE ATT&CK information	53
Ilustración 49 - Creación de una regla de detección	54
Ilustración 50 - Reglas de detección propias	54
Ilustración 51 - Activación de la regla: IntelStack - detección de Dominio	55
Ilustración 52 - Elastic SIEM, rule action (ver 7.7)	56
Ilustración 53 - Connections of EMOTET, DRIDEX, URSNIF and BitPaymer	56
Ilustración 54 - Detección de acceso SSH	57
Ilustración 55 - pfSense, Kibana dashboard	58
Ilustración 56 - yaramatch.py	60
Ilustración 57 - Estructura del log de detecciones Yara	61
Ilustración 58 - Aviso de anomalía en index pattern	62
Ilustración 59 - Resultado de la ingesta del log de Yara en Elasticsearch	62
Ilustración 60 - yaraZProcessor.py, insertLog	63
Ilustración 61 - yaraProcessor.py, contexto	64
Ilustración 62 - extract-some-files.zeek	64
Ilustración 63 - Ejemplo de regla Yara, pdf_contains_link.yar	65
Ilustración 64 - ElastAlert, notificación por e-mail (parcial)	68
Ilustración 65 - SIEM Detections, signals	69
Ilustración 66 - SIEM Detections, Yara Rules - PoetRAT	70
Ilustración 67 - Kibana, Dashboard Yara	70
Ilustración 68 - Kibana, Dashboard IntelStack	71
Ilustración 69 - Zeek, atributos Intel	71
Ilustración 70 - /usr/lib/systemd/system/elasticsearch.service	77
Ilustración 71 - /etc/elasticsearch/elasticsearch.yml	77
Ilustración 72 - Elasticsearch, autenticación de usuario	80
Ilustración 73 - ELK, usuarios de sistema	81
Ilustración 74 - Elasticsearch, configuración	81
Ilustración 75 - Kibana, configuración de conexión	82
Ilustración 76 - Kibana, acceso seguro	83
Ilustración 77 - Kibana, portada	83
Ilustración 78 - Logstash, configuración pipeline	84
Ilustración 79 - /etc/filebeat/filebeat.yml	85
Ilustración 80 - /etc/metricbeat/metricbeat.yml	87
Ilustración 81 - /etc/metricbeat/modules.d/elasticsearch-xpack.yml	88
Ilustración 82 - /etc/metricbeat/modules.d/elasticsearch-xpack.yml	88
Ilustración 83 - Dashboard, Stack Monitoring	89
Ilustración 84 - /etc/metricbeat/modules.d/kibana-xpack.yml	89
Ilustración 85 - Dashboard, Elasticsearch	90
Ilustración 86 - /etc/packetbeat/packetbeat.yml	90
Ilustración 87 - comprobación del HTTP endpoint	91
Ilustración 88 - /etc/metricbeat/modules.d/beat-xpack.yml	91
Ilustración 89 - script de extracción de ficheros	95
Ilustración 90 - Yara, modificación en local.zeek	96
Ilustración 91 - Validación de reglas Yara	97
Ilustración 92 - YaraProcessor.py opción de uso de reglas compiladas	99



# 1. Introducción

## 1.1 Contexto y justificación del Trabajo

La sociedad actual es una sociedad digital, conectada y a cada día la dependencia en las infraestructuras de comunicaciones y sistemas informáticos es más importante. La disponibilidad de los servicios de uso diario como los transportes o el comercio son esenciales y son soportados por sistemas informáticos sobre los cuales se asume un determinado grado de seguridad, pero la cuantificación de este grado de seguridad es prácticamente imposible en el modelo actual.

Los servicios críticos como hospitales, seguridad y emergencias, entre otros, comparten en muchos casos infraestructura informática y de comunicaciones con los restantes agentes por la necesidad de acceso, colaboración y transferencia de información.

En ambos escenarios se encuentran además los usuarios que requieren acceso simplificado a los recursos, en muchos casos mediante sus propios dispositivos (BYOD), sobre los cuales no existe un control sistemático de las medidas de seguridad aplicadas y las aplicaciones instaladas.

Por último, se puede identificar también un crecimiento acelerado de lo que se conoce como la *Industry 4.0* y toda una nueva gama de dispositivos (IoT) de bajo coste y que pueden ser fácilmente desplegados en cualquier red introduciendo graves problemas de privacidad y seguridad.

Por un lado, la dependencia de la sociedad del correcto funcionamiento de los sistemas de información y, por otro, el aumento constante del valor de la información que se produce y se guarda en esos sistemas, crea un entorno susceptible a ataques que se aprovechan de toda esa superficie expuesta, referida anteriormente.

Los *cyberattacks* no se reducen más a intentos de acceso aleatorios a sistemas que están indebidamente protegidos, aunque estos sigan existiendo, pero se presentan como objetivos direccionados, con dedicación de medios humanos y materiales para alcanzarlos.

La sofisticación de los ataques que las redes informáticas y de comunicaciones sufren actualmente requieren más que dispositivos que filtran el tráfico en función de reglas conocidas o de sus origen y destino.

Los costes asociados a las intrusiones en sistemas informáticos pueden asumir valores significativos porque tienen la capacidad de afectar negativamente los tres vectores de la seguridad Informática: Confidencialidad, Integridad y Disponibilidad. Por lo tanto, la implementación de una solución con capacidad de detectar estas intrusiones es crítica, no solo para la organización que la implementa como también para las demás organizaciones del ecosistema con las cuales colabora.

La realización de este TFM pretende aportar una solución a este problema, mediante el uso de la herramienta – Zeek IDS. Zeek es una herramienta *open-source* que aporta grande flexibilidad a la hora de analizar el tráfico de red.

Zeek es extensible y programable mediante un modelo de programación *event-driven* de tal forma que es posible analizar determinado tipo de tráfico sin necesidad de capturar masivamente todos los datos de la red.

La definición y activación de scripts en función de las necesidades es una de las capacidades que se pretenden explotar facilitando su implementación en escenarios de menor complejidad.

## **1.2 Objetivos del Trabajo**

El objetivo principal del TFM, lo cual define su alcance, se encuentra determinado por el título y descripción del trabajo a realizar. Se considera como objetivos a conseguir

- Desplegar la herramienta "Zeek IDS"
- Su explotación para el análisis de actividades sospechosas en la red

Cierto de que el segundo punto solo es posible habiendo alcanzado el primero, pero también de que la realización del primero sin el segundo no sería más que un tutorial de instalación de la herramienta.

La materialización de los objetivos definidos se conseguirá mediante la consecución de las actividades listadas a continuación:

- Determinación de la infraestructura de red y del sistema donde se desplegará Zeek.
- Instalación y configuración del Sistema Operativo con especial enfoque en la seguridad en el entorno de despliegue. Por familiaridad con la distribución y por considerarla una de las más seguras y con más soporte se optará por Debian.
- Instalación y configuración de Zeek como IDS
- Instalación y configuración de ELK (Elasticsearch, Logstash, Kibana) de forma a poder representar visualmente las notificaciones generadas por Zeek para los eventos de interés.
- Integración de la solución con listas de reputación de "*malware/botnets*" de forma a lograr alertas de actividad sospechosa
- Pruebas de funcionalidad y de integración de la solución
- Elaboración de documentación de soporte a la solución

La investigación hecha hasta el momento permite concluir que Zeek además de IDS puede, mediante frameworks, como es la NetControl Framework, actuar de forma activa sobre los dispositivos de la red, es una perspectiva que se muestra interesante sobre todo para entornos con dispositivos IoT pero que probablemente no será posible explotar por restricciones de tiempo.

Un subobjetivo, con motivación profesional, es la posibilidad de desplegar Zeek en una unidad central y sensores en redes periféricas de acceso limitado sobre las cuales no se tiene conocimiento de los equipos y sus configuraciones de seguridad. Mediante el uso de Zeek Agent Framework se ve interesante la posibilidad de desplegar agentes en equipos de interés. La posibilidad de incluir este subobjetivo en el ámbito del proyecto dependerá del progreso en los objetivos principales.

### **1.3 Enfoque y método seguido**

Este proyecto se inicia con el objetivo de desplegar la herramienta Zeek, anteriormente conocida como Bro (Bro-IDS), como solución para la monitorización y análisis de un entorno de red heterogéneo. Para demostrar el

proceso de implementación y explotación de la solución se crea un entorno virtual al que se hará referencia como laboratorio.

Se considera la opción del laboratorio de seguridad, como una forma de abstraer el entorno, sobre el cual se pretende desplegar la solución y poder presentar los ejemplos de explotación de funcionalidades sin necesidad de contextualizarlo en el ambiente empresarial.

La selección de Zeek y también del entorno Elasticsearch, Logstash y Kibana se debe a que son herramientas open-source, con una amplia base de usuarios y muy buena documentación. La integración con listas de reputación se hará mediante IntelStack, un producto también open-source mediante el cual se pueden suscribir listas y que está debidamente integrado con Zeek.

En esta primera fase se ha procedido a una instalación básica de las herramientas para una primera toma de contacto y sobre todo a consulta de la información principal para evaluar la complejidad del proyecto.

El desarrollo inicial de la solución se hará en un entorno virtual, usando para virtualización un sistema de VirtualBox, bien para los equipos donde se configuran Zeek y ELK, como para los equipos que van a simular la red que se pretende supervisar.

En una fase más avanzada del proyecto, si es viable por el estado de desarrollo, se pretende conectar la maquina Zeek a una red en modo IDS usando Tap con un Switch Gigabit que permite Port-Mirroring (TP-Link TL-SG108E).

En el apartado 2 se presentará una descripción sumaria de la infraestructura de red y de las características de los equipos, así como las opciones de configuración en cada caso y los escenarios alternativos.

La complejidad y las dificultades que puedan surgir en la programación de los scripts van a determinar si es posible acometer las tareas en el entorno real, por ejemplo, mediante el despliegue de un agente.

## 1.4 Planificación del Trabajo

Como forma de simplificar la representación del trabajo mediante el gráfico de Gantt se ha optado por considerar 'días de trabajo' y no especificar las horas que efectivamente se podrán/tendrán que dedicar para su realización.

Se ha mantenido la configuración estándar de Microsoft Project, considerando solo los días de lunes a viernes y tampoco se han excluido los días festivos que corresponden en el periodo de realización del trabajo y que son 5.

La planificación presentada es la *baseline* del proyecto sobre la cual se pueden producir algunas modificaciones como resultado de la fase inicial en que se realiza una actividad exploratoria y de descubierta más firme de las funcionalidades del software.

En las entregas posteriores se mantendrá esta planificación como base y las modificaciones, si las hay, se presentarán como estado del proyecto. La fecha final del proyecto es fija, por lo tanto, cualquier retraso temporal, no asumible por la holgura entre tareas se tendrá que solucionar aumentando las horas de dedicación en los días disponibles o recortando la duración de las tareas afectas.

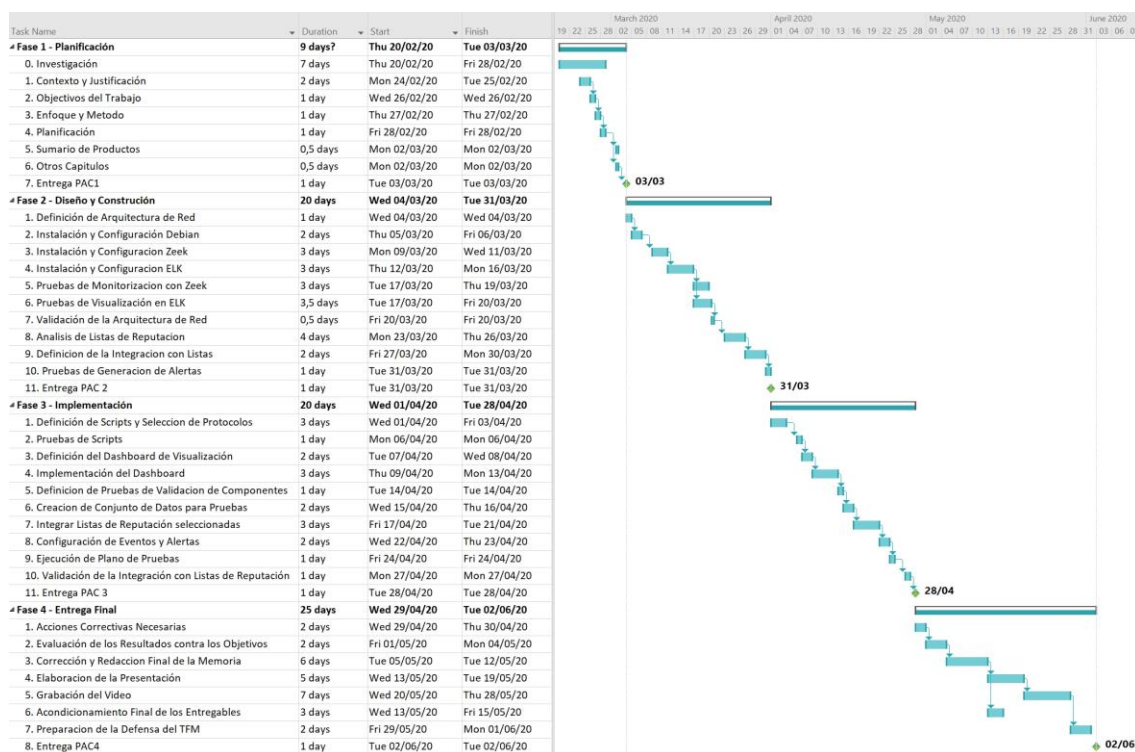


Ilustración 1 - Planificación del Proyecto

En la primera fase, que termina con la entrega de este documento, se han realizado estas tareas, sobre todo de investigación y planificación.

<b>4 Fase 1 - Planificación</b>	<b>9 days?</b>	<b>Thu 20/02/20</b>	<b>Tue 03/03/20</b>
0. Investigación	7 days	Thu 20/02/20	Fri 28/02/20
1. Contexto y Justificación	2 days	Mon 24/02/20	Tue 25/02/20
2. Objetivos del Trabajo	1 day	Wed 26/02/20	Wed 26/02/20
3. Enfoque y Metodo	1 day	Thu 27/02/20	Thu 27/02/20
4. Planificación	1 day	Fri 28/02/20	Fri 28/02/20
5. Sumario de Productos	0,5 days	Mon 02/03/20	Mon 02/03/20
6. Otros Capítulos	0,5 days	Mon 02/03/20	Mon 02/03/20
7. Entrega PAC1	1 day	Tue 03/03/20	Tue 03/03/20

**Ilustración 2 - Planificación: Fase 1**

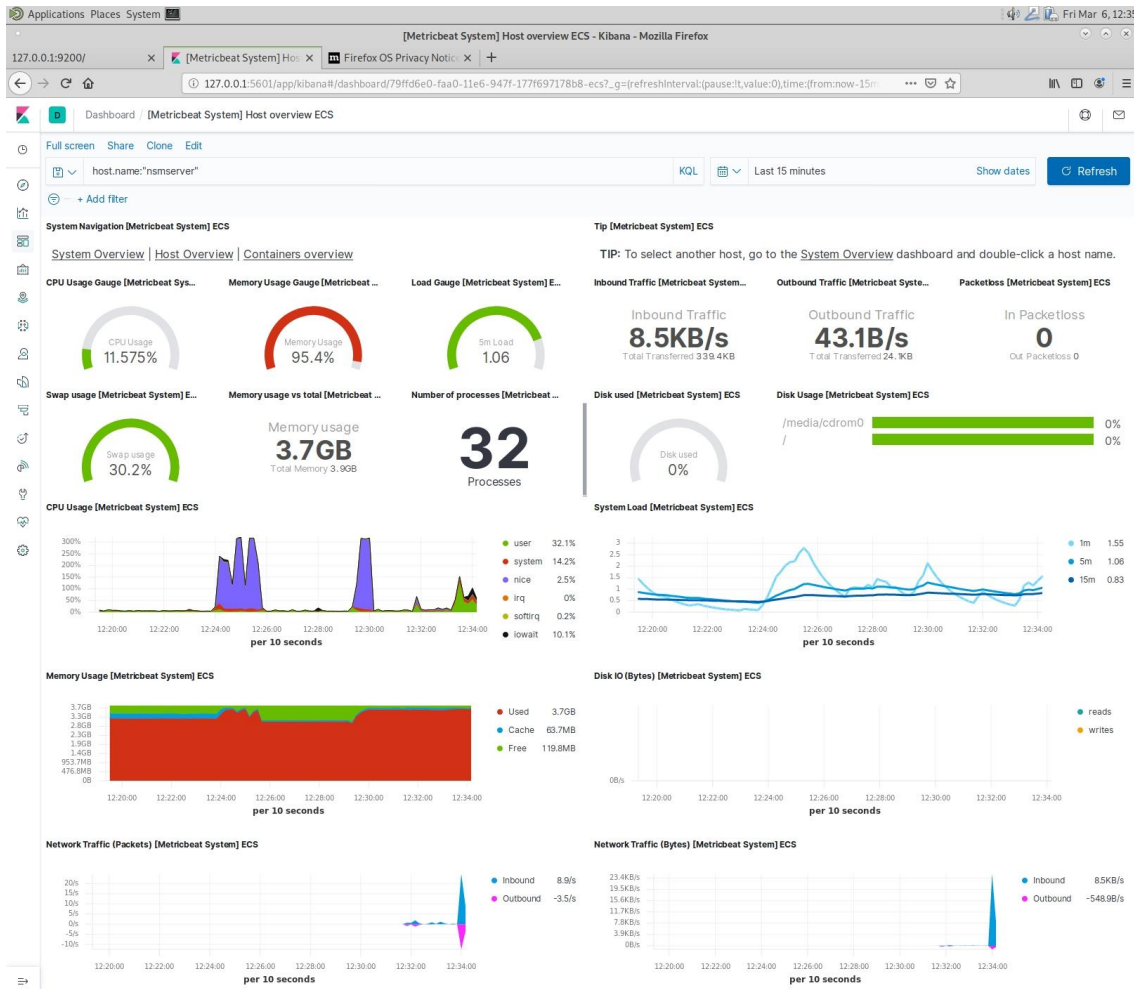
Como punto de partida, se ha procedido a la instalación de los elementos principales, en una única máquina virtual, con excepción del agregador de listas de reputación. Se ha podido verificar que la instalación y configuración son un proceso manual que implica la modificación de opciones en diversos ficheros de opciones, por lo cual, requiere una planificación previa.

Se ha verificado también que, aun partiendo del desconocimiento de las aplicaciones, se ha podido lograr su configuración y puesta en marcha, pues existe documentación disponible y es actualizada y útil.

A partir de aquí se va a crear el entorno virtual donde se podrán ejecutar las tareas de preparación para la implementación.

<b>4 Fase 2 - Diseño y Construcción</b>	<b>20 days</b>	<b>Wed 04/03/20</b>	<b>Tue 31/03/20</b>
1. Definición de Arquitectura de Red	1 day	Wed 04/03/20	Wed 04/03/20
2. Instalación y Configuración Debian	2 days	Thu 05/03/20	Fri 06/03/20
3. Instalación y Configuración Zeek	3 days	Mon 09/03/20	Wed 11/03/20
4. Instalación y Configuración ELK	3 days	Thu 12/03/20	Mon 16/03/20
5. Pruebas de Monitorización con Zeek	3 days	Tue 17/03/20	Thu 19/03/20
6. Pruebas de Visualización en ELK	3,5 days	Tue 17/03/20	Fri 20/03/20
7. Validación de la Arquitectura de Red	0,5 days	Fri 20/03/20	Fri 20/03/20
8. Análisis de Listas de Reputación	4 days	Mon 23/03/20	Thu 26/03/20
9. Definición de la Integración con Listas	2 days	Fri 27/03/20	Mon 30/03/20
10. Pruebas de Generación de Alertas	1 day	Tue 31/03/20	Tue 31/03/20
11. Entrega PAC 2	1 day	Tue 31/03/20	Tue 31/03/20

**Ilustración 3 - Planificación: Fase 2**



**Ilustración 4 - Despliegue de prueba de ELK en VM**

En la fase de implementación se supone que toda la infraestructura está funcionando correctamente y que es conforme a la arquitectura definida.

4 Fase 3 - Implementación	20 days	Wed 01/04/20	Tue 28/04/20
1. Definición de Scripts y Selección de Protocolos	3 days	Wed 01/04/20	Fri 03/04/20
2. Pruebas de Scripts	1 day	Mon 06/04/20	Mon 06/04/20
3. Definición del Dashboard de Visualización	2 days	Tue 07/04/20	Wed 08/04/20
4. Implementación del Dashboard	3 days	Thu 09/04/20	Mon 13/04/20
5. Definición de Pruebas de Validación de Componentes	1 day	Tue 14/04/20	Tue 14/04/20
6. Creación de Conjunto de Datos para Pruebas	2 days	Wed 15/04/20	Thu 16/04/20
7. Integrar Listas de Reputación seleccionadas	3 days	Fri 17/04/20	Tue 21/04/20
8. Configuración de Eventos y Alertas	2 days	Wed 22/04/20	Thu 23/04/20
9. Ejecución de Plano de Pruebas	1 day	Fri 24/04/20	Fri 24/04/20
10. Validación de la Integración con Listas de Reputación	1 day	Mon 27/04/20	Mon 27/04/20
11. Entrega PAC 3	1 day	Tue 28/04/20	Tue 28/04/20

**Ilustración 5 - Planificación: Fase 3**

Al finalizar la fase 3 los objetivos tendrán que estar cumplidos con excepción de algunas correcciones o cambios que se requieran, por lo tanto, la fase 4

consistirá en preparar la memoria final y también la presentación y video de defensa del TFM.

4 Fase 4 - Entrega Final	25 days	Wed 29/04/20	Tue 02/06/20
1. Acciones Correctivas Necesarias	2 days	Wed 29/04/20	Thu 30/04/20
2. Evaluación de los Resultados contra los Objetivos	2 days	Fri 01/05/20	Mon 04/05/20
3. Corrección y Redaccion Final de la Memoria	6 days	Tue 05/05/20	Tue 12/05/20
4. Elaboracion de la Presentación	5 days	Wed 13/05/20	Tue 19/05/20
5. Grabación del Video	7 days	Wed 20/05/20	Thu 28/05/20
6. Acondicionamiento Final de los Entregables	3 days	Wed 13/05/20	Fri 15/05/20
7. Preparacion de la Defensa del TFM	2 days	Fri 29/05/20	Mon 01/06/20
8. Entrega PAC4	1 day	Tue 02/06/20	Tue 02/06/20

**Ilustración 6 - Planificación: Fase 4**

En esta fase se realizarán un conjunto de pruebas formales que se considerarán como pruebas de aceptación.

### 1.5 Análisis de Riesgos

Este proyecto se hará en simultaneo con la realización de la actividad profesional a tiempo completo, en un entorno de trabajo en base a proyectos para clientes y que requiere flexibilidad y adaptación en función de sus necesidades.

Es importante considerar que los diversos factores identificados y teniendo en cuenta que el tiempo es un recurso finito y limitado se pueden producir desvíos puntuales en las tareas programadas.

Se procurará hacer los ajustes necesarios para que, en escenarios menos favorables, se puedan cumplir los hitos de los entregables definidos y respetar la planificación.

A continuación, se identifican dos tipos de riesgos que se pueden materializar durante la realización del proyecto y se procuran formas de mitigar su impacto. Por un lado, los riesgos derivados de la disponibilidad temporal y de su impacto en las tareas programadas. Por otro, los riesgos técnicos resultantes de la indisponibilidad de los recursos tecnológicos con los cuales se propone hacer la implementación de la solución.



Como forma de mitigar los riesgos resultantes de la disponibilidad de dedicación al proyecto, se hace una planificación con holgura suficiente para permitir alguna volatilidad. Se dará precedencia a la realización de las tareas de implementación e integración sobre aquellas de preparación de la memoria, aunque el impacto puede ser evidente en la medida que la memoria es la medida del avance del proyecto.

En lo relacionado con los riesgos de tipo técnico, más allá de todos aquellos que están relacionados con los suministros y acceso a los dispositivos, que se dan por garantizados, se consideran dos casos:

Falla técnica de los dispositivos usados para implementar el entorno virtual y viabilidad del conjunto de datos generados durante el proyecto incluyendo la memoria.

Se pretende realizar el proyecto sobre un equipo Dell XPS 8930, que aún no es un recurso disponible al inicio del proyecto. Dadas sus características – Intel Core i7 9700 de 9.<sup>a</sup> generación (8 núcleos, 12 MB de caché), 32 GB DDR4 y discos SSD – es un equipo bien dimensionado para la creación del entorno planteado para el proyecto.

En su defecto se cuenta con un equipo portátil, Dell Inspiron 7559 – Intel Core i7 6700HQ (4 núcleos, 6 MB de caché), 16 GB DDR3, con discos SSD. La principal dificultad además de la limitación de memoria es la cantidad de espacio disponible que exigirá dimensionar las máquinas de forma menos óptima.

Con el objetivo de minimizar el impacto de esta circunstancia se cuenta también con un sistema remoto virtualizado y con la posibilidad de añadir máquinas temporales si se requiere.

Por viabilidad del conjunto de datos se entiende en este contexto toda la infraestructura no física, desde las máquinas virtuales hasta la documentación generada. El riesgo principal es el de pérdida de la información o bien por problemas de los dispositivos o a consecuencia de las pruebas que se pretenden hacer con algún malware.

Las medidas de mitigación de este riesgo en relación con la documentación pasan por definir un plan de backup diario de la información en cloud, además de un backup periódico en dispositivo externo. En relación a las máquinas virtuales se creará una máquina virtual base con la configuración considerada ideal para desplegar los servidores de la solución. A partir de ella se clonarán las VMs de Zeek y de ELK. Se guardarán snapshots periódicos para preservar el estado y poder retroceder si necesario. Los backups se harán solo en dispositivos externos que se dejarán desconectados del equipo.

Además de estos riesgos existen tareas de difícil planificación pues su complejidad no es conocida antes de acometerlas, se debe a que el conocimiento de algunas de las aplicaciones y herramientas es casi inexistente, por lo tanto, además del tiempo requerido para el despliegue se requiere tiempo de formación y práctica.

## **1.6 Estado del Arte**

Existe un objetivo común que es mantener el sistema de información seguro y detectar los intentos de acceso indebido, además de permitir hacer un análisis forense en el caso de que la intrusión no pueda evitarse.

Existen diversas soluciones para lograr estos objetivos las cuales, por un lado, siguen distintas estrategias y, por otro, implementan diferentes tipos de licencias.

En el primer caso, podemos distinguir entre soluciones que se aplican a toda la red o a un host en concreto o, también, por la forma por la cual evalúan el tráfico, bien en función de firmar o a partir de detección de anomalías.

En relación con el tipo de licencia se puede, de forma genérica, considerar por un lado las soluciones comerciales y, por otro, las soluciones de código abierto. La tabla siguiente presenta un elenco de algunas de las soluciones disponibles.

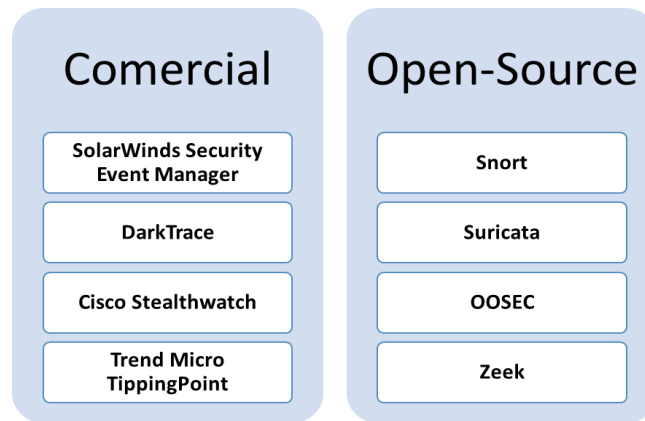


Ilustración 7 - Soluciones IDS

En relación con los sistemas comerciales se destacan estas características:

**SolarWinds Security Event Manager**<sup>1</sup> es sobre todo un HIDS (host-based intrusion detection system) pero es también un sistema de gestión de amenazas que puede gestionar y procesar logs de otras herramientas, por ejemplo, de Snort.

Se caracteriza por disponer de una base de datos de amenazas y en cierta medida puede considerarse que es también un IPS pues se puede configurar para que responda de forma automática a eventos.

**DarkTrace**<sup>2</sup> es casi exclusivamente un HIDS, se caracteriza por su sistema de autoaprendizaje basado en AI. La solución define un baseline del sistema a proteger y detecta los casos anómalos en comparación con él. Su principal característica es que no depende de reglas o firmas predefinidas por lo cual puede detectar un ataque que aún no se hubiera verificado nunca.

**Cisco Stealthwatch**<sup>3</sup> es también un HIDS basado en detección por autoaprendizaje, en este caso mediante machine-learning. El modelo es también de detención de anomalías.

**Trend Micro TippingPoint**<sup>4</sup> es un IDS que se basa en el análisis de tráfico con vistas a la detección de malware, vulnerabilidades y exploits. Es también una

<sup>1</sup> <https://www.solarwinds.com/security-event-manager?CMP=ORG-BLG-DNS>

<sup>2</sup> <https://www.darktrace.com/en/products/enterprise/>

<sup>3</sup> <https://www.cisco.com/c/en/us/products/security/stealthwatch/index.html>

solución que implementa el modelo de auto respuesta por lo cual puede ser considerado en alguna medida un sistema IPS.

En relación con las soluciones Open-Source presentadas:

**Snort**<sup>5</sup> es un sistema de captura de tráfico de red, logging y detección de intrusiones. Es un NIDS (Network Intrusion Detection System). Es un sistema basado en reglas que compara el tráfico con firmas conocidas.

**Suricata**<sup>6</sup> es también un NIDS basado en reglas. Permite además la definición de alertas y filtros y puede complementar con funcionalidades de IPS. Permite inspección de protocolos y detección de anomalías basadas en reglas y patrones. Es además un sistema multihilo por lo cual permite inspeccionar grandes cantidades de tráfico con pequeña latencia.

**OSSEC**<sup>7</sup> es un HIDS basado en políticas caracterizado por crear una baseline o snapshot del sistema y después compara cambios con esas firmas de forma a inferir el tipo de cambios.

**Zeek**<sup>8</sup> es un NIDS basado en eventos para monitorización de la red. Por un lado, Zeek hace el logging del tráfico de red y, por otro, hace su análisis. Zeek opera al nivel de aplicación y transforma el tráfico en eventos y después procede a su análisis en función de las reglas definidas.

Mas allá de la configuración básica, Zeek permite programación mediante scripts, los cuales permiten generar alertas en función de detención de anomalías, firmas y análisis de procesos. La opción en cuanto al IDS a usar es Zeek porque es un sistema open-source, es programable mediante scripts y sobre el cual existe documentación extensa y una base de usuarios activa.

---

<sup>4</sup>[https://www.trendmicro.com/en\\_us/business/products/network/intrusion-prevention/tipping-point-threat-protection-system.html](https://www.trendmicro.com/en_us/business/products/network/intrusion-prevention/tipping-point-threat-protection-system.html)

<sup>5</sup> <https://www.snort.org/>

<sup>6</sup> <https://suricata-ids.org/>

<sup>7</sup> <https://www.ossec.net/>

<sup>8</sup> <https://zeek.org/>

Zeek almacena los datos en ficheros de log, los cuales pueden ser configurados, pero no dispone de interfaz para visualización de la información. Para cumplir ese requisito se usará Elastic Stack.

El almacenamiento de los logs resultado de la captura de Zeek se envían a **Elasticsearch** que es una solución que también dispone de una licencia Open-Source además de una licencia comercial y de una opción en Cloud. **Elastic**<sup>9</sup> es un stack, es decir, es un ecosistema de aplicaciones que se usan de forma combinada para construir una solución escalable y dinámica.

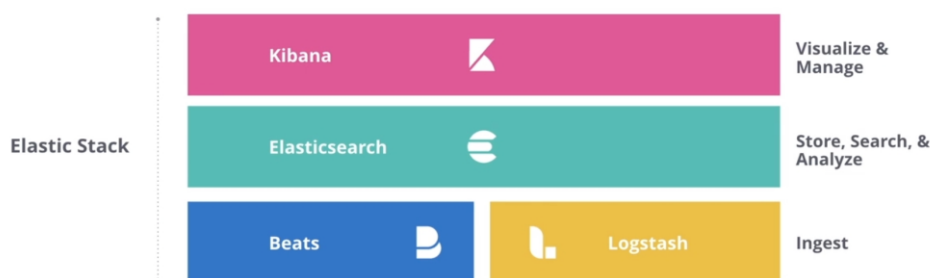


Ilustración 8 - Elastic Stack

**Elasticsearch** es el repositorio de la información que se pretende analizar. **Beats** y **Logstash** son los elementos mediante los cuales se inserta la información en Elasticsearch cuyo proceso se describe como ingesta de datos.

**Kibana** es la plataforma de visualización, la cual permite usar los índices de Elasticsearch para la construcción de visualizaciones y cuadros de mando. Existen otras alternativas como podrían ser Splunk, DataLog, GrayLog y Logz.io pero Elastic Stack es la solución con mejor relación entre funcionalidades y requerimientos.

La optimización del sistema de IDS se logrará mediante la utilización de listas de reputación. La solución elegida es **IntelStack**<sup>10</sup> de la cual más adelante se describe el proceso de suscripción (gratuito) del servicio, así como la configuración del sensor que se será incorporado a Zeek.

---

<sup>9</sup> <https://www.elastic.co/>

<sup>10</sup> <https://intelstack.com/>

Se procurará también añadir al sistema la posibilidad de análisis de los ficheros que pueden ser extraídos por Zeek, para ello se estudiará la viabilidad de integrar con Zeek las reglas **YARA**<sup>11</sup>

### **1.7 Breve resumen de productos obtenidos**

Los productos obtenidos serán la memoria final del proyecto y los anexos. Se preparará también una presentación que servirá de base al video, lo cual es también un entregable. Las máquinas virtuales que se van a construir no se consideran como entregables pues son un recurso que se quedará desactualizado a corto plazo y el documento ofrecerá todos los pasos para su correcta configuración.

Del punto de vista técnico de la solución, se dispondrá de un sistema IDS implementado usando la aplicación Zeek e sus módulos. Por las opciones tomadas en la definición del proyecto se tratará de una solución modular, versátil y expandible.

La arquitectura propuesta aísla el sensor de tráfico (Zeek) de la plataforma de almacenamiento y visualización de la información (ELK), por lo tanto, por un lado, se podrán añadir uno o varios sensores adicionales a la solución y, por otro, se podrá si se pretende enviar los logs a una solución Elastic en Cloud.

### **1.8 Breve descripción de los otros capítulos de la memoria**

Además de este capítulo de Introducción, el segundo capítulo: Diseño y Construcción, describirá la toma de decisión por la cual se decide el modelo de implementación que se desarrollará y se aportarán los pasos necesarios para la configuración de las herramientas a usar en la solución.

También en el capítulo 2 se hará un análisis y resumen de las listas de reputación disponibles y se justificará la selección hecha y se definirá como integrar las listas con el resto de solución. Además, se realizarán pruebas para verificar que cada uno de los componentes está configurado y operativo.

---

<sup>11</sup> <https://virustotal.github.io/yara/>

En el capítulo 3 se procede a la implementación de la solución de tal forma que sea posible cumplir la segunda parte del objetivo la “explotación para el análisis de actividades sospechosas en la red”.

Se procederá a la selección de los protocolos a monitorizar y de los scripts a usar. También se presentará el *dashboard* que permitirá visualizar los alertas generados. El objetivo último es generar alertas en función de determinados eventos anómalos en la actividad de red.

La conclusión incluirá el resultado de la evaluación de los resultados obtenidos en función de las pruebas realizadas y su adecuación o no a los objetivos planteados.

## 2. Diseño y Construcción

La implementación de la solución se hace en un entorno simulado en el cual toda la infraestructura es virtual, el conjunto de todos los elementos es considerado a efectos de este trabajo como laboratorio de seguridad. Se ha elegido la plataforma de virtualización VirtualBox<sup>12</sup>.

Para reproducir este laboratorio se considera como requisito la instalación y configuración de una plataforma de virtualización, pero no necesariamente la que se ha utilizado pues no existe cualquier dependencia de ella.



Ilustración 9 - Laboratorio de Seguridad VirtualBox

El escenario principal requiere 4 máquinas virtuales, representadas en la ilustración anterior. Se despliega **pfSense**<sup>13</sup> como router y firewall para permitir crear las redes necesarias y mediante el cual se procederá a la duplicación de tráfico que será pasado a Zeek para su captura y análisis.

Este proceso de mirroring se hace mediante la configuración del *span* entre dos interfaces y se documentará adelante. La configuración de seguridad del router pfSense está fuera del alcance de este documento y solo se indicarán las acciones necesarias para lograr el objetivo.

Zeek y ELK son 2 máquinas virtuales desplegadas en Debian<sup>14</sup> 10 (“buster”) y las opciones de configuración del SO se presenta en el punto 2.2. Las opciones

---

<sup>12</sup> <https://www.virtualbox.org/>

<sup>13</sup> <https://www.pfsense.org/>

<sup>14</sup> <https://www.debian.org/>



de configuración son comunes a ambos servidores. Win7 será la maquina cliente para ejecución de las pruebas no se documenta su instalación porque representa cualquier puesto de usuario.

Además del sistema operativo los servidores zeek y elk requieren la instalación y configuración de las aplicaciones seleccionadas para la implementación del proyecto. De forma simplificada, la ilustración siguiente representa las aplicaciones requerías y el flujo principal de información.



**Ilustración 10 - Aplicaciones de la solución**

La simplificación de la relación entre los componentes deja de fuera alguna complejidad del proceso, pero se procederá a una presentación más específica en el capítulo de configuración de Zeek y ELK.

Se tiene que Zeek procede a la captura del tráfico y lo guarda en ficheros de log, uno por cada protocolo que se pretende controlar; mediante la integración con IntelStack, se hace un análisis sobre ese mismo tráfico, generando un archivo de log adicional como resultado de la evaluación del tráfico en función de las listas de reputación.

Los ficheros de log son procesados por componentes, los Beats. Estos módulos se usarán no solo para procesar los logs del IDS pero también para supervisión del propios hosts recabando métricas e información analítica.

La información colectada por los beats, se envía directamente a Elasticsearch. Mediante LogStash se puede procesar los logs, procediendo al filtrado de los datos y también a la normalización de aquellos si se requiere, pero Filebeats cuenta con un soporte muy completo a los logs de Zeek si estos se configuran para que se guarden en formato Json. Elasticsearch es el repositorio de la información indexada y, en general, los logs colectados por cada Beat son tratados como un índice del cual forman parte una colección de documentos.

Si el desarrollo del proyecto lo permite en términos de tiempo y de capacidad técnica, se pretenden añadir también los logs del firewall pfSense con el objetivo de poder asociar a eventos internos posibles intentos de acceso externos. Y también la ingesta del resultado del análisis de los ficheros que extraídos por Zeek.

En ambos casos se plantea usar Logstash para la ingesta de esos datos. La solución para pfSense es inmediata en la medida que el propio firewall implementa el envío a en formato sysmon, lo cual es reconocido y procesado por Logstash.

En el caso de YARA no parece existir una solución disponible que integre el resultado de la aplicación de las reglas con ELK por lo cual el planteamiento es crear un log que se pueda procesar mediante Logstash.

Por último, se usará Kibana para la presentación y visualización de los datos, generación de cuadros de mando, etc. En este aspecto se dispondrá de cuadros de mando predefinidos, como el SIEM disponible en la instalación base, pero también de visualizaciones y cuadros de mando construidos para representar la información de inteligencia recibida de Zeek e IntelStack.

El conjunto de todas las aplicaciones, su proceso de configuración y automatización es presentado a continuación.

## **2.1 - Definición de Arquitectura de Red**

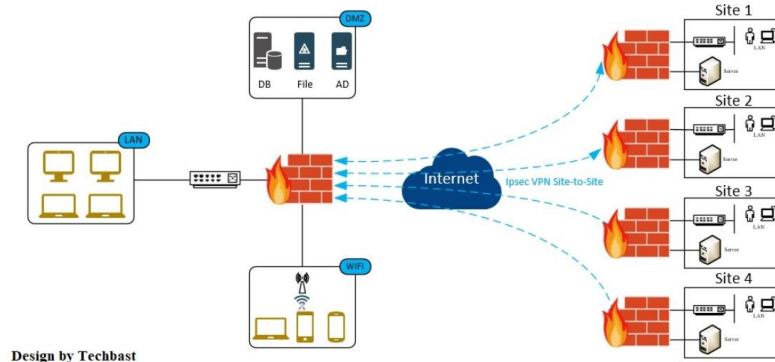
La infraestructura de red elegida es una representación abstracta de una red de mediana complejidad<sup>15</sup>. Representa el escenario de una organización con una sede y unas sucursales que se comunican con aquella mediante VPN *site-to-site*.

La red correspondiente a la sede a su vez se representa como un conjunto de 3 zonas de seguridad, aislando la red local cableada (lan), la red inalámbrica (wifi) y la red donde se despliegan los servidores (dmz), podría además

---

<sup>15</sup><https://techbast.com/2019/05/visio-stencil-model-of-multi-office-connection-system-using-ipsec-vpn-and-vigor-modem.html>

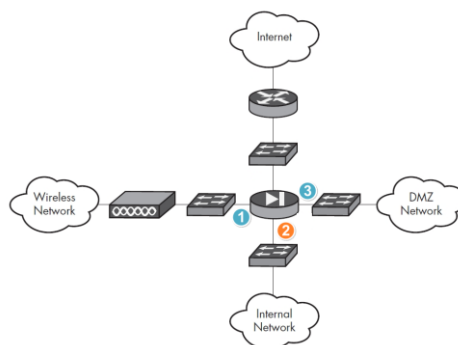
considerarse otra zona donde representar los servidores directamente expuestos a Internet, pero añadiría complejidad y no aportaría ventajas para el escenario a representar.



**Ilustración 11 - Diagrama de red**

Los elementos representados pretenden construir un escenario que, por un lado, permita considerar la existencia de varios sensores y, por otro, la existencia de agentes remotos de forma a adecuarse al alcance del trabajo presentado anteriormente.

El diagrama presentado anteriormente se simplifica mediante la representación siguiente:



**Ilustración 12 - Red Laboratorio Seguridad**

Considerando las zonas de red representadas se identifican 3 puntos donde se pueden localizar idealmente los sensores de IDS. La actividad de red en cada una de las zonas presenta características propias y, por lo tanto, el tipo de riesgos que se deben identificar son distintos.

En la zona 1 (red wifi), es posible que los usuarios puedan conectar sus dispositivos personales y, por lo tanto, hay una serie de riesgos resultantes directamente de las aplicaciones instaladas, aplicación o no de políticas de seguridad y de actualizaciones, etc.

En la zona 2 (red local), los equipos tendrán seguramente una política de seguridad centralizada que asegura la actualización del sistema operativo y de las aplicaciones, antivirus, etc. Los riesgos en esta zona son sobre todo derivados de la acción de los usuarios, por ejemplo, acceso a páginas web o apertura de ficheros recibidos por e-mail, etc.

En la zona 3 (red de servidores), se requiere medidas de seguridad estrictas, de tal forma que el acceso a los equipos sea limitado tanto desde la interfaz de red externa como de red interna. Esta es por lo tanto la zona más segura haciendo de elevada dificultad bien ataques externos como internos.

En el escenario para explotación de la solución propuesta se trabajará en la zona 2 y, por lo tanto, se implementará el sensor 2 del diagrama anterior. El tipo de ataques que se pretenden identificar mediante el sensor 2 son los de *Client-Side Compromise*, en este trabajo se procura seguir el proceso propuesto en el libro *The Practice of Network Security Monitoring*.

*Client-side compromise involves an intruder exploiting an application with which a user interacts. This application could be a web browser, email client, media player, or any other program that users rely on for access to network resources. An attacker might trick a user into visiting a compromised site and revealing her credentials, or he might simply position himself to take advantage of a routine that the user follows.*<sup>16</sup>

A través de la elección de este sensor y la supervisión dinámica de la actividad de los usuarios en esta zona de la red se podrán ejercitar las reglas de detección mediante las listas de reputación que se quieren implementar.

---

<sup>16</sup> The Practice of Network Monitoring, pag 236

## 2.1.1 – Diagrama de red

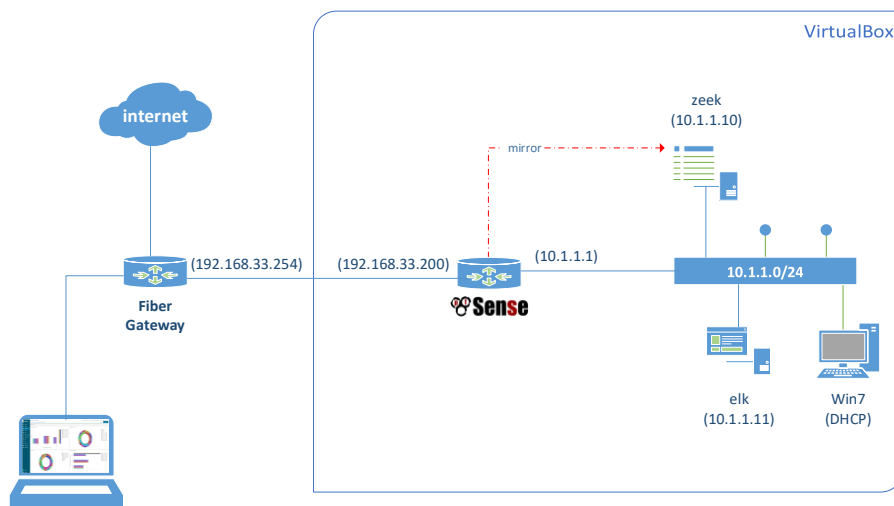


Ilustración 13 - Diagrama de red

La infraestructura virtual se ha configurado con dos redes de distintos rangos de IP. La máquina virtual pfSense se ha desplegado con 3 NICs. El NIC 1 se ha configurado como 'adaptador puente' y se le asigna una IP de la red local (192.168.33.0/24). La IP asignada (192.168.33.200) a la interface vtnet0 es una IP estática, la Gateway de red es 192.168.33.254.

```
FreeBSD/amd64 (pfSense.ampaulo.lab) (ttyv0)
VirtualBox Virtual Machine - Netgate Device ID: e9bfa7fa29a8c5d4f47b
*** Welcome to pfSense 2.4.5-RELEASE (amd64) on pfSense ***
WAN (wan)      -> vtnet0      -> v4: 192.168.33.200/24
                v6: 2001:8a0:5d00:c400:a00:27ff:fef6:ded2/64
LAN (lan)      -> vtnet1      -> v4: 10.1.1.1/24
MIRROR (opt1) -> vtnet2      ->
OPT2 (opt2)   -> bridge0     ->
```

Red	
Adaptador 1:	Red paravirtualizada (Adaptador puente, «Realtek PCIe GBE Family Controller»)
Adaptador 2:	Red paravirtualizada (Red interna, «intnet»)
Adaptador 3:	Red paravirtualizada (Red interna, «mirror»)

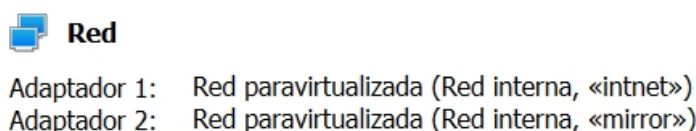
Ilustración 14 - Configuración de red VM pfSense

Los NICs 2 y 3 se han configurado como 'red interna' y a cada uno de ellos se le ha asignado un nombre de red. En VirtualBox el nombre define la red y máquinas con redes internas distintas no pueden verse entre sí.

La red **intnet** corresponde a la red local, será la red definida también en las VMs zeek, elk y win7. En esta red, el interfaz vtnet1 del pfsense tiene la IP

10.1.1.1/24, la cual es la Gateway de red de los demás equipos. La VM zeek tendrá la IP estática 10.1.1.10 y a la VM elk se le asigna la IP 10.1.1.11. Por otro lado, la VM win7 tendrá una IP dinámica asignada por DHCP, se requiere por lo tanto que el pfSense asigne estos IPs, para ello se ha configurado la opción de DHCP server.

La red **mirror** corresponde al interfaz por donde se hará la duplicación del tráfico de forma a que pueda ser capturado por zeek. Esta red corresponde al interfaz de pfsense vtnet2 y será configurada también en la VM zeek.



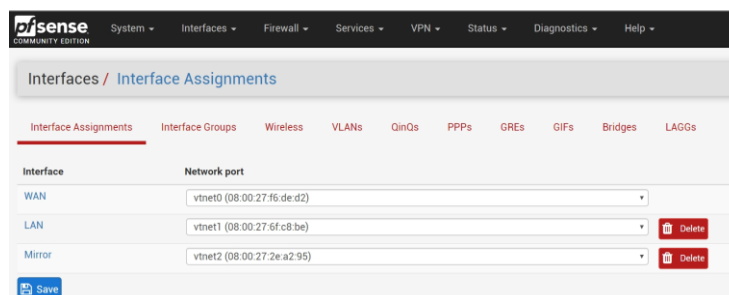
**Ilustración 15 - Configuración de red VM zeek**

Esta red no requiere la asignación de rango de IP pues únicamente se transmite tráfico generado por otros interfaces.

## 2.1.2 – Configuración de mirroring de trafico de red

Se requiere que el router/firewall esté debidamente configurado, se recomienda consultar la documentación en caso de problemas<sup>17</sup>. Efectuadas las pruebas de comunicación desde la interfaz vtnet1 (LAN) hacia internet se puede proceder a la configuración del interfaz vtnet2. Después de hacer login en el **WebConfigurator** accedemos al menú

### Interfaces -> Assignments



**Ilustración 16 - pfSense configuración de Interfaces - antes**

<sup>17</sup> <https://docs.netgate.com/pfsense/en/latest/>

El objetivo de la configuración a realizar es hacer con que el tráfico de la red LAN (vtnet1) se replique en el interfaz Mirror (vtnet2). La única configuración necesaria en el interfaz Mirror es que esté *enabled*. El proceso de mirroring o span del tráfico se hace mediante la creación de un recurso bridge. Para ello accedemos al apartado Bridges del menú Assignments.

## Interfaces -> Assignments -> Bridges

Se añade un nuevo bridge usando el boton **Add** y después en la ventana de configuración se procede a la configuración desplegando opciones con **Display Advanced**. Únicamente hay que seleccionar el interfaz de origen del tráfico y el de destino o donde se pretende que se envíe la copia de este. Se pueden seleccionar más de una interface de origen y se debe tener en cuenta que la interface de destino o span no puede estar seleccionado en las interfaces de origen.

Interfaces / Bridges / Edit

**Bridge Configuration**

**Member Interfaces**  
WAN  
LAN  
MIRROR  
Interfaces participating in the bridge.

**Description**  
Lan Mirror

**Advanced Options** [Hide Advanced](#)

**Advanced Configuration**

**Cache Size**  
Set the size of the bridge address cache. The default is 2000 entries.

**Cache expire time**  
Set the timeout of address cache entries to this number of seconds. If seconds is zero, then address cache entries will not be expired. The default is 1200 seconds.

**Span Port**  
WAN  
LAN  
MIRROR

Add the interface named by interface as a span port on the bridge. Span ports transmit a copy of every frame received by the bridge. This is most useful for snooping a bridged network passively on another host connected to one of the span ports of the bridge. The span interface cannot be part of the bridge member interfaces.

**Ilustración 17 - pfSense añadir Bridge**

Se puede también asignar una descripción y por último, al final de la página se graban los cambios. Después de grabarlos hay que aceptar la notificación de que se necesita aplicar cambios.

pfSense  
System - Interfaces - Firewall - Services - VPN - Status - Diagnostics - Help

Interfaces / Interface Assignments

Interface Assignments Interface Groups Wireless VLANs QinQs PPPs GREs GIFs Bridges LAGs

Interface	Network port
WAN	vtnet0 (08:00:27:f6:de:d2)
LAN	vtnet1 (08:00:27:6f:c8:be) <a href="#">Delete</a>
Mirror	vtnet2 (08:00:27:2e:a2:95) <a href="#">Delete</a>
OPT2	BRIDGED (Lan Mirror) <a href="#">Delete</a>

**Ilustración 18 - pfSense configuración de Interfaces - después**

## 2.2 - Instalación y Configuración Debian

Se ha elegido la distribución Debian para instalar las VMs zeek y elk, la versión a fecha del proyecto es la 10.3.0 codename 'buster'. Se ha optado por la instalación manual pues la utilización de una herramienta de despliegue de infraestructura esta fuera del alcance del proyecto.

Para asegurar que ambas VMs compartan similar configuración y permitir reconstruir el sistema de forma rápida, se ha optado por crear una maquina inicial y después clonarla haciendo las modificaciones a la configuración pertinentes, entre ellas la cantidad de memoria asignada, MAC de la interface de red, etc.

Se ha optado por una instalación minimalista para incluyendo únicamente el servidor de SSH y las utilidades del sistema, disminuyendo de esta forma los recursos necesarios para ejecución y también la superficie de exposición del sistema. Para firewall local se ha optado por **nftables** recomendado por Debian para nuevas instalaciones. Se han instalado además algunos paquetes de utilidades y el sistema de gestión de host – cockpit<sup>18</sup>.

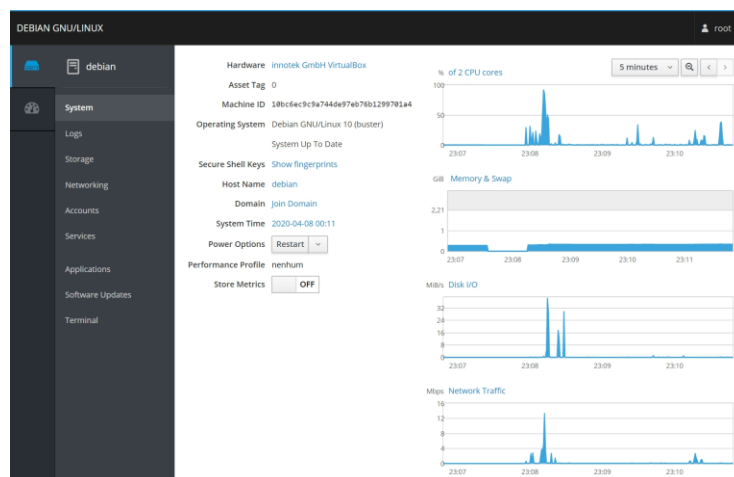


Ilustración 19 - Cockpit interface

La instalación del proyecto cockpit es trivial

```
apt-get install cockpit
```

```
systemctl enable cockpit.socket
```

---

<sup>18</sup> <https://cockpit-project.org/>



Se procede a la instalación y configuración de las reglas del firewall del sistema base, las cuales bloquean todo el tráfico entrante con excepción del ping, ssh y web interface de la solución cockpit. En las VMs zeek y elk se requiere posterior configuración de los puertos de aplicación.

```
apt-get install nftables
systemctl enable nftables
nano /etc/nftables.conf
```

```
#!/usr/sbin/nft -f

flush ruleset

table inet filter {
  chain input {
    type filter hook input priority 0; policy drop;

    ct state invalid counter drop comment "drop invalid packets"
    ct state {established, related} counter accept comment "accept all connections started inside"

    iifname lo accept comment "accept loopback"

    # ping
    ip protocol icmp counter accept comment "accept all icmp types"
    ip6 nexthdr icmpv6 counter accept comment "accept all icmp types"

    # ssh
    tcp dport { 22 } counter log prefix "[nftables]: " accept
    tcp dport { 22 } ct state new limit rate 15/minute accept

    # cockpit
    tcp dport { 9090 } counter log prefix "[nftables]: " accept

    #log flags all counter drop
    log prefix "[nftables] Input Denied: " flags all counter drop
  }
  chain forward {
    type filter hook forward priority 0; policy drop;
    # allow existing connections
    ct state related,established accept
    counter comment "count dropped packets"
  }
  chain output {
    type filter hook output priority 0; policy accept;
    counter comment "count accepted packets"
  }
}
```

#### Ilustración 20 - /etc/nftables.conf

Después de crear las reglas se procede a su validación y en ausencia de errores al reinicio del servicio

```
nft -f /etc/nftables.conf
service nftables restart
```

Además, se ha procedido a instalar las utilidades siguientes:

```
apt-get install git tcpdump ethtool tree htop iotop sysstat curl ntp
dnsutils iptraf-ng netsniff-ng gawk tmux zip unzip
```

y también porque es un requisito para ELK el Java:

```
apt-get install default-jre
```

Se recomienda la instalación de las herramientas de VirtualBox:

```
apt-get update && apt-get upgrade
apt-get install net-tools
```

```

apt-get install build-essential module-assistant dkms
m-a prepare
umount /media/cdrom
mount -o exec /media/cdrom
cd /media/cdrom
sh VBoxLinuxAdditions.run
umount /media/cdrom
reboot

```

Partiendo de este sistema base se han construido 2 VMs con estas características, las cuales junto con la VM pfSense representan la infraestructura minima de red.

	pfSense	zeek	elk
CPU	1	2	2
Memoria	512 Mb	2048 Mb	4096 Mb
Disco	8 Gb	100 GB SSD	500 GB SSD
Red	bridge intnet mirror	intnet mirror	intnet

**Ilustración 21 - características de las VMs**

### 2.3 - Instalación y Configuración Zeek

Se procede a la documentación de la instalación de Zeek y sus dependencias porque es la base del proyecto a acometer. La configuración de Zeek se ha efectuado siguiendo los pasos recomendados en la página de soporte del producto<sup>19</sup>. A continuación, se lista los comandos a destacando únicamente las opciones tomadas que difieren o complementan la documentación.

Se procede a la instalación de las dependencias del proyecto con el objetivo de compilar Zeek a partir de las fuentes ya activar la opción de GeolIP.

```

apt-get install cmake make gcc g++ flex bison libpcap-dev libssl-dev
python-dev swig zlib1g-dev python-ipaddress libmaxminddb-dev sendmail
libgoogle-perftools4 libjemalloc-dev libgeoip-dev

```

---

<sup>19</sup> <https://docs.zeek.org/en/master/install/install.html>

Primeramente, se procede al acondicionamiento de la máquina virtual, en particular a la configuración de la red. La VM zeek tiene dos NICs, la interface de red enp0s3 es la conexión a la red local (LAN) definida en las características de la infraestructura como red 'intnet'. La interface de red enp0s8 es la conexión que recibe el mirroring de los datos de la LAN y es definido en la infraestructura como red 'mirror'. La configuración de red de esta VM queda como sigue:

### nano /etc/network/interfaces

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static

    address 10.1.1.10
    network 10.1.1.0
    netmask 255.255.255.0
    broadcast 10.1.1.255
    gateway 10.1.1.1

# The mirror network interface
auto enp0s8
iface enp0s8 inet manual
    up ifconfig enp0s8 -arp up
    up ip link set enp0s8 promisc on
    down ip link set enp0s8 promisc off
    down ifconfig enp0s8 down
    post-up for i in rx tx sg tso ufo gso gro lro; do ethtool -K enp0s8 $i off; done
    post-up echo 1 > /proc/sys/net/ipv6/conf/enp0s8/disable_ipv6
    # You probably don't need to enable or edit the following setting,
    # but it is included for completeness.
    # Note that 4096 is just an example and your NIC may have a different maximum rx size.
    # To determine the maximum rx setting for your NIC: ethtool -g ethX
    # Also note that increasing beyond the default may result in inconsistent traffic:
    # https://taosecurity.blogspot.com/2019/04/troubleshooting-nsm-virtualization.html
    # post-up ethtool -G enp0s8 rx 4096
```

### Ilustración 22 - /etc/network/interfaces

La interface enp0s8 no requiere la asignación de una dirección IP, es un enlace pasivo que solo escucha y captura el tráfico del resto de los demás nodos de la red. Se activa el modo 'promisc' y se ajustan algunas configuraciones en especial desactivando las funcionalidades de checksum.

La reconfiguración de la interface requiere el utilitario **ethtool** lo cual se ha instalado junto con las utilidades en la configuración de la VM base. El desactivar de las funciones de checksum es recomendado para una correcta captura de los paquetes. La instalación y ejecución de Zeek se hará en un usuario con permisos limitados, pero miembro del grupo sudo. Usuario y password se definen como 'zeek'.

Es posible instalar Zeek desde el repositorio, usando apt-get, por ejemplo. Si se opta por esta vía hay que añadir la localización de los archivos.

```
echo 'deb libjemalloc-  
devhttp://download.opensuse.org/repositories/security:/zeek/Debian_10/  
' > /etc/apt/sources.list.d/security:zeek.list  
wget -nv  
https://download.opensuse.org/repositories/security:zeek/Debian_10/Rele  
ase.key -O Release.key  
apt-key add - < Release.key  
apt-get update  
sudo apt-get install zeek
```

Conforme referido anteriormente este no ha sido el proceso seguido en este trabajo. Se ha optado por la instalación manual de forma a habilitar la opción de GeoIP lo cual requiere compilar la solución con los parámetros oportunos. Se empieza por añadir el usuario/grupo zeek y posteriormente se descargan las fuentes y se prepara el entorno. No se describe el proceso porque es trivial:

```
sudo groupadd zeek  
sudo useradd -m zeek -g zeek  
sudo usermod -aG sudo zeek  
sudo passwd zeek  
  
su - zeek  
  
cd  
wget https://download.zeek.org/zeek-3.1.3.tar.gz  
tar -xzvf zeek-3.1.3.tar.gz  
cd zeek-3.1.3  
./configure --prefix=/opt/zeek --with-geoip=/usr/share/GeoIP  
make  
  
sudo mkdir /opt/zeek  
sudo chown -R zeek:zeek /opt/zeek  
sudo chmod 740 /opt/zeek  
  
make install
```

a continuación, se añade la ruta de zeek al PATH del sistema, este punto es esencial para que se puedan automatizar la solución. Si se elige otra path de instalación habrá que hacer las modificaciones respectivas.

```
sudo nano /etc/profile.d/zeekctl.sh
export PATH="${PATH}:/opt/zeek/bin"
```

Se procede a la configuración de Zeek en función de los objetivos planteados y conforme referido anteriormente se necesitan los logs en formato Json por lo cual se hace este cambio opcional en la configuración del sensor, añadiendo al final del documento (local.zeek) la instrucción relativa al formato de output:

```
sudo nano /opt/zeek/share/zeek/site/local.zeek
# Output to JSON
@load policy/tuning/json-logs.zeek
```

Se ha considerado también relevante la capacidad de analizar de forma dinámica y al nivel del sensor el tráfico en forma de ficheros. Por ello, se activa su extracción, posteriormente se afinará esta instrucción de forma a que el impacto en el sistema no sea tan relevante, eligiendo solo determinado tipo de ficheros.

```
sudo nano /opt/zeek/share/zeek/site/local.zeek
# Extract Files
@load frameworks/files/extract-all-files
```

A continuación, se procede a los cambios de configuración para definir la interface de red donde se reciben los datos a inspeccionar.

```
sudo nano /opt/zeek/etc/node.cfg
[zeek]
type=standalone
host=localhost
interface=enp0s8
```

**Ilustración 23 - /opt/zeek/etc/node.cfg**

```
sudo nano /opt/zeek/etc/networks.cfg
10.1.1.0/24      Lab Net
192.168.33.0/24 Home Net
10.0.0.0/8      Private IP space
172.16.0.0/12   Private IP space
192.168.0.0/16  Private IP space
```

**Ilustración 24 - /opt/zeek/etc/networks.cfg**

Se requiere ejecutar los comandos siguientes para que el usuario `zeek` pueda capturar el tráfico (Después de algunos cambios de configuración en que el comando **`zeekctl diag`** reportaba fallos se ha mostrado útil volver a ejecutar estos procedimientos).

```
sudo setcap cap_net_raw,cap_net_admin=eip /opt/zeek/bin/zeek
sudo setcap cap_net_raw,cap_net_admin=eip /opt/zeek/bin/zeekctl
sudo setcap cap_net_raw,cap_net_admin=eip /opt/zeek/bin/capstats
```

La configuración base está concluida y por lo tanto se procede a la instalación y arranque del sensor. Este proceso puede ser simplificado mediante el comando `deploy` que detiene el servicio, si este está en ejecución, ejecuta el comando `install` y finalmente el comando `start`. Si se producen errores se podrán consultar mediante la opción `diag` del comando `zeekctl`. Se revisará la configuración y después se repetirá el `deploy`.

```
zeekctl deploy
zeekctl diag
```

Y, por último, se añade la opción de control al cron de forma a que se haga la supervisión del sensor. Ejecutar desde el usuario `zeek`.

```
crontab -e
    */5 * * * * /opt/zeek/bin/zeekctl cron
```

Para la creación de mapas con información geográfica basada en la información de las direcciones IP es importante actualizar el fichero de geolocalización. Es necesario previamente crear una cuenta gratuita y descargar el fichero `GeoLite2-City_{fecha}` y copiarlo a la localización local.

<https://www.maxmind.com/en/geolite2/signup>

Se descargan los ficheros `GeoLite2-City_yyyyxxzz.tar.gz` y `GeoLite2-Country_yyyyxxzz.tar.gz`, donde `yyyyxxzz` corresponderá a la última fecha disponible.

```
tar -zxvf GeoLite2-City_20200421.tar.gz
sudo mv GeoLite2-City_20200421/GeoLite2-City.mmdb /usr/share/GeoIP/
tar -zxvf GeoLite2-Country_20200421.tar.gz
sudo mv GeoLite2-Country_20200421/GeoLite2-Country.mmdb \
/usr/share/GeoIP/
```

Para testar la configuración de la funcionalidad geoup se ejecuta el comando:

```
zeek -e "print lookup_location(216.58.211.32);"
```

```
$ zeek -e "print lookup_location(216.58.211.32);"  
[country_code=US, region=CA, city=Mountain View, latitude=37.4043, longitude=-122.0748]
```

Ilustración 25 - validación funcionalidad GeolIP

Para concluir el proceso de instalación de dependencias en el servidor Zeek, se procede a la instalación de un parser Json, **JQ**<sup>20 21</sup>. Esta utilidad será útil a la hora de localizar información dentro de los logs que están en formato Json en la medida que el utilitario Zeek-Cut solo soporta los ficheros en formato nativo.

```
sudo apt-get install jq
```

Se puede probar el correcto funcionamiento de JQ ejecutando una consulta contra uno de los logs de Zeek, por ejemplo, el log de conexiones:

```
cat /opt/zeek/logs/current/conn.log | jq 'select(.service == "ntp")'
```

```
{  
  "ts": 1590437622.623487,  
  "uid": "ChyyUH12Xxv1fE6Nw4",  
  "id.orig_h": "194.117.47.42",  
  "id.orig_p": 123,  
  "id.resp_h": "10.1.1.10",  
  "id.resp_p": 123,  
  "proto": "udp",  
  "service": "ntp",  
  "conn_state": "S0",  
  "local_orig": false,  
  "local_resp": true,  
  "missed_bytes": 0,  
  "history": "D",  
  "orig_pkts": 1,  
  "orig_ip_bytes": 76,  
  "resp_pkts": 0,  
  "resp_ip_bytes": 0  
}
```

Ilustración 26 – prueba de instalación con query JQ

En el servidor Zeek quedan pendientes de instalar y configurar los Beats, no solo el Filebeat para enviar los logs a Elasticsearch, pero también Auditbeat, Metricbeat y Packetbeat. El objetivo es que además del análisis de tráfico de red se pueda tener control y estadísticas sobre el uso del sensor. Se procederá también a la configuración de IntelStack e YARA pero primeramente se prepara el servidor ELK para disponer de la información de conexión para configurar los demás módulos.

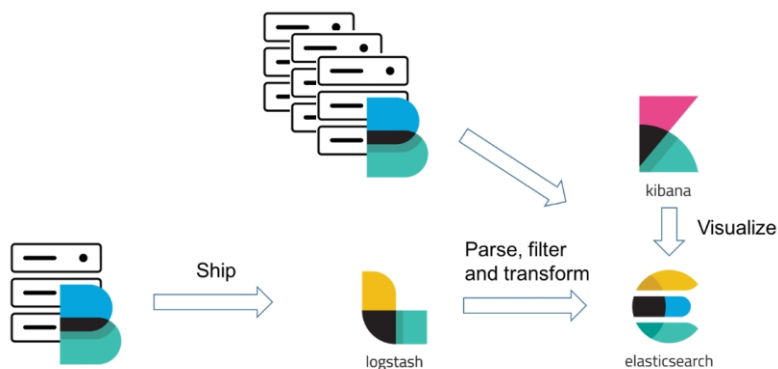
---

<sup>20</sup> <https://stedolan.github.io/jq/>

<sup>21</sup> <https://www.sans.org/blog/parsing-zeek-json-logs-with-jq/>

## 2.4 - Instalación y Configuración ELK

Para la instalación de **Elastic Stack** (ELK) se ha seguido la documentación<sup>22</sup> de forma a lograr una instalación actualizada y segura. La versión del stack es la 7.6.2-amd64, disponible a fecha del proyecto.



**Ilustración 27 - ELK diagrama de flujo**

La instalación de ELK tiene como dependencia JavaVM, la cual ya se ha instalado en la configuración base. Se procede a la descarga de los paquetes de Elasticsearch, Kibana, Logstash, Beats y su instalación y después a su configuración. El proceso detallado se añade como anexos a esta memoria.

La configuración seleccionada para Elasticsearch condiciona la configuración de todos los demás módulos, por lo tanto, se empieza por su instalación y configuración. La arquitectura elegida y los requisitos del proyecto requieren la activación de las opciones de seguridad de ELK. Para activar las opciones de seguridad y poder hacer la instalación distribuida es necesario activar la encriptación en la comunicación entre los módulos.

Si se optara por configurar todos los módulos en una misma VM se podrían activar las opciones de seguridad sin necesidad de la configuración TLS. Podría opcionalmente evitarse la configuración de la encriptación si los módulos distribuidos enviaran su información a Logstash, pero los Beats no se configurarían de esa forma.

---

<sup>22</sup> <https://www.elastic.co/guide/en/elastic-stack-get-started/current/get-started-elastic-stack.html>



Los Beats se instalarán de forma a que hagan la ingesta de los logs directamente en Elasticsearch y la configuración de Logstash será un proceso posterior para algunos logs generados por herramientas para las cuales no existe un template en Filebeats.

Los datos del servidor ELK relevantes al nivel de la configuración son los siguientes:

<b>Hostname</b>	<b>elk.ampaulo.lan</b>
<b>Ip Address</b>	<b>10.1.1.11</b>
<b>Cluster Name</b>	<b>elk-zeek-ids</b>

En el servidor ELK se procede a la instalación de Elasticsearch, pero inclusive para este proyecto se podría haber optado por una instalación temporal en ElasticCloud. Es importante considerar que la configuración distribuida y segura implica la configuración de certificado de seguridad – TLS. Se ha optado por la creación de un certificado privado. Todo el proceso está documentado en el anexo de instalación de ELK.

La falta de implementación de TLS impide que se pueda activar la seguridad cuando la instalación es distribuida y ese es uno de los motivos por los cuales se pueden encontrar diversas instancias de Elasticsearch y de Kibana accesibles desde internet, sin necesidad de usuario y clave para acceder a sus datos, como se podrá comprobar, por ejemplo, haciendo algunas consultas en shodan.io

Después de la configuración de los módulos principales de ELK se documenta también el proceso de configuración de los Beats. La configuración de Logstash se verá más en detalle a continuación al configurar los logs de pfSense.

## **2.5 - Pruebas de Monitorización con Zeek**

Se ha procedido a la instalación y configuración de Zeek y podemos verificar mediante ZeekControl que el servicio se encuentra en ejecución.

```
Welcome to ZeekControl 2.1.0-11
Type "help" for help.
[ZeekControl] > status
Name      Type      Host      Status   Pid      Started
zeek      standalone localhost running  8973    10 Apr 19:30:30
```

Ilustración 28 - ZeekControl, verificación de estado de proceso

Zeek genera logs en función de la actividad de red y de los protocolos que se activan para captura y análisis. La ruta de instalación de Zeek y de los logs son:

`/opt/zeek`

`/opt/zeek/logs/`

`/opt/zeek/logs/current`

En la carpeta *logs* se guardan los ficheros en carpetas por cada día y en *current* los ficheros en procesamiento. Esta estructura es relevante para configuración del módulo Zeek del Filebeat que se configurará para hacer la ingesta en Elasticsearch.

En la configuración de Zeek se ha optado por guardar los logs en formato Json, por lo tanto, cada fichero de capturas generadas contiene una o más líneas en que cada línea corresponde a un documento Json. Inspeccionando la carpeta `/opt/zeek/logs/current` se verifica la existencia de algunos ficheros de log de las operaciones actuales, la configuración actual es por defecto.

Para comprobar el funcionamiento se ejecutará un comando de **ping** a un nombre de dominio, para lo cual se elige [www.google.es](http://www.google.es). Se puede verificar que por esta acción se captura información en dos ficheros, en el primer lugar, en el fichero **dns.log** y, después, en el fichero **conn.log**

```
dns.log
{"ts":1586543643.780092,"uid":"ClIKbn1BfX89VcqNR4","id.orig_h":"10.1.1.10","id.orig_p":47998,"id.resp_h":
{"ts":1586543643.807337,"uid":"CKHiG73N9Do8cuEApe","id.orig_h":"192.168.33.200","id.orig_p":31649,"id.res
{"ts":1586543643.811785,"uid":"CtACVU2Oc29BBKqMwd","id.orig_h":"10.1.1.10","id.orig_p":43109,"id.resp_h":

conn.log
{"ts":1586543574.337752,"uid":"ClaSGo3CcUef5IAyg2","id.orig_h":"193.136.164.4","id.orig_p":123,"id.resp_h
{"ts":1586543643.780092,"uid":"ClIKbn1BfX89VcqNR4","id.orig_h":"10.1.1.10","id.orig_p":47998,"id.resp_h":
{"ts":1586543643.807337,"uid":"CKHiG73N9Do8cuEApe","id.orig_h":"192.168.33.200","id.orig_p":31649,"id.res
```

Ilustración 29 - Logs Zeek

Cada línea es un documento json y su estructura depende del protocolo. El fichero **local.zeek** es el fichero principal de configuración de Zeek y ahí se

especificado que los logs se guarden en formato Json como se ha indicado anteriormente.

```
dns.log
{
  "ts": 1586543643.780092,
  "uid": "CLIKbn1BfX89VcqNR4",
  "id.orig_h": "10.1.1.10",
  "id.orig_p": 47998,
  "id.resp_h": "192.168.33.254",
  "id.resp_p": 53,
  "proto": "udp",
  "trans_id": 47711,
  "rtt": 0.009515047073364258,
  "query": "www.google.es",
  "qclass": 1,
  "qclass_name": "C_INTERNET",
  "qtype": 1,
  "qtype_name": "A",
  "rcode": 0,
  "rcode_name": "NOERROR",
  "AA": false,
  "TC": false,
  "RD": true,
  "RA": true,
  "z": 0,
  "answers": ["172.217.17.3"],
  "TTLs": [66.0],
  "rejected": false
}

conn.log
{
  "ts": 1586543643.780092,
  "uid": "CLIKbn1BfX89VcqNR4",
  "id.orig_h": "10.1.1.10",
  "id.orig_p": 47998,
  "id.resp_h": "192.168.33.254",
  "id.resp_p": 53,
  "proto": "udp",
  "service": "dns",
  "duration": 0.012840032577514649,
  "orig_bytes": 62,
  "resp_bytes": 106,
  "conn_state": "SF",
  "local_orig": true,
  "local_resp": false,
  "missed_bytes": 0,
  "history": "Dd",
  "orig_pkts": 2,
  "orig_ip_bytes": 118,
  "resp_pkts": 2,
  "resp_ip_bytes": 162
}
```

Ilustración 30 - Logs Zeek ejemplo

## 2.6 – Configuración de Filebeat con modulo Zeek

La ingesta de los logs de Zeek en Elasticsearch se configura para que sea efectuada directamente por Filebeat sin pasar por el Logstash. Se activa el módulo correspondiente y se editan dos ficheros de configuración.

El proceso de instalación del Beat se encuentra documentado en el anexo y por ello se listan únicamente los pasos a efectuar.

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

```
sudo apt-get install apt-transport-https
```

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

```
sudo apt-get update && sudo apt-get install filebeat
```

```
sudo filebeat modules enable zeek
```

```
sudo nano /etc/filebeat/filebeat.yml
```

A continuación, se procede a la configuración específica correspondiente a Zeek. El módulo activado anteriormente se configura habilitando los tipos de protocolos que se pretenden analizar e indicando su fichero de log. Adelante analizaremos algunos de los protocolos más en detalle.

```
sudo nano /etc/filebeat/modules.d/zeek.yml
```

```
- module: zeek
  capture_loss:
    enabled: true
    var.paths: [ "/opt/zeek/logs/current/capture_loss.log" ]
  connection:
    enabled: true
    var.paths: [ "/opt/zeek/logs/current/conn.log" ]
  dce_rpc:
    enabled: true
    var.paths: [ "/opt/zeek/logs/current/dce_rpc.log" ]
```

**Ilustración 31 - /etc/filebeat/modules.d/zeek.yml (parcial)**

Las opciones de configuración de Filebeat en el servidor Zeek quedan así:

```
##### Filebeat inputs #####
filebeat.inputs:

- type: log
  enabled: false
  paths:
    - /var/log/*.log

##### Filebeat modules #####
filebeat.config.modules:
  path: ${path.config}/modules.d/*.yaml
  reload.enabled: false

##### Elasticsearch template setting #####
setup.template.settings:
  index.number_of_shards: 1

##### Kibana #####
setup.kibana:
  host: "10.1.1.11:5601"

##### Elasticsearch output #####
output.elasticsearch:
  protocol: "https"
  hosts: ["https://10.1.1.11:9200"]
  pipeline: geoiip-info
  ssl.certificate_authorities: ["/home/ac/certs/elasticsearch-ca.pem"]
  username: "elastic"
  password: "elastic"

##### Processors #####
processors:
  - add_host_metadata: ~

xpack.security.enabled: true
```

**Ilustración 32 - /etc/filebeat/filebeat.yml**

Es importante considerar las opciones de seguridad pues van a ser determinantes para la ingesta de logs en elasticsearch. Son relevantes no solo el usuario y clave, pero también la definición del host con https, que queda indicado también en el campo 'protocol'. La indicación de la autoridad de certificación (CA) solo es necesaria por consecuencia del certificado ser auto-generado y, por lo tanto, sin reconocimiento por las CAs públicas. También hay que habilitar el xpack.security, lo cual se hace al final del fichero.

Se confirma que la configuración es correcta y se reinicia el servicio

```
sudo filebeat test config
```

```
sudo filebeat test output
```

```
sudo service filebeat start
sudo service filebeat status
```

Por último, se activa el servicio para arranque automático

```
sudo systemctl enable filebeat
```

Si se producen errores se pueden identificar mediante el *journal* de sistema

```
sudo journalctl -u filebeat.service | tail -f -n 50
```

## 2.7 - Pruebas de Visualización en ELK

Efectuando algunos accesos web se comprueba que existen datos y se pueden visualizar en el dashboard '**Filebeat Zeek Overview**'

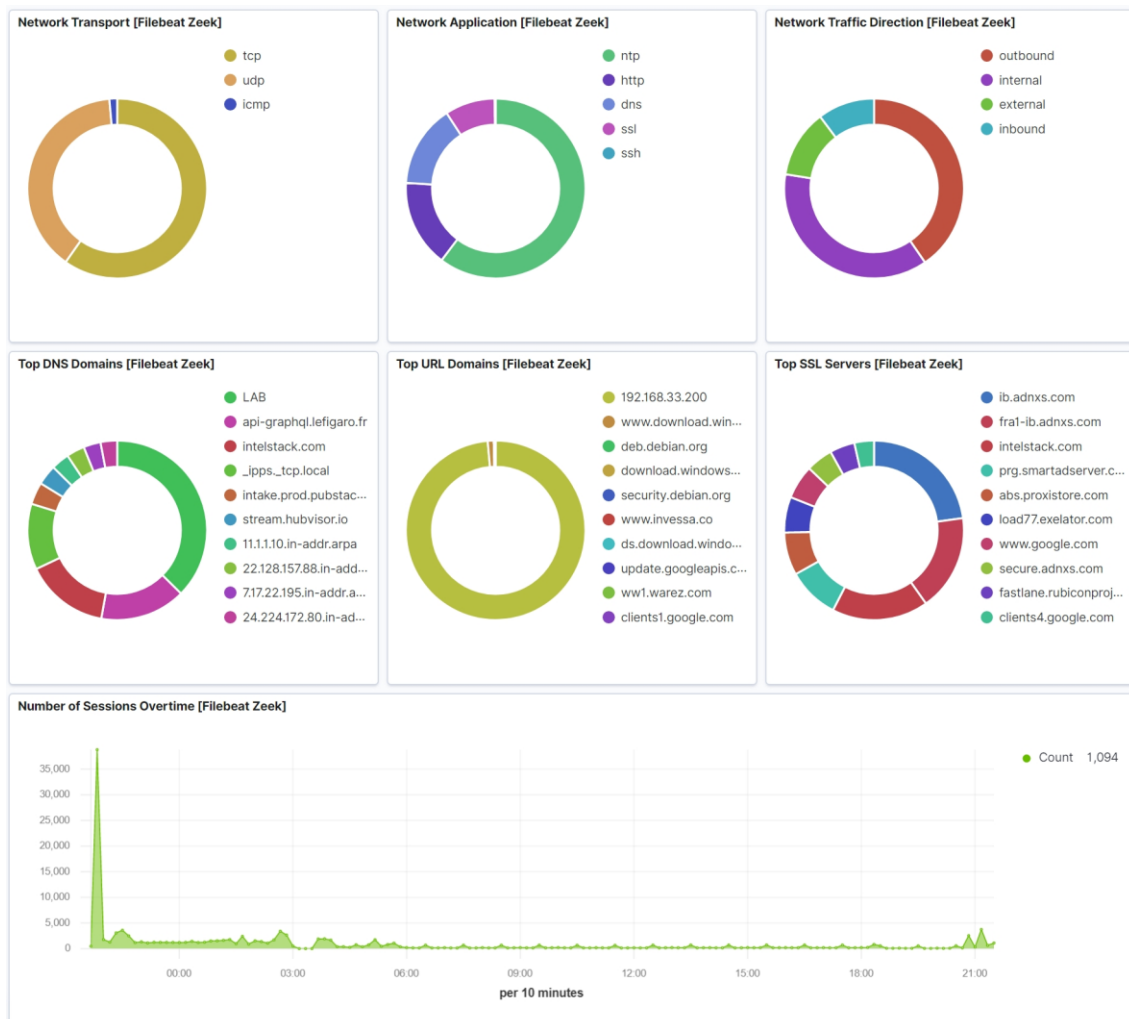


Ilustración 33 - Kibana, Zeek dashboard

## 2.8 – Análisis de Listas de Reputación

*“Uno de los mecanismos más efectivos y económicos para reducir el spam es bloquear el tráfico de correos electrónicos con origen en ciertas direcciones IP sospechosas, que se incluyen en Listas de Reputación, actualizadas en tiempo real.”<sup>23</sup>*

Las Listas de Reputación como aquella que la RedIRIS gestiona permiten filtrar el tráfico en función de determinados criterios y el spam es posiblemente el caso más reconocido. Por otro lado, existen determinadas actividades con menor visibilidad, pero posiblemente con mayor impacto en la seguridad.

Están disponibles listas de dominios usados para Phishing, para distribución de Malware, bien como identificadores de equipos que son parte de Botnets y direcciones de servidor C&C (Command and Control). Por otro lado, hay listas también de hash de ficheros que representan malware, aplicación de persistencia después de la invasión de un dispositivo, etc.

La creación y mantenimiento de estas listas son el resultado de la acción de organizaciones y entidades que las hacen disponibles para uso general. En todo caso su mantenimiento de forma manual fácilmente se puede hacer imposible.

Por ello existen agregadores que permiten seleccionar y acceder a las listas de forma unificada y disponen además de ficheros normalizados para que sean compatibles con las distintas soluciones de seguridad.

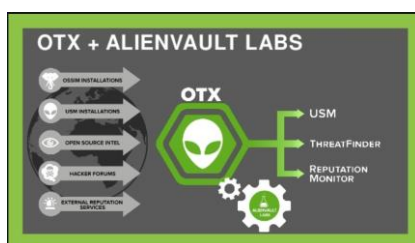


Ilustración 34 - OTX AlienVault

Se ha empezado por evaluar la solución **OTX AlienVault**<sup>24</sup> la cual es bastante completa y en actualización permanente y dispone de una API para integración

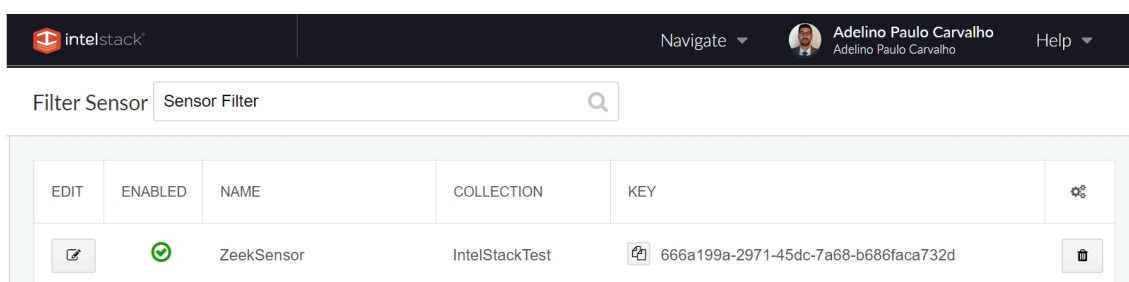
<sup>23</sup> [https://www.rediris.es/servicios/calidad\\_correo/irisrbl/index.html.es](https://www.rediris.es/servicios/calidad_correo/irisrbl/index.html.es)

<sup>24</sup> <https://otx.alienvault.com/>

con Zeek. Es una solución a la cual se puede acceder de forma gratuita, pero es parte de un ecosistema comercial más completo. Por ello, se decidió avanzar para una solución más limitada pero totalmente open-source.

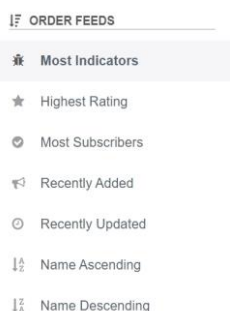
**IntelStack** es un servicio de suscripción gratuita que permite elegir listas de reputación y crear grupos a los cuales en su terminología se llaman de 'sensor'. Cada sensor está asociado a una KEY (api key) que en el proceso de configuración se pasa como parámetro y permite que las colecciones suscritas se actualicen automáticamente.

<https://intelstack.com/>



**Ilustración 35 - IntelStack, suscripción**

El proceso de configuración del sensor y generación de la Key es sencillo y lineal, el primer paso es crear una Collection y asignarle un nombre. Después, se eligen los Feeds seleccionando de entre las listas disponibles.



**Ilustración 36 - criterios de ordenación de listas**

La selección se puede hacer mediante criterios como la cantidad de indicadores que contiene la lista, el número de suscriptores o su actualización. En cada lista es posible acceder a un resumen, comentarios de usuarios y link de acceso. Es posible también asignarle un valor de una a cinco estrellas que

puede usarse para promover las listas más útiles. Como se verá más adelante algunas listas no están actualizadas o no se logra la conexión.

Existe también un apartado de métricas que permite comparar los indicadores globales con aquellos de las listas seleccionadas en el sensor creado. Es posible verificar que la mayoría de las listas contiene información sobre dominios.

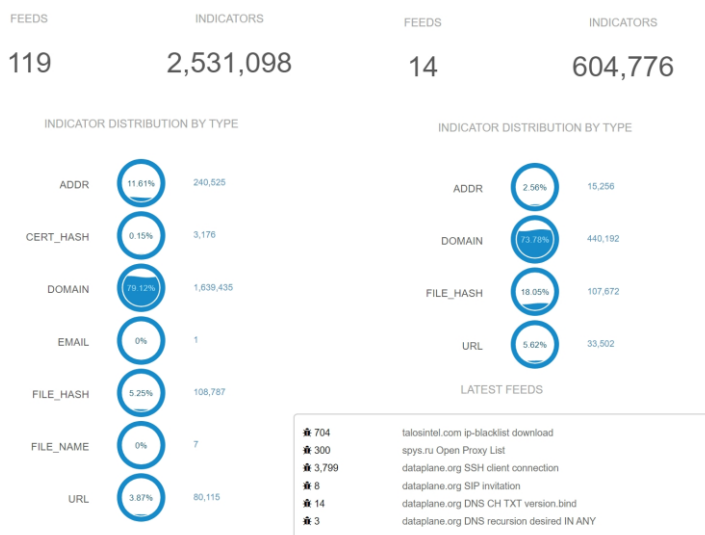


Ilustración 37 - IntelStack, comparativo de metricas

Mediante la configuración de la colección y actualización del sensor creado en IntelStack es posible, configurando la actualización automática de Zeek, hacer con que los cambios efectuados en la web se verifiquen en Zeek en corto espacio de tiempo. El proceso de automatización se configura en el siguiente punto, pero es importante verificar después de la suscripción que las listas siguen disponibles. La lista que no tiene fecha de actualización no se ha descargado.

```

root@zeek:~# sudo -u intel-stack-client intel-stack-client list
intel-stack-client 04:58:28 [ ] Pulling feed list from Intel Stack.
-----+-----+-----+-----+
ID | NAME | LAST UPDATED | INDICATOR COUNT |
-----+-----+-----+-----+
7 | Known-Tor-Exit-Nodes | 05/05/20-01:33-am (UTC) | 7916 |
9 | Cyber-Crime-Tracker | 05/05/20-01:32-am (UTC) | 19049 |
18 | PhishTank-Intel-Feed-(Verified) | 05/05/20-01:33-am (UTC) | 14352 |
59 | ET:-Botnet-Command-and-Control | 05/05/20-01:33-am (UTC) | 1338 |
72 | danger.rulez.sk-SSH-Brute-Force-Report | 05/05/20-01:33-am (UTC) | 845 |
74 | hosts-file.net-Malware-Domains | 05/03/20-04:34-am (UTC) | 268251 |
82 | hosts-file.net-Phishing-Domains | 05/04/20-04:18-pm (UTC) | 157207 |
86 | sysctl.org-Domain-Blocklist-(Ads) | 04/29/20-10:25-pm (UTC) | 14693 |
87 | vxvault.net-Malware-URLs | 05/05/20-01:34-am (UTC) | 101 |
96 | malwareconfig.com-AFTnotes-(Hashes) | - | 0 |
104 | torproject.org-Official-Exit-Node-List | 05/05/20-01:34-am (UTC) | 1355 |
216 | volexity.com-Virtual-Private-Keylogging-report-IGCs | 05/05/20-01:33-am (UTC) | 44 |
220 | abuse.ch-Feedo-Tracker-Malware-Hashes-(hashes-only)- | 05/05/20-01:32-am (UTC) | 107672 |
230 | dataplane.org-SSH-client-connection | 05/03/20-12:57-am (UTC) | 3799 |
-----+-----+-----+-----+
TOTAL | 596622 |
-----+-----+-----+-----+
Feed-name> = Not Fetched Yet
Feed-name> = On Disk
* = Update Available
LAST UPDATED: Column = Server time last updated - not local time.

```

Ilustración 38 - IntelStack, mantenimiento de listas



## 2.9 – Configuración de listas de reputación - IntelStack

El proceso de instalación y configuración de IntelStack tiene algunas particularidades y, por ello, se documentan todos los pasos requeridos para lograr una configuración funcional.

A fecha de proyecto la instalación se hace mediante la ejecución de un script<sup>25</sup> para obtener la información del repositorio. Se recomienda revisar el script después de la descarga, pero si lo damos por válido podemos ejecutarlo directamente y de seguida se procede a la instalación de forma convencional.

```
curl
https://packagecloud.io/install/repositories/intelstack/client/script.
deb.sh | sudo bash
sudo apt-get install intel-stack-client
```

Las opciones de configuración disponibles y su impacto sobre la funcionalidad del servicio se pueden consultar en la página del producto<sup>26</sup> los comandos listados a continuación son el conjunto mínimo para lograr la integración con Zeek que se ha implementado en este proyecto (el carácter \ denota quiebra de línea en comandos más largos)

```
sudo chown intel-stack-client.intel-stack-client \
/opt/zeek/share/zeek/site/local.zeek
sudo -u intel-stack-client -g intel-stack-client intel-stack-client
nsm zeek
sudo -u intel-stack-client -g intel-stack-client intel-stack-client \
api 666a199a-2971-45dc-7a68-b686faca732d
sudo -u intel-stack-client -g intel-stack-client intel-stack-client \
config --set zeek.restart=true
sudo -u intel-stack-client -g intel-stack-client intel-stack-client \
config --set zeek.include=false
```

**intel-stack-client** se ejecuta mediante un usuario con permisos limitados por lo cual ha de darse permisiones sobre la configuración de zeek y a continuación se definen los parámetros de ejecución y de acceso al servicio web.

---

<sup>25</sup> <https://intelstack.com/client/>

<sup>26</sup> <https://intelstack.com/client/usage>

```
sudo chmod 775 /opt/intel-stack-client/logs/intel-stack-client-sql.log
sudo chown -R intel-stack-client:intel-stack-client /opt/intel-stack-client/
```

El parámetro **nsm** permite elegir si configuramos la aplicación para integración con Bro o con Zeek y, el parámetro **api** requiere que indiquemos la Api Key que identifica el sensor generado en la web de IntelStack.

Los demás parámetros de configuración definen la relación con Zeek, en especial es relevante el parámetro **restart**, mediante este parámetro se indica a Intel-stack-client que después de la actualización de las listas se procede al reinicio de Zeek.

Para activación de la funcionalidad en Zeek añadimos al final del fichero de configuración esta entrada:

```
sudo nano /opt/zeek/share/zeek/site/local.zeek
# intelstack
@load /opt/intel-stack-client/frameworks/intel
```

Y por último activamos el servicio y comprobamos la configuración

```
sudo systemctl enable intel-stack-client
sudo -u intel-stack-client intel-stack-client list
```

La verificación de las listas suscritas y su estado de actualización se hace mediante el comando **list** y el resultado es el presentado en la imagen anterior. Previa comprobación del correcto funcionamiento faltaría crear el script de actualización periódica lo cual se hará desde la sesión del usuario zeek.

```
sudo nano /opt/zeek/bin/intel-update
echo "****Updating Threat Feeds****"
sudo -u intel-stack-client intel-stack-client pull
echo "****Applying Updates****"
/opt/zeek/bin/zeekctl check
/opt/zeek/bin/zeekctl install
echo "****Restarting Zeek****"
/opt/zeek/bin/zeekctl restart
```

```
sudo chown -R zeek:zeek intel-update
sudo chmod +x /opt/zeek/bin/intel-update
```

Por último, añadimos la configuración para ejecución automática

```
sudo nano /etc/crontab
30 * * * * zeek /opt/zeek/bin/intel-update
```

Intel-stack-client se ha instalado en **/opt**, las entradas de las listas suscritas son descargadas y se genera un fichero de nombre **master-public.dat** que es la base de datos del servicio. La inspección del fichero permite verificar que, por ejemplo, para URLs consiste en la misma URL, el tipo de detección y su lista de origen.

```
cat /opt/intel-stack-client/frameworks/intel/master-public.dat | tail
```

```
www.barentor.tk Intel::DOMAIN from http://hosts-file.net/emd.txt via https://intelstack.com, F
www.360torontomls.com Intel::DOMAIN from http://hosts-file.net/emd.txt via https://intelstack.com, F
www.boatrepairdiy.com Intel::DOMAIN from http://hosts-file.net/emd.txt via https://intelstack.com, F
www.talentedtwo.us Intel::DOMAIN from http://hosts-file.net/emd.txt via https://intelstack.com, F
www.tnescluborg200.tk Intel::DOMAIN from http://hosts-file.net/psh.txt via https://intelstack.com, F
euro-union-uk.com Intel::DOMAIN from http://hosts-file.net/emd.txt via https://intelstack.com, F
```

Ilustración 39 - IntelStack master-public.dat

Se prueba que la integración con Zeek es correcta y funcional eligiendo una URL del fichero y navegando hacia ella. Por inspección del resultado se puede verificar que en el fichero de logs de Zeek, **intel.log** se añade una nueva línea, entre otras es significativa la propiedad 'matched' donde consta el tipo de indicador que ha activado la detección, en este caso "Intel:DOMAIN".

```
root@zeek:/opt/zeek/logs/current# cat intel.log
{"ts":1586491133.251249,"uid":"CoXMki4Uyq3SLS1KU5","id.orig_h":"10.1.1.10","id.orig_p":39297,"id.resp_h":"192.168.33.254","id.resp_p":53,"seen.indicator":"www.talentedtwo.us","seen.indicator_type":"Intel::DOMAIN","seen.where":"DNS::IN_REQUEST","seen.node":"zeek","matched":["Intel:DOMAIN"],"sources":["from http://hosts-file.net/emd.txt via https://intelstack.com,"]}
```

Ilustración 40 - IntelStack prueba de detección

Se puede así concluir que la fase 2 del proyecto – Diseño y Construcción, se ha terminado logrando un entorno funcional, lo cual se ha validado realizando unas pruebas básicas de verificación de la infraestructura. Los resultados se han presentado donde corresponden.

En resumen, el proyecto al final de esta fase consiste en un servidor donde se ha desplegado Zeek y que se considera el sensor de recolección de actividad de red; un segundo servidor donde se ha desplegado ELK como repositorio de los datos indexados y como entorno de visualización de los eventos.

Ambos servidores y demás hosts de la red se encuentran conectados por un router pfSense lo cual se ha configurado también como firewall. Los logs de conexión de este firewall se añadirán también a la información que se ingesta en Elasticsearch.

En la fase 3 – Implementación SIEM, se definirá el entorno de gestión de la solución desarrollada y se harán las pruebas de funcionalidad. El planteamiento consiste en el abordaje de 3 puntos:

- 1- La puesta en marcha del sistema de SIEM (Security Information and Event Management) añadido por Elastic a la solución ELK con la con la consecuente integración de los índices de Zeek en ese conjunto de datos para ejercitar las reglas de Detección que se activarán. Se complementará la información del tráfico de red interna con los eventos del log de pfSense mediante la ingesta de estos en el repositorio.
- 2- Añadir el soporte para el análisis e identificación de ficheros mediante la integración de las reglas YARA con Zeek y envío de los casos positivos a Elasticsearch.
- 3- Implementar el envío de notificaciones mediante la configuración de Elastalert en función de la detección de determinadas actividades de red o de la superación de determinados umbrales, por ejemplo, de transferencia de datos hacia al exterior. De esta forma se pretende lograr un mecanismo de alerta en tiempo real que permita impedir que el problema se siga produciendo.

### 3. Implementación SIEM

*Cyber security is the protection of internet-connected systems, including hardware, software and data, from cyberattacks. In a computing context, security comprises cyber security and physical security -- both are used by enterprises to protect against unauthorized access to data centers and other computerized systems. The goal of cyber security is to limit risk and protect IT assets from attackers with malicious intent.<sup>27</sup>*

La seguridad de los sistemas de información de toda y cualquier organización es crítica y las organizaciones reconocen esta realidad. En efecto, la inversión en sistemas de protección del perímetro de red y de los propios equipos de trabajo es vista como una necesidad básica por las organizaciones y no como una comodidad que se puede prescindir. Las organizaciones que aún no lo incluyan en sus planes estratégicos se exponen a un riesgo que, de materializarse, tendrán dificultad en justificar a sus inversores y clientes.

El despliegue de múltiples capas de firewall de red, firewall de aplicación y soluciones de antivirus centralizadas o locales, y todo un conjunto adicional de medios de protección activa, parece ofrecer toda la seguridad necesaria. Un IDS es un sistema pasivo, lo cual al contrario de los elementos referidos no sirve para impedir el ataque al recurso que se pretende defender.

No obstante toda la inversión en medios de protección activa de los sistemas de información la realidad demuestra que los ciberataques siguen afectando a todo el tipo de organizaciones y, en muchos casos las pérdidas y costes derivados son millonarios.

El informe “2019 Data Breach Investigations Report”<sup>28</sup> publicado por Verizon, deja en evidencia que organizaciones de sectores críticos, como son las entidades financieras y organizaciones de salud, las cuales tienen regulado el cumplimiento obligatorio de normativas y buenas prácticas de seguridad de los sistemas de información, son también víctimas de ciberataques.

---

<sup>27</sup> <https://searchsecurity.techtarget.com/definition/cybersecurity>

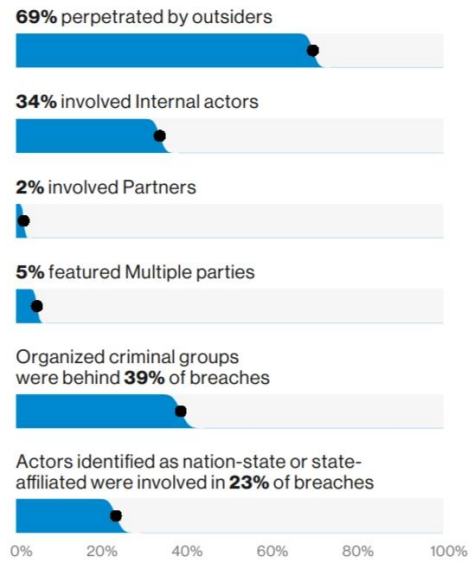
<sup>28</sup> <https://enterprise.verizon.com/resources/reports/2019-data-breach-investigations-report.pdf>

El resumen del estudio se usa como base de justificación de la implementación de un IDS, como el que se ha propuesto desplegar en este trabajo, y también, se pretende hacer evidente que el control en tiempo real de las evidencias recogidas es una necesidad para las empresas.

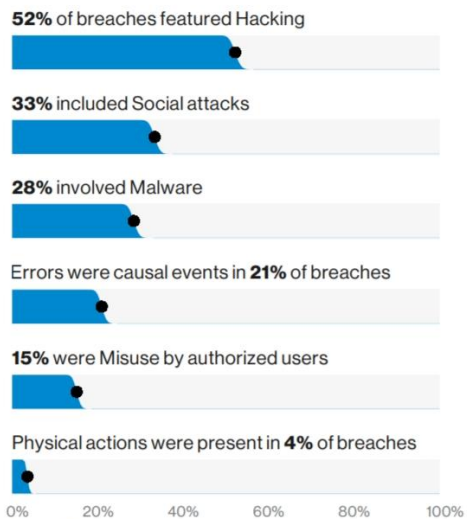
## Summary of findings



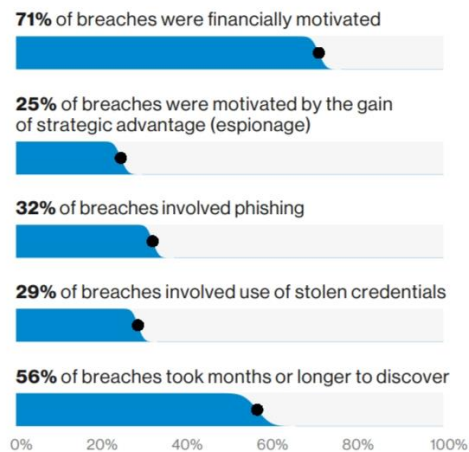
**Breaches**  
Figure 2. Who are the victims?



**Breaches**  
Figure 4. Who's behind the breaches?



**Breaches**  
Figure 3. What tactics are utilized?



**Breaches**  
Figure 5. What are other commonalities?

### Ilustración 41 - 2019 Data Breach Investigations Report

Se comparte necesariamente la idea generalizada de que un sistema de información debe ser protegido de forma proactiva, identificando y mitigando los riesgos y, para ello, desplegando los medios adecuados a lo que se

pretende defender. Pero, por otro lado, se considera como cierto que todo el sistema, independientemente de los medios y medidas de seguridad implementadas, podrá (y probablemente va a) sufrir ataques que superen esas medidas.

La tarea de proteger un sistema de información y lograr su protección total, requiere teóricamente considerar todos los casos posibles y mitigar todos esos riesgos, lo cual es imposible; por otro lado, a quien ataca le es suficiente con encontrar un solo caso que pueda ser explotado con suceso. No pocas veces, como se puede comprobar consultando la base de datos CVE<sup>29</sup>, los fallos de seguridad están mismo en las herramientas y soluciones usadas para protección.

Si a todas estas posibilidades al alcance del atacante le añadimos un entorno cambiante que no permite definir una superficie de exposición estática y constante, entonces, se ha de concluir que la monitorización activa de lo que sucede en toda la red y en sus nodos es una necesidad no menos importante que las medidas de protección referidas anteriormente.

El despliegue de sensores de red con la finalidad de monitorizar la actividad y con el objetivo de detección de anomalías tendrá que ser complementado con soluciones que permitan de forma automatizada analizar los problemas desde varias perspectivas y correlacionando distintas fuentes de datos.

La ingesta de la información que circula por la red, así como la recoja de los eventos en los nodos, como firewalls, servidores y puestos de trabajo que resulta de la implementación propuesta, permitirá identificar posibles amenazas de forma automática, analizar su evolución en el tiempo y hacer el cruzamiento de datos con fuentes de información externas que permitirán actuar de forma adecuada en respuesta al riesgo.

Los sistemas SIEM (Security Information and Event Management) son la herramienta adecuada para dotar cualquier organización de esta capacidad. De esta forma podrán disponer de información actualizada y válida sobre las amenazas que se detectan en su entorno.

---

<sup>29</sup> <https://cve.mitre.org/>

Elastic SIEM<sup>30</sup> es un módulo más en el ecosistema de ELK, disponible de forma gratuita al alcance de cualquier organización permitiendo el acceso a toda una base de conocimiento que de otra forma tendría que ser construida.

Esta herramienta será la base de la solución de monitorización propuesta, se demostrará no solo el abanico de funcionalidades ya disponibles como la posibilidad de incrementar su valor añadiendo reglas de detección propias sobre los resultados de las listas de seguridad implementadas y también un sistema de alerta y notificación en tiempo real.

Para que un sistema de Gestión de Eventos e Información de Seguridad (SIEM) sea verdaderamente efectivo se requiere que además de la información de red se pueda contar con las informaciones de los nodos, por ello se ha optado por configurar los distintos módulos de la solución Elastic y a continuación se añade la información del log del firewall, en este caso pfSense, pero que será válida para cualquier firewall que pueda enviar sus logs en formato Syslog a un dispositivo remoto.

La gestión de potenciales vulnerabilidades que permiten defender a una organización debe considerar que en muchos casos el proceso no resulta de un suceso aislado, pero de una secuencia de acciones identificables y repetibles.

Este proceso se conoce como *The Cyber Kill Chain* y consiste en la secuencia de fases de un ciberataque.

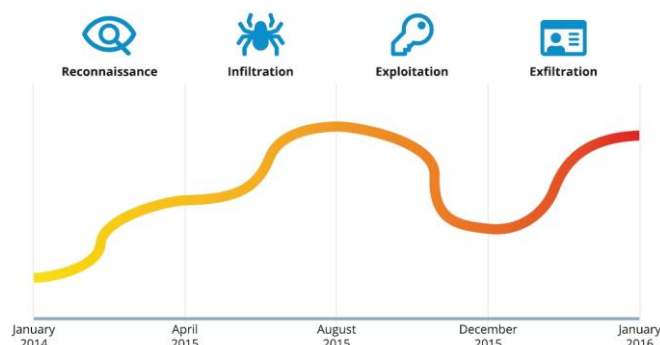
*The Cyber Kill Chain was introduced by Lockheed Martin in the late 1990s and documents the anatomy of a typical cyber breach by plotting the path of attack from start to finish. In many ways, it has become a reference model for cyber defense thinking for more than 20 years.*<sup>31</sup>

---

<sup>30</sup> <https://www.elastic.co/siem>

<sup>31</sup> Identity Attack Vectors: Implementing an Effective Identity and Access Management Solution.





**Ilustración 42 - The Cyber Kill Chain, example**

Al desplegar un SIEM es importante considerar que los eventos actuales en algunos casos pueden requerir el acceso a datos pasados, por lo tanto, uno de los aspectos inherentes es la cantidad de información que se debe almacenar.

No toda la información gestionada tiene la misma importancia y si por un lado los datos sobre las métricas pueden ser útiles para evaluar a la sobrecarga o no del sistema y, por ello, solo son relevantes a corto plazo, las informaciones de los logs de conexión e inteligencia (que se verán adelante) o los logs de intentos de acceso bloqueados por el firewall tendrá importancia a medio y largo plazo.

El termino SIEM se puede encontrar desde al menos 2005, y es atribuido a Gartner en uno de sus informes sobre Seguridad de TI, y su concepto no ha cambiado sustancialmente en el tiempo. Un SIEM representa la capacidad de monitorización en tiempo real de eventos y su correlación con distintos conjuntos de datos. Son herramientas visuales que normalmente se despliegan como cuadros de mando autónomos y que van actualizando la información en función de la ingesta de datos.

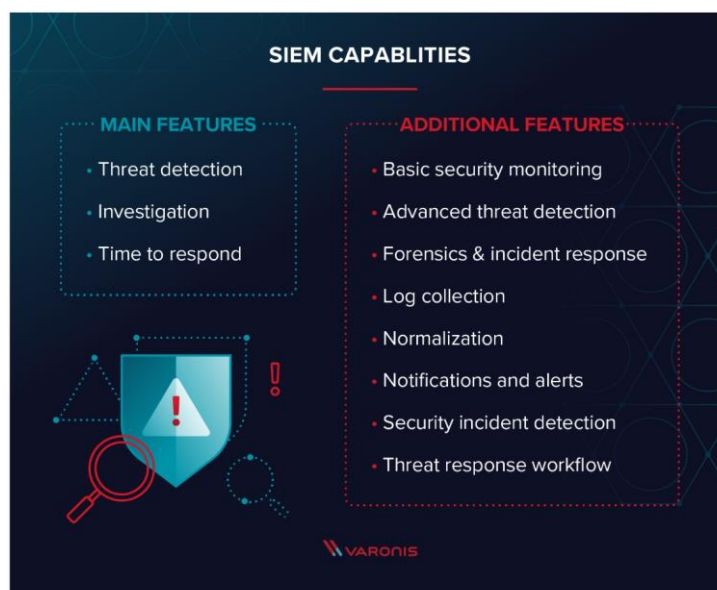
Si por un lado el almacenaje de datos y su visualización en tiempo real son los aspectos base de un SIEM, por otro, se requiere la necesidad de gestionar las acciones en respuesta a esas detecciones. La posibilidad de eliminar falsas alarmas y marcar los casos de interés es un aspecto importante y otro es la capacidad de escalar el problema, por ejemplo, enviándolo a un sistema de gestión de incidencias.



**Ilustración 43 - SIEM Process<sup>32</sup>**

En resumen, un SIEM tiene que permitir centralizar los datos de las eventuales amenazas detectadas, determinar cuáles de ellas requieren seguimiento, en su caso permitir escalar conforme el tipo o gravedad y, también, permitir disponer de la información de contexto para analizar el incidente.

En base a un documento de análisis de Gartner, Varonis hace este cuadro resumen de las 3 principales funciones de un SIEM y describe las principales características que deben incluir.



**Ilustración 44 - Security Information and Event Management Capabilities**

<sup>32</sup> <https://www.varonis.com/blog/what-is-siem/>

### 3.1 – Elastic SIEM

De la configuración efectuada en la fase 2 ha resultado un sistema operacional, con capacidad de recoja de datos, tanto desde los logs de Zeek, como también, desde los logs de métricas y auditoria de Elastic. Se cuenta por lo tanto con un conjunto de datos en la aplicación.

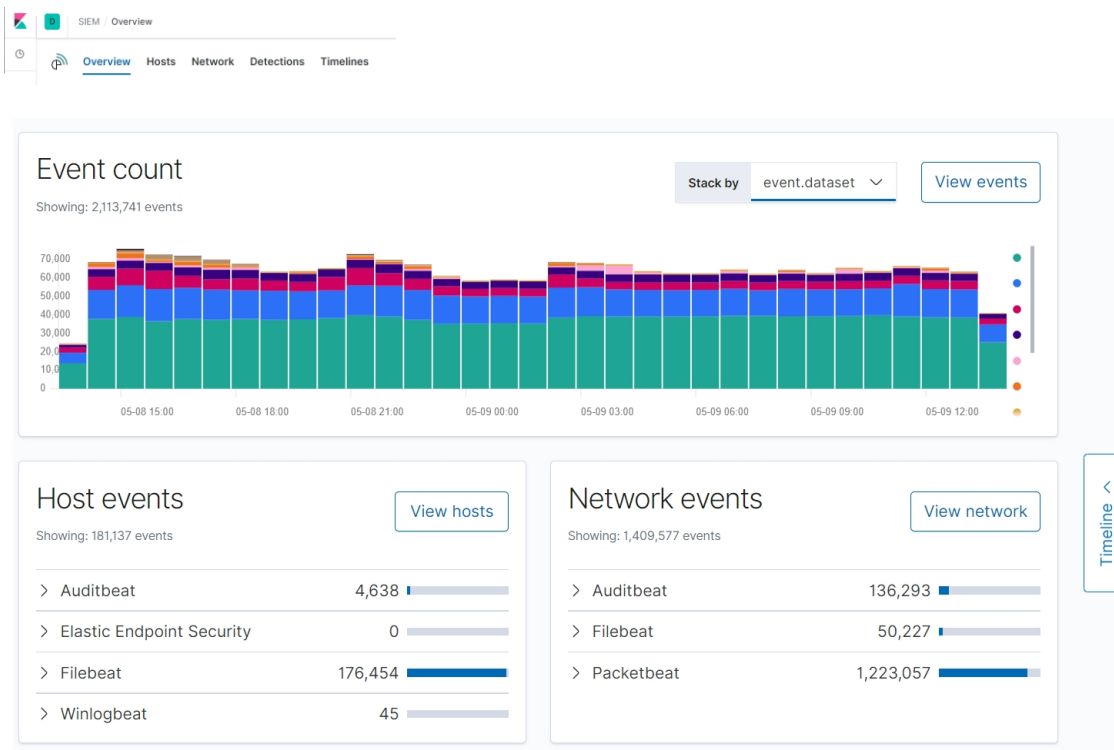


Ilustración 45 - Elastic SIEM dashboard

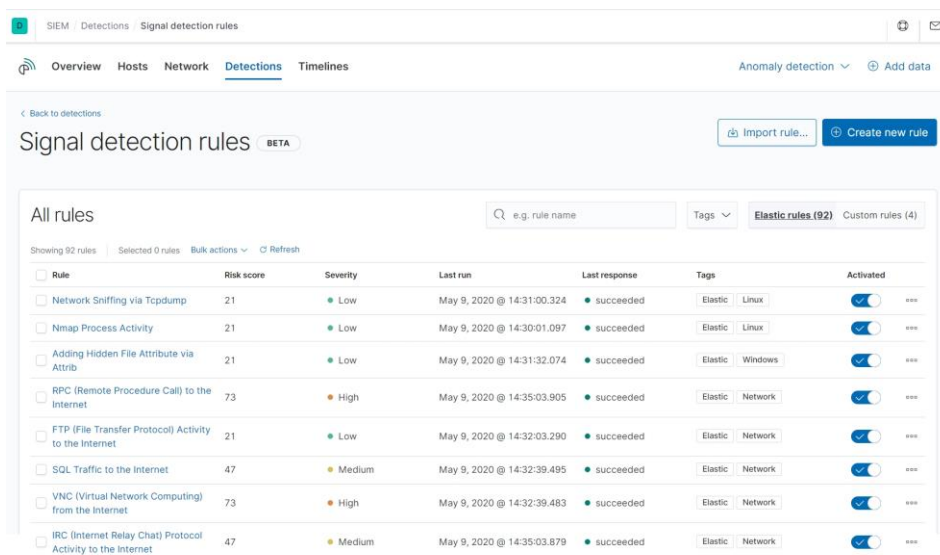
Hacer el análisis de los eventos de los nodos está fuera del alcance de este trabajo y lo que se pretende es usar las funcionalidades de detección y de gestión de los incidentes – *Detections* y *Timeline* (y a partir de la versión 7.7 la opción *Cases*).

La funcionalidad de detección de amenazas, que es aún una función beta de la solución, consiste en la aplicación de reglas al conjunto de datos indexados. Se pueden usar reglas predefinidas y también crear reglas propias. Se hace un comentario breve sobre las reglas predefinidas pero lo importante es demostrar las funcionalidades de que se disponen a la hora de crear reglas propias.

Las reglas son parte de conjunto 'Signal detection rules' y el resultado de la aplicación de las reglas es la ingesta automática en un índice de sistema (.siem-signals-default-*nnn*) pero que se puede usar en los cuadros de mando.

A fecha de desarrollo del proyecto (versión 7.6.2) existían 92 reglas disponibles que corresponden a 6 grupos: Elastic, Network, Windows, Linux, Endpoint y APM. Algunas funcionalidades y reglas solo están disponibles en la versión con suscripción. La actualización de la versión 7.7 ha elevado el número de reglas disponibles a 127, y elevando los conjuntos a 8, añadiendo ML y Packetbeat. Las reglas ML solo se pueden usar mediante suscripción.

Además, se pueden crear reglas personalizadas o importar otras reglas.



**Ilustración 46 - Elastic SIEM, Signal Detection Rules**

Todas las reglas se pueden activar, si están disponibles bajo la licencia en que se ejecuta ELK, pero en el caso de las reglas base no se pueden hacer modificaciones. Las reglas pueden usarse para detectar casos anómalos o más bien actividad de aplicaciones bien conocidas pero que se pretenden controlar.

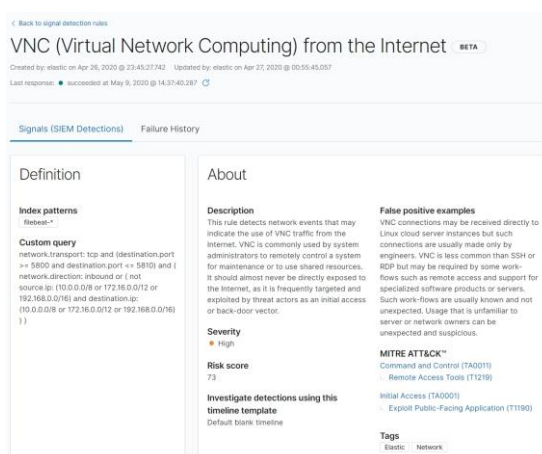
La actividad de red puede resultar en la detención de una cantidad de señales elevada por lo cual es necesario proceder a una activación cuidada evaluando porque se producen falsos positivos.

La utilidad de las reglas para el profesional de seguridad o para un SOC son inmensas pues además de permitir generar notificaciones en tiempo real cada

regla después de seleccionada permite visualizar de forma inmediata cuantas veces se ha activado y también cuales los eventos que han resultado en su activación.

Se verifican así las funcionalidades identificadas como base de un SIEM: detección, notificación y contexto para permitir el análisis.

Cada una de las reglas base dispone de información relevante para contextualizar, pero también un acceso correspondiente al MITRE ATT&C<sup>33</sup>. Siguiendo el hilo de la regla de detección de actividad VNC desde internet y considerando que en un entorno seguro no debería existir tal conexión desde internet a no ser mediante VPN o tunnel SSH, entonces la existencia de ese tráfico puede ser una señal de que el sistema está comprometido.



**Ilustración 47 - Regla de detección: VNC from Internet**

Si siguiendo el enlace de la regla se puede verificar que existe una amenaza (Dridex) que usa VNC.

Techniques Used			
Domain	ID	Name	Use
Enterprise	T1090	Connection Proxy	Dridex contains a backconnect module for tunneling network traffic through a victim's computer. Infected computers become part of a P2P botnet that can relay C2 traffic to other infected peers. <sup>[1]</sup>
Enterprise	T1185	Man in the Browser	Dridex can perform browser attacks via web injects to steal information such as credentials, certificates, and cookies. <sup>[1]</sup>
Enterprise	T1219	Remote Access Tools	Dridex contains a module for VNC. <sup>[1]</sup>
Enterprise	T1071	Standard Application Layer Protocol	Dridex has used HTTPS for C2 communications. <sup>[2]</sup>
Enterprise	T1032	Standard Cryptographic Protocol	Dridex has encrypted traffic with RSA and RC4. <sup>[2]</sup>

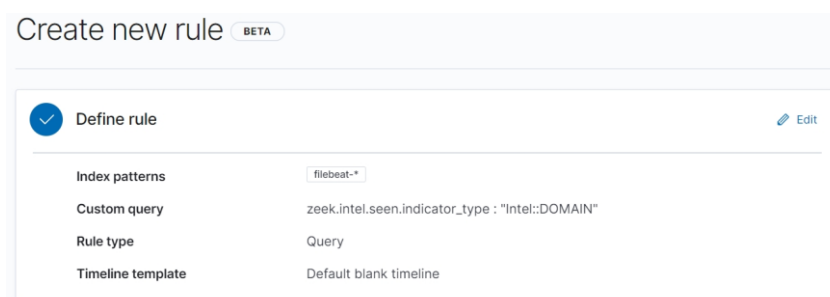
**Ilustración 48 - Regla de detección, MITRE ATT&CK information**

<sup>33</sup> <https://attack.mitre.org/>

La importancia del uso de las reglas de detección queda por lo tanto demostrada, pues permite automatizar la actividad de detección de amenazas y/o situaciones de sistemas comprometidos.

Crear reglas personalizadas permite añadir funcionalidades al sistema en función del contexto de la organización y la personalización del sistema de monitorización. El proceso de creación de reglas es sencillo y lineal por lo cual no se verá en detalle. Pero se van a demostrar los puntos clave.

Las condiciones de la regla se definen con una consulta KQL y/o filtros como los usados en la creación de visualización, etc. Previamente se define el índice o índices sobre los cuales será de aplicación.



**Ilustración 49 - Creación de una regla de detección**

En la creación de la regla disponemos de todas las funcionalidades que permiten añadir una regla con la información relevante vista anteriormente, se puede completar no solo una descripción como también ejemplos o vínculos a otros recursos externos. Se deben de definir también la gravedad y el nivel de riesgo y, por último, definir su periodicidad de ejecución.

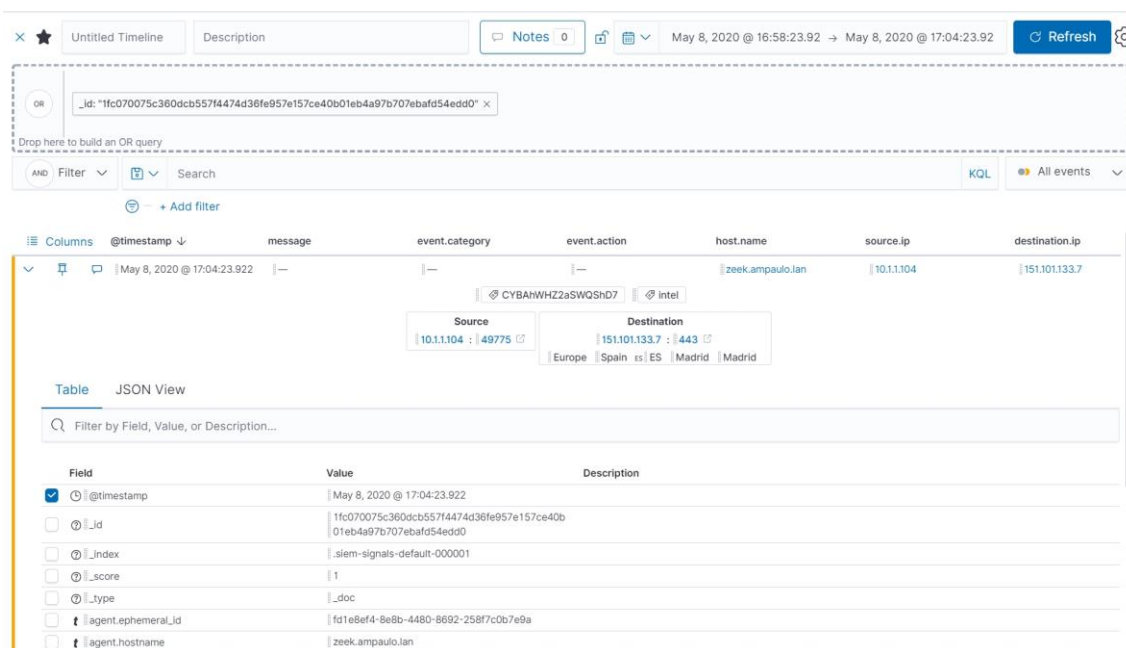
Se han definido reglas para detectar los casos de resultantes de la activación de entradas en las listas de reputación implementadas en la fase 2, además de una regla para los casos de identificación de ficheros mediante YARA que se verá a continuación.

Rule	Risk score	Severity	Last run	Last response
<input type="checkbox"/> IntelStack - detección de Dominio	50	Low	5 minutes ago	succeeded
<input type="checkbox"/> IntelStack - detección de acceso a nodo TOR	50	Medium	4 minutes ago	succeeded
<input type="checkbox"/> YARA - Detección de fichero sospechoso	75	High	4 minutes ago	succeeded
<input type="checkbox"/> IntelStack - detección de Hash de fichero sospechoso	75	High	57 seconds ago	succeeded
<input type="checkbox"/> IntelStack - detección de URL	50	Low	15 seconds ago	succeeded

**Ilustración 50 - Reglas de detección propias**

Se ha visto anteriormente que cuando determinado acceso web se corresponde con una entrada en la lista de reputación del sensor de IntelStack, se añade una entrada al log Intel con la marca **Intel::Domain** o **Intel::URL**. Para la identificación de acceso a nodos TOR la marca es **Intel::ADDR** y para ficheros sospechosos conocidos **Intel::FILE\_HASH**.

La regla "IntelStack – detección de Dominio" se activa cuando se cumple la condición: **zeek.intel.seen.indicator\_type : "Intel::DOMAIN"**, se puede verificar que es posible ejercitar la regla accediendo a alguna de las URLs del fichero de listas de reputación. Se crean reglas similares para las restantes condiciones que se pueden verificar y se verifica que la detención es efectiva.

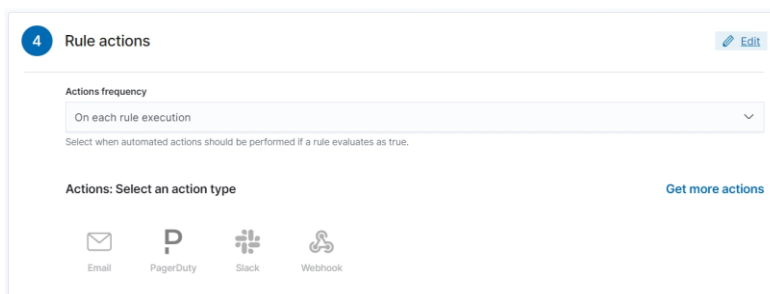


**Ilustración 51 - Activación de la regla: IntelStack - detección de Dominio**

Cada uno de estos incidentes puede ser analizado por separado o en conjunto y gestionado mediante la *timeline*. Se permite añadir notas y dar el incidente por solucionado, cuando corresponda. No se entra en detalle de ese proceso pues esta fuera del alcance del proyecto, pero se destaca la capacidad de navegación desde la señal detectada hasta el evento y posible correlación con otros eventos asociados.

A partir de la versión 7.7 de Elastic se ofrece la posibilidad de añadir a las reglas la opción de notificación, mediante e-mail u otros conectores como

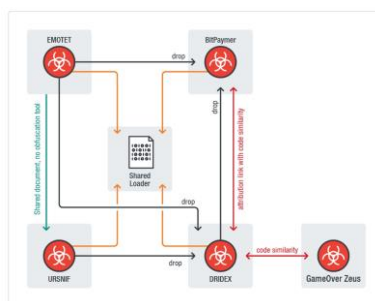
Slack. De todas formas, esta funcionalidad (rule action) no está incluida en la versión gratuita. Se añadirá una funcionalidad alternativa mediante ElastAlert que se configura en apartado posterior y es open-source.



**Ilustración 52 - Elastic SIEM, rule action (ver 7.7)**

Conforme se ha indicado anteriormente en la referencia a la *The Cyber Kill Chain* los eventos detectados en la red normalmente no son casos aislados, un acceso mediante VNC puede una actividad normal, si el uso de la aplicación está autorizado, pero puede ser también una violación de seguridad de un usuario interno o el resultado de un ataque.

Uno de los ataques prevalente en los últimos meses hace uso de la infección inicial, efectuada, por ejemplo, mediante el envío de un e-mail con un fichero que se intenta que el usuario ejecute. De la ejecución de ese fichero resulta la descarga de otros ficheros que permitan tomar el control del equipo o secuestrar el equipo. En el primer caso se tratan de aplicaciones convencionales de acceso remoto como VNC y en el segundo de utilidades de *ransomware* que pretenden encriptar los ficheros del usuario.



**Ilustración 53 - Connections of EMOTET, DRIDEX, URNIF and BitPaymer<sup>34</sup>**

<sup>34</sup><https://blog.trendmicro.com/trendlabs-security-intelligence/ursnif-emotet-dridex-and-bitpaymer-gangs-linked-by-a-similar-loader/>



Por lo tanto, es esencial poder controlar todo este ecosistema ya continuación se añaden los dos elementos en falta, la ingesta de los logs de del firewall y el análisis de los ficheros detectados en la red, mediante reglas YARA.

### 3.2 – Ingesta del log de pfSense en Elasticsearch

Los ataques a las empresas, o bien direccionados o aleatorios incluyen un componente de escaneo de la red procurando identificar oportunidades de acceso y generar conocimiento sobre la red objetivo. El seguimiento de estas actividades en general puede no llevar a ningún destino pues en muchos casos se tratan de escaneos aleatorios automatizados.

Pero, a partir del momento en que algún puerto con un servicio conocido es identificado el tipo y la calidad de los ataques asume un nuevo perfil y la información recogida también tiene un nuevo interés del lado de protección del sistema. La rapidez con que un puerto que haga accesible el servicio de SSH es escaneado y empieza a registrar intentos de acceso es indicador objetivo de los riesgos de exposición de servicios en internet.

Lo que se obtiene con la integración de la ingesta del log de pfSense en particular, pero de cualquier otro firewall en general, es disponer de este tipo de alertas.

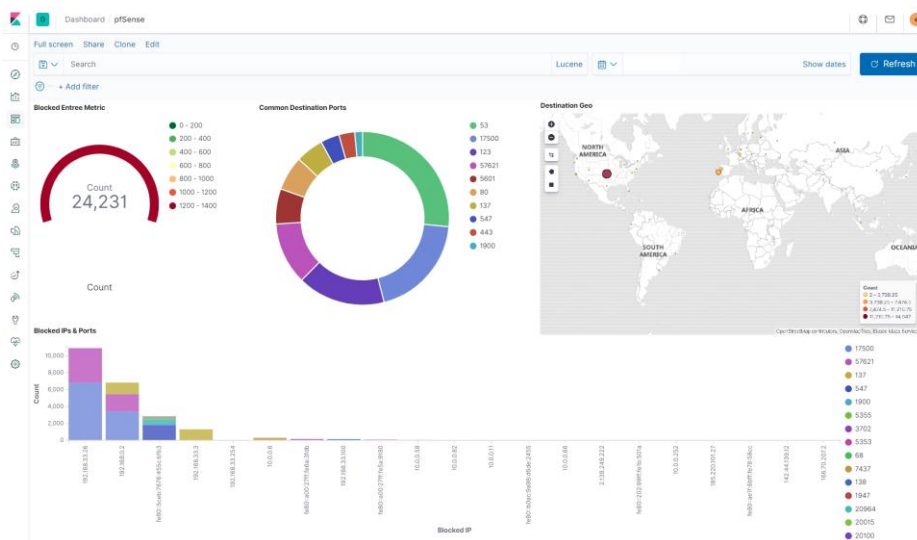


The screenshot shows the 'Signals' interface in Elasticsearch. It displays a list of signals with columns for '@timestamp', 'Rule', 'Version', 'Method', 'Severity', 'Risk Score', and 'event.module'. A single signal is visible, with the 'Rule' column highlighted in red. The signal details are: '@timestamp: May 9, 2020 @ 03:29:12.198', 'Rule: SSH (Secure Shell) from th...', 'Version: 2', 'Method: query', 'Severity: medium', 'Risk Score: 47', and 'event.module: zeek'.

@timestamp	Rule	Version	Method	Severity	Risk Score	event.module
May 9, 2020 @ 03:29:12.198	SSH (Secure Shell) from th...	2	query	medium	47	zeek

**Ilustración 54 - Detección de acceso SSH**

Se podrá también obtener una imagen general complementar a la información que se recoge con Zeek y mediante los módulos de auditoria de los hosts.



**Ilustración 55 - pfSense, Kibana dashboard**

La configuración en general se resume a la activación del log remoto en las opciones de los logs de sistema en pfSense. La ingesta en Elasticsearch se hace mediante Logstash. Logstash se ha configurado para recibir eventos de pfSense siguiendo un script<sup>35</sup> disponible en Github y de uso público.

Por diferencias de versiones de pfSense y también de Logstash se han tenido que hacer algunas correcciones. Además de las instrucciones de configuración y creación del pipeline, está disponible un script de Grok que hace la normalización entre Syslog y los índices de Elasticsearch.

### 3.3 – Yara, Intelligence-Driven Incident Response

Mediante la configuración de las reglas de análisis de los eventos procesados por Zeek, los cuales se han enriquecido con la correlación con los datos de las listas de reputación elegidas, se dispone de información para una monitorización avanzada de la red.

Se pretendía además proceder a la identificación de contenidos que por sus características supongan un riesgo para la seguridad del sistema de información. Con el intuito de cumplir este objetivo se añadieron algunas listas que incluyeran el hash de ficheros. La dificultad en actualizar esas listas ha

<sup>35</sup> <https://github.com/patrickjennings/logstash-pfsense>

llevado a la identificación de alternativas y la investigación condujo a un artículo del SANS Institute, del 2014 pero publicado recientemente.

“**Intelligence-Driven Incident Response with YARA**”<sup>36</sup> es un artículo sobre un escáner de *malware* que permite identificar y clasificar archivos, aplicaciones e inclusive procesos en memoria. El artículo pone el foco en el uso de YARA y de reglas Yara como una aplicación de uso local para analizar ficheros y procesos, específicamente en análisis forense de incidentes.

Sin duda con interés para el objetivo planteado, pero no adecuado en la forma pues se pretende una funcionalidad que pueda actuar de forma automática y genere un resultado que pueda complementar los datos recogidos por las demás partes del proceso.

Siguiendo la investigación se ha identificado una presentación de la conferencia BlackHat del 2019: **Detecting malicious files with YARA rules as they traverse the network.**<sup>37</sup> La presentación y el script disponible en Github<sup>38</sup> han servido de base a la implementación presentada a continuación.

De la solución presentada se mantiene el flujo inicial, que consiste en activar las opciones de Zeek para detectar y extraer ficheros del tráfico de red, los cuales se guardan en una carpeta del proceso, de forma similar al que se hace con los logs. Se añade un script de filtrado para elegir los tipos de ficheros a extraer por Zeek.

El flujo sigue, mediante la activación de un script de Python de forma periódica usando el cron y evaluando cada fichero capturado contra las reglas disponibles. Aquí se han requerido algunas modificaciones en relación con el script original, por un lado, se ha optado por el uso de reglas Yara compiladas previamente y, por otro, se tuvo que cambiar el método de identificación del contexto pues el script original no está desarrollado para logs en formato json.

---

<sup>36</sup><https://www.sans.org/reading-room/whitepapers/forensics/intelligence-driven-incident-response-yara-35542>

<sup>37</sup><https://i.blackhat.com/USA-19/Wednesday/us-19-Bernal-Detecting-Malicious-Files-With-YARA-Rules-As-They-Traverse-The-Network.pdf>

<sup>38</sup> <https://github.com/SCILabsMX/yaraZeekAlert>

El proceso sigue evaluando cada fichero contra el conjunto de reglas e identificando cada uno que devuelva un caso positivo (el mismo fichero puede activar una o más reglas).

Por último, el proceso original tiene como salida el envío de un e-mail de notificación y esa función no era adecuada al objetivo, porque lo que se pretende es añadir la información a Elasticsearch de forma a poder usarla de forma complementar con los restantes logs de zeek. La notificación y alertas se verán en el punto siguiente.

Se ha optado por recoger el resultado de los casos positivos y añadirlos a un fichero de log. La ingesta en Elasticsearch se ha hecho mediante configuración de Filebeat, descrita a continuación, y logrando de esta forma eliminar la dependencia de Logstash y syslog-ng que se había planificado inicialmente, simplificando el despliegue.

Como consecuencia de lograr la ingesta mediante Filebeat, se consigue que los datos del log de resultados de Yara sean parte del índice Filebeat-\*. De esta forma la correlación es inmediata. En relación con las reglas Yara, no se ha entrado en el desarrollo de las reglas pues existen varios repositorios de reglas mantenidos por investigadores en ciberseguridad<sup>39 40</sup>

Empezando por el resultado, se ha creado una estructura de datos donde se definen las propiedades de cada ejecución con identificación positiva. El resultado del método toJSON es la línea que será añadida al fichero de log a procesar.

```
import json

class MatchInfo():

    def __init__(self, ts, fuid, file_path, md5sum, sha1sum, sha256sum,
file_name, generated_zip, file_size, matched_rules, run_time, context):
        self.ts = ts
        self.fuid = fuid
        self.file_path = file_path
        self.md5sum = md5sum
        self.sha1sum = sha1sum
        self.sha256sum = sha256sum
        self.file_name = file_name
        self.generated_zip = generated_zip
        self.file_size = file_size
        self.matched_rules = matched_rules
        self.run_time = run_time
        self.context = context

    def toJSON(self):
        return json.dumps(self, default=lambda o: o.__dict__)
```

#### Ilustración 56 - yaramatch.py

---

<sup>39</sup><https://github.com/Neo23x0/signature-base>

<sup>40</sup><https://github.com/Yara-Rules/rules>

El atributo principal es el identificador de fichero asignado por Zeek a cada elemento extraído y que es parte del nombre del fichero que se guarda. La información detallada sobre la configuración del nombre de fichero y selección de tipos de fichero se verá más adelante.

El identificador de fichero se guarda en el atributo **fuid**, se registran también los nombres de las reglas activadas por el fichero, bien como, demás información relevante para identificación del fichero y explotación posterior.

Todas las propiedades que permiten identificar el contexto del fichero identificado son rellenadas de forma automática pues en *context* se le pasara el evento, en formato Json, que ha dado origen a la extracción de ese fichero.

```
{
  "fuid": "FD2fJL2tWMWGeaGrdh",
  "run_time": 0.4452791213989258,
  "matched_rules": ["fileover_200kb", "pdf_contains_link"],
  "file_name": "HTTP-FD2fJL2tWMWGeaGrdh.pdf",
  "md5sum": "386c39f44af9d23b8aa7b101947d453e",
  "ts": 1590891257.059082,
  "sha256sum": "b4431a3e0923e3d03452f45beb7e197664524d749c9345622ecb68b4545465b",
  "generated_zip": "/opt/zeek/yara/alertedFiles/HTTP-FD2fJL2tWMWGeaGrdh.pdf.zip",
  "file_size": 890298,
  "file_path": "/opt/zeek/extracted/HTTP-FD2fJL2tWMWGeaGrdh.pdf",
  "sha1sum": "93dd3269b60a69bbd64b8cf26c0bd1fb65060d4c",
  "context": {
    "rx_hosts": ["10.1.1.11"],
    "sha1": "93dd3269b60a69bbd64b8cf26c0bd1fb65060d4c",
    "conn_uids": ["CT9pVW3EISJsIw0Qcf"],
    "extracted": "/opt/zeek/extracted/HTTP-FD2fJL2tWMWGeaGrdh.pdf",
    "analyzers": ["SHA1", "MD5", "EXTRACT"],
    "duration": 1.875412940979004,
    "timeout": false,
    "ts": 1590891245.776108,
    "extracted_cutoff": false,
    "source": "HTTP",
    "depth": 0,
    "fuid": "FD2fJL2tWMWGeaGrdh",
    "local_orig": false,
    "is_orig": false,
    "tx_hosts": ["173.236.166.169"],
    "missing_bytes": 0,
    "overflow_bytes": 0,
    "seen_bytes": 953502,
    "md5": "386c39f44af9d23b8aa7b101947d453e",
    "mime_type": "application/pdf",
    "total_bytes": 953502
  }
}
```

### Ilustración 57 - Estructura del log de detecciones Yara

La información en el atributo *context* solo existirá si es posible asociar el fuid con igual identificador en alguno de los logs de zeek actualmente activos. El resultado de la ingesta en Elasticsearch genera los nuevos campos definidos. Es importante en la primera ejecución proceder a la actualización del *index pattern* de Filebeat-\*, de no hacerse es inviable el acceso a los nuevos campos desde las visualizaciones.

Así, cualquier campo que en Discover aparezca con la marca de problema indica que no está declarado en la definición del índice. Este problema se solucionará accediendo a **Management \ Index patterns \ filebeat-\*** y después ejecutar la opción de rehacer el index pattern.

```

f matched_rules      fileover_200kb, pdf_contains_link
f md5sum             386c39f44af9d23b8aa7b101947d453e
f run_time           -0.46893297684699787
f sha1sum            93dd3269b68e69bd64b8cf26c0bd1fb65868e4c
f sha256sum          b4431a3e0923e3d83452f45beb7e197664524d7494c9345622ecb68b4545465b

```

**Ilustración 58 - Aviso de anomalía en index patern**

Posteriormente a esta operación todos los campos estarán disponibles y pueden usarse en consultas y visualizaciones. La configuración que permite con que el fichero json sea convertida en varios campos y no en uno campo con todo el contenido del mensaje, se describe de seguida.

```

f input_type         log
f log_file_path      /var/log/yara/yara.log
# log_offset         6,344
f matched_rules      fileover_200kb, pdf_contains_link
f md5sum             386c39f44af9d23b8aa7b101947d453e
# run_time           0.445
f sha1sum            93dd3269b68e69bd64b8cf26c0bd1fb65868e4c
f sha256sum          b4431a3e0923e3d83452f45beb7e197664524d7494c9345622ecb68b4545465b
f suricata_event_timestamp May 31, 2020 08:14:24.802
f tags               yara
# ts                 1,598,891,257.859

```

**Ilustración 59 - Resultado de la ingesta del log de Yara en Elasticsearch**

Es esencial que el fichero de log sea json bien formado y con un único mensaje por cada línea. Para permitir la correlación es necesario también que exista un campo de relación, conforme indicado anteriormente se trata del atributo *fileid*. Los parámetros de configuración requeridos en el fichero de configuración de Filebeat son:

```
nano /etc/filebeat/filebeat.yml
```

```
filebeat.inputs:
```

```
- type: log
```

```
  enabled: true
```

```
  paths:
```

```
    - /var/log/yara/yara.log
```

```
    - input.json
```

```
  json.keys_under_root: true
```

```
  processors:
```

```
    - decode_json_fields:
```

```
      fields: ["context"]
```

```
    - rename:
```

```

fields:
  - from: "fuid"
    to: "zeek.files.fuid"
  - from: "context.mime_type"
    to: "zeek.files.mime_type"
ignore_missing: false
fail_on_error: true
tags: ["yara"]

```

En general se resume a activar la funcionalidad de proceder a la ingesta de otros logs además de los activados en el módulo Zeek, indicando para ello la ruta del fichero a procesar.

La asignación de un procesador es lo que permite indicar el tipo de tratamiento que se pretende dar al fichero, decalrando el fichero de tipo json (input.json) y con los restantes parámetros se logra que el mensaje sea procesado correctamente de forma automática. Es importante renombrar los campos para permitir la correlación de los datos con la información del log zeek.file.

El proceso ejecutado por el script yaraProcessor.py termina insertando en el log el resultado del proceso de validación de los ficheros, previamente, se guarda una copia del fichero y porque en general las reglas se especifican para la detección de malware, el fichero se guarda en formato zip, protegido mediante password. Convencionalmente se usa la password 'infected' y se sigue esa práctica.

```

def insertLog(message, filepath, pattern, context):
    filename = filepath.split("/")[-1]
    generatedZip = alertedFilesFolder + "/" + filename + ".zip"

    if os.path.isfile(generatedZip):
        os.remove(generatedZip)

    rc = subprocess.call([sevenZipCommand, 'a', '-pinfected', '-y', generatedZip, filepath])
    filesize = os.path.getsize(generatedZip)
    now = time.time()

    fi = MatchInfo(
        ts = now,
        fuid = pattern,
        matched_rules = message[0],
        file_path = str(message[1]),
        md5sum = str(message[2]),
        sha1sum = str(message[3]),
        sha256sum = str(message[4]),
        file_name = filename,
        generated_zip = alertedFilesFolder + "/" + filename + ".zip",
        file_size = filesize,
        run_time = now - start,
        context = json.loads(context))

    logger.info(fi.toJSON())

```

**Ilustración 60 - yaraZProcessor.py, insertLog**

El contexto se obtiene mediante el identificador del fichero y la extracción de toda la línea correspondiente al mensaje procesado.

```
for f in files:
    flog.write("searching in " + f)

    if not archived:
        command = "/bin/cat " + f + " | jq 'select(.fuid == \"" + pattern + "\" )'"
        flog.write("command :" + command)
```

### Ilustración 61 - yaraProcessor.py, contexto

En este punto también se han hecho cambios sobre la implementación inicial pues por el hecho de que los ficheros se guarden en Json no es posible usar el Bro-cut o en alternativa el Zeek-cut.

Se ha identificado como solución óptima el uso de **jq** del cual ya se ha visto el proceso de configuración en la fase 2. Permite hacer la búsqueda mediante cualquier atributo json del fichero y recibir de vuelta todo el registro o el resultado de algún campo en particular.

El nombre del fichero junto con los criterios de tipos de fichero a extraer se ha definido en el fichero (script completo en Anexos):

```
/opt/zeek/share/zeek/policy/frameworks/files/extract-some-files.zeek
```

Se requiere además indicar este fichero como elemento de configuración en local.zeek. A continuación, se presentan las líneas relevantes y se añade el fichero completo como anexo.

```
if ( meta?$mime_type )
    ext = ext_map[meta$mime_type];

local fname = fmt("/opt/zeek/extracted/%s-%s.%s", f$source, f$id, ext);
Files::add_analyzer(f, Files::ANALYZER_EXTRACT, [$extract_filename=fname]);
```

### Ilustración 62 - extract-some-files.zeek

El cambio a efectuar en local.zeek consiste en sustituir el fichero original por el que se ha creado.

```
sudo nano /opt/zeek/share/zeek/site/local.zeek
# Extract Files
@load frameworks/files/extract-some-files
```



Por último, referir que las reglas usadas como ejemplo en el fichero anterior simplemente pretenden demostrar la funcionalidad, por ejemplo, evaluando la cantidad de bytes de un fichero (file\_over200kb.yar) o evaluando determinadas características del fichero (pdf\_contains\_link.yar):

```
rule pdf_contains_link {  
  
  meta:  
    author = "Sean Whalen"  
    last_updated = "2017-06-08"  
    tlp = "white"  
    category = "informational"  
    confidence = "high"  
    description = "A PDFv1.7 that contains a link or external content"  
  
  strings:  
    $pdf_magic = {25 50 44 46}  
    $$anchor_tag = "<a " ascii wide nocase  
    $$uri = /\(http.{,1000}\)/ ascii wide nocase  
  
  condition:  
    $pdf_magic at 0 and any of ($s*)  
}
```

### Ilustración 63 - Ejemplo de regla Yara, pdf\_contains\_link.yar

La versatilidad de Yara permite definir reglas complejas, determinando cuantas condiciones se deben cumplir, etc. La información sobre Yara está disponible en <https://support.virustotal.com/hc/en-us/articles/115002178945-YARA>

El resultado de la configuración desplegada permite crear un dashboard para la inspección de los eventos detectados como se verá adelante.

## 3.4 – Elastalert, procesamiento de alertas en tiempo real

Elastalert<sup>41</sup> es un framework open-source que permite definir reglas para notificación externa y es un proyecto que cuenta con extensa documentación<sup>42</sup>, el número de sistemas de alerta es amplio y cubre casi todas las necesidades, aunque el modelo puede ser expandido si es necesario.

Es posible no solo la notificación clásica, por e-mail, como otras más inmediatas como Twilio o Telegram y una cantidad de integraciones con plataformas de gestión de servicio para creación de incidencias automáticas, etc.

---

<sup>41</sup> <https://github.com/Yelp/elastalert>

<sup>42</sup> <https://elastalert.readthedocs.io/en/latest/index.html>

Los casos de uso planteados son diversos y pueden ser tan sencillos como alertar que se ha generado una señal por activación de una regla conforme explorado en el apartado sobre el SIEM, como otros más complejos, por ejemplo, resultado de la evaluación de una cantidad de condiciones en determinado periodo temporal. El proceso de instalación es sencillo y solo se describen los pasos a continuación, se requiere el uso de Python3.

```
sudo apt-get install python3-pip -y
sudo pip3 install "setuptools>=11.3" PyYAML
sudo pip3 install pytest
```

```
git clone https://github.com/Yelp/elastalert.git
cd elastalert && sudo python3 setup.py install
mkdir bin && mkdir rules
sudo mv /usr/local/bin/elastalert* bin/
cd ../ && sudo mv elastalert/ /etc/
```

Para facilitar la gestión se crea un servicio para ElastAlert

```
sudo nano /etc/systemd/system/elastalert.service

[Unit]
Description=elastalert
After=multi-user.target

[Service]
Type=simple
WorkingDirectory=/etc/elastalert

ExecStart=/etc/elastalert/bin/elastalert --config
/etc/elastalert/config.yaml
StandardOutput=file:/var/log/elastalert.log
StandardError=file:/var/log/elastalert.log
Restart=Always

[Install]
WantedBy=multi-user.target
```

Se habilita el servicio y se prepara la configuración, lo cual se puede hacer mediante el fichero de ejemplo que viene con la solución. Se requiere editar el fichero con las opciones de conexión a Elasticsearch para acceso a los datos e iniciar el servicio.

```
sudo systemctl enable elastalert.service
sudo cp /etc/elastalert/config.yaml.example \
/etc/elastalert/config.yaml
sudo nano /etc/elastalert/config.yaml
sudo systemctl start elastalert
```

Se tendrán que crear los índices en Elasticsearch, lo cual se podrá hacer mediante el comando:

```
sudo /etc/elastalert/bin/elastalert-create-index --host 10.1.1.11 --
port 9200 --ssl --username 'elastic' --password 'elastic' --index
'elastalert_status' --alias 'elastalert_alerts' --old-index '' --url-
prefix '' --no-verify-certs
```

Las reglas se crean en la carpeta /rules, por ejemplo:

```
sudo nano /etc/elastalert/rules/test_rule.yaml
# Type FREQUENCY
name: Example rule
type: frequency
index: filebeat-*
num_events: 5
timeframe:
  hours: 4
filter:
- query:
  query_string:
    query: "network.protocol:dns"
alert:
- debug
alert:
- email
email: "ac@elk.ampaulo.lan"
from_addr: "root@elk.ampaulo.lan"
```

Se puede ejecutar la regla manualmente para inspeccionar la salida del proceso mediante el comando siguiente:

```
cd /etc/elastalert/  
python3 -m elastalert.elastalert --config ./config.yaml --rule  
./rules/test_rule.yaml --debug --verbose
```

Ejercitando las condiciones de la regla se puede verificar que se genera el e-mail de notificación.

```
Content-Type: text/plain; charset="utf-8"  
Content-Transfer-Encoding: base64  
Subject: ElastAlert: Example rule  
  
Example rule  
  
At least 3 events occurred between 2020-05-31 05:06 CEST and 2020-05-31 06:06 CEST  
  
@timestamp: 2020-05-31T04:06:26.682Z  
_id: J7zoaHIBj_5SeKyJ0mzd  
_index: filebeat-7.7.0-2020.05.24-000001  
_type: _doc  
agent: {  
  "ephemeral_id": "e2afca94-a6a7-4263-8f2b-5fe50a1ee225",
```

**Ilustración 64 - ElastAlert, notificación por e-mail (parcial)**

ElastAlert se usará como alternativa a la funcionalidad de *rule action* de Elastic porque esta solo está disponible en las versiones Gold o superior y suscripción.

## 4. Pruebas de validación

Al largo del proyecto se han demostrado los resultados de las funcionalidades desplegadas, como validación de las opciones de proyecto y su traducción en parámetros de configuración. Se pueden considerar como pruebas de integración de la solución.

Las pruebas por realizar en este apartado son pruebas de aceptación de los resultados del proyecto por validación de los objetivos enunciados. Así, se opta por hacer la representación visual de los artefactos añadidos a Kibana como resultado de la integración de las listas de reputación de Intelstack y también de la implementación de las reglas Yara sobre ficheros.

Mediante el acceso a dominios o URLs que constan en la base de datos de IntelStack es posible forzar la activación de las reglas creadas para ese efecto. Lo mismo se puede hacer fácilmente en relación a los nodos de TOR.

@timestamp	Rule	Version	Method	Severity	Risk Score
May 31, 2020 @ 13:08:42.112	IntelStack - detección de acceso a nodo TOR	2	query	medium	50
May 31, 2020 @ 13:08:42.112	IntelStack - detección de acceso a nodo TOR	2	query	medium	50
May 31, 2020 @ 13:08:21.559	IntelStack - detección de Dominio	1	query	low	50
May 31, 2020 @ 13:08:21.559	IntelStack - detección de Dominio	1	query	low	50
May 31, 2020 @ 13:07:48.248	IntelStack - detección de URL	3	query	low	50
May 31, 2020 @ 12:59:09.157	YARA - Detección de fichero sospechoso	5	query	high	75

**Ilustración 65 - SIEM Detections, signals**

La activación de las reglas de File\_Hash no se han podido probar, la cantidad de registros en la base descargada es muy limitada y al tratarse de malware no es (felizmente) fácil encontrarlo accesible en un cualquier enlace. En todo caso se ha optado por descargar ficheros .pcap desde theZoo<sup>43</sup> y de Malware Traffic Analysis<sup>44</sup> y hacer algunas pruebas.

Se ha procedido también a modificar el script de Yara para usar reglas compiladas y se ha descargado el conjunto de reglas del repositorio oficial. El proceso se encuentra documentado en el anexo.

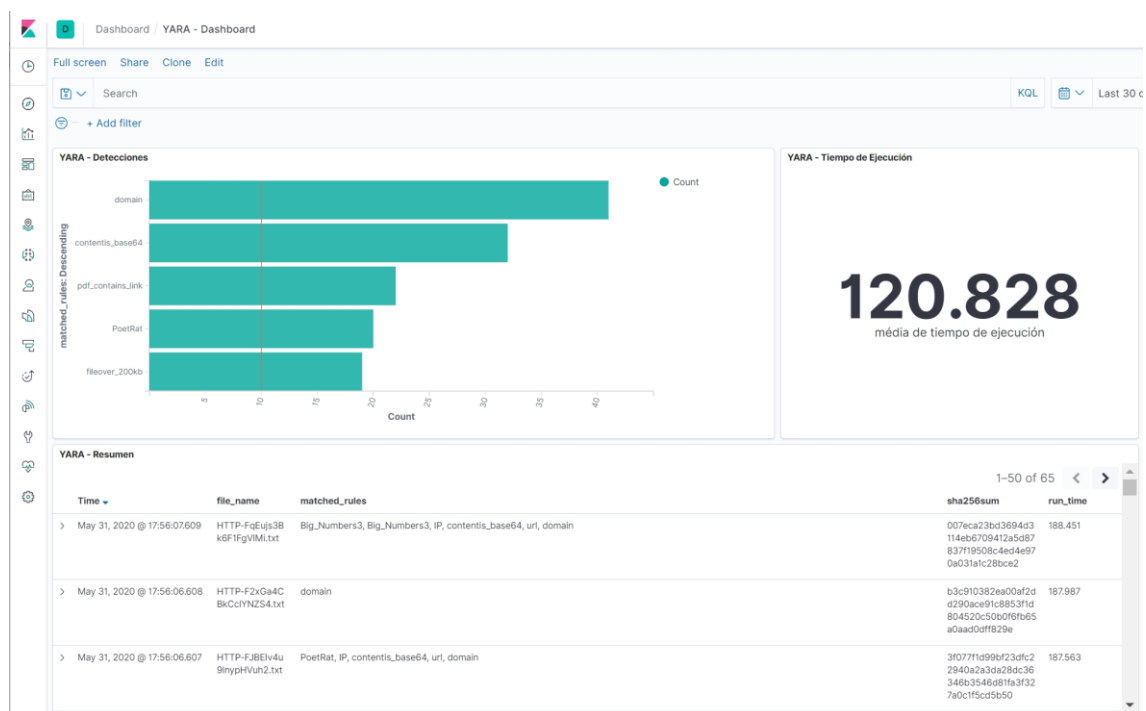
<sup>43</sup> <https://github.com/ytisf/theZoo>

<sup>44</sup> <http://www.malware-traffic-analysis.net/>

<input type="checkbox"/>	log.file.path	/var/log/yara/yara.log
<input type="checkbox"/>	log.offset	38651
<input type="checkbox"/>	matched_rules	PoetRat contentis_base64 domain
<input type="checkbox"/>	md5sum	5a014414b96edc2183523931a8826cc0
<input type="checkbox"/>	run_time	186.1849730014801
<input type="checkbox"/>	sha1sum	3370d2855535d2494fd22f21b1da190e13c83a39
<input type="checkbox"/>	sha256sum	48c9bec83dc9eb0ea7eae65d0c64f4533ea41944a966cbcf185ba1162080a7f

**Ilustración 66 - SIEM Detections, Yara Rules - PoetRAT**

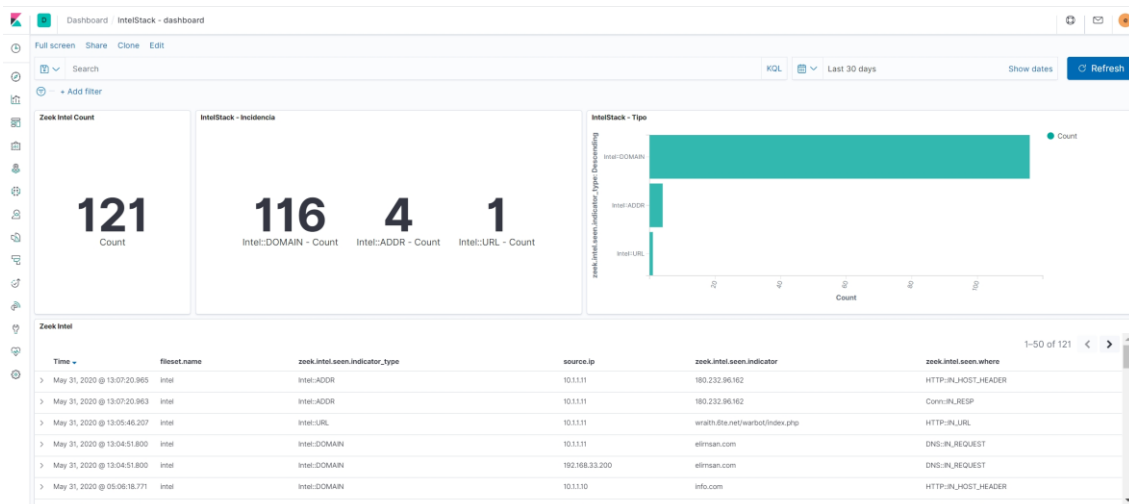
Al usar un conjunto mas completo de reglas y al hacer el tcp replay de ficheros .pcap con malware real se han podido activar diversas reglas de Yara.



**Ilustración 67 - Kibana, Dashboard Yara**

El riesgo de usar malware es alto al no disponer de un equipo aislado para el efecto, por lo tanto, las pruebas se resumen a la validación de la funcionalidad. Se ha desarrollado también un dashboard para el análisis de las activaciones que resultan de las listas de reputación de IntelStack.

La información deberá ser complementada mediante la demás información recogida por Zeek y presentada anteriormente.



**Ilustración 68 - Kibana, Dashboard IntelStack**

El dashboard de IntelStack se ha construido en base a los atributos del log Intel<sup>45</sup> que son recogidos por Zeek.

**intel.log**  
Hits on indicators from the intel framework

Field	Type	Description
ts	time	Timestamp of hit
uid	string	Connection unique id
id	record	ID record with orig/resp host/port. See <a href="#">conn.log</a>
fuid	string	The UID for a file associated with this hit, if any
file_mime_type	string	A mime type if the hit is related to a file
file_desc	string	Additional context for file, if available
seen.indicator	string	The intelligence indicator
seen.indicator_type	string	The type of data the indicator represents
seen.where	string	Where the data was discovered
sources	set	Sources which supplied data for this match

**Ilustración 69 - Zeek, atributos Intel**

En el punto anterior se ha demostrado que las notificaciones mediante ElastAlert envían el mensaje correspondiente en caso de que se cumplan las condiciones especificadas. En la presentación se intentará reproducir un escenario más efectivo, por ejemplo, el envío de notificaciones a Telegram, Twilio o Slack.

<sup>45</sup> [http://gauss.eecs.uc.edu/Courses/c6055/pdf/bro\\_log\\_vars.pdf](http://gauss.eecs.uc.edu/Courses/c6055/pdf/bro_log_vars.pdf)

## 5. Conclusión

Concluido el proceso de despliegue de la solución Zeek y de todo el ecosistema elegido para su implementación, evaluando el proceso de configuración y de pruebas realizado, se puede concluir que los objetivos del proyecto se han conseguido alcanzar.

Revisando ahora los objetivos principales y sus componentes secundarias se pretende generar una imagen final que se pueda comparar con el alcance inicial del proyecto y al mismo tiempo identificar líneas de trabajo que se consideren oportunas para que el proyecto siga avanzando.

El despliegue de Zeek y ELK se ha realizado conforme con las especificaciones inicialmente previstas, pero incluyendo algunas modificaciones que se han mostrado oportunas en el transcurso del trabajo. Al tratarse de un proyecto de investigación sobre una solución sobre la cual no se tenía conocimiento previo relevante, las líneas iniciales eran en cierta medida orientativas.

Los aspectos sobre los cuales se partía con un conocimiento más afianzado, bien por formación como por experiencia, se han conseguido en conformidad con el plan definido en la planificación inicial. Se deben incluir en este conjunto la definición e implementación de la infraestructura de red, activa y pasiva, la instalación y configuración de los servidores y todo el despliegue de la seguridad de red que se ha intentado poner en práctica siguiendo el principio de *seguridad desde el diseño*.

Después de la puesta en marcha de la infraestructura, se ha procedido a la instalación y configuración de Zeek, tratando-se del elemento principal del proyecto se preveía alguna dificultad en función de la complejidad que se ha podido identificar desde el inicio.

En efecto, la configuración de Zeek se ha realizado varias veces a lo largo del proyecto, procurando solucionar los problemas detectados y validar esa solución volviendo a ejecutar el despliegue de raíz. Se puede en base a esto afirmar, con un elevado grado de seguridad, que la documentación del proyecto es fiable y permite lograr los objetivos especificados.



También en lo referente a ElasticStack se debe concluir algo similar, se ha impuesto un cambio de paradigma en la forma de pensar la base de datos con la cual se contaba trabajar. El modelo de ElasticStack era casi totalmente desconocido, más allá del conocimiento teórico de su finalidad. Por lo tanto, el despliegue de ELK exigió estudiar a fondo la solución y al igual que Zeek, proceder a la repetición del proceso repetidas veces.

Siendo la premisa base de este proyecto la seguridad y no obstante algunas dificultades encontradas, se ha optado por no reducir o limitar las especificaciones de seguridad. Por lo tanto, se ha hecho la investigación necesaria para lograr una configuración usando TLS y garantizando las comunicaciones entre los diversos elementos de la solución.

Conseguido el despliegue de los dos elementos base de la solución, Zeek y ELK, se ha validado su funcionamiento procurando asegurar desde temprano que se cumplían ambos objetivos principales, el despliegue de Zeek y su uso para detección de anomalías en la red.

Partiendo de esta posición se ha procedido a la ejecución del primer objetivo complementario que era la integración del sistema con listas de reputación para identificación de malware y otras actividades en la red. En este punto se preveían algunas dificultades en la medida que no se tenía un punto de partida conocido, pero contrariamente al punto anterior el proceso se ha mostrado más fácil de conseguir.

Aun así, se ha comprobado que la simplicidad no necesariamente conduce más directamente al alcance de los objetivos. La funcionalidad y el alcance de IntelStack, la framework elegida para gestión de listas de reputación se ha mostrado estable y completa en lo general, pero no así para el procesamiento de los ficheros que se reciben en la red.

Este hecho ha derivado en la introducción de un objetivo complementario que consistió en la configuración de reglas Yara de forma integrada con Zeek, proyecto que ha tenido más de creación que de despliegue conforme queda documentado en la memoria.

Aun, en cumplimiento de los objetivos propuestos se ha podido desplegar un sistema de notificación y se ha elegido ElastAlert para hacerlo. El desarrollo de las reglas de notificación es claramente una línea de trabajo futura.

En este sentido, la integración de las reglas Yara con Zeek se deben considerar también una prueba de concepto, que se ha demostrado funcional mediante pruebas, pero sin la calidad, estabilidad y escalabilidad requeridas para que puedan ser aplicadas a un despliegue en entorno de producción.

Como objetivos secundarios no alcanzados se pueden identificar dos. En primer lugar, se había considerado como de interés añadir la posibilidad de creación de firmas para protocolos específicos y similar. Es decir, el primer contacto exploratorio con la documentación de Zeek ha llevado hacia la Signature Framework y se ha considerado que sería la forma adecuada de detección de anomalías en algunos protocolos, posiblemente por influencia del conocimiento de otras herramientas como Snort que funcionan a base de firmas.

Las limitaciones temporales son la primera causa por la cual no se ha intentado esa opción, pero en un análisis a posteriori posiblemente se deja caer esa posibilidad. Por otro lado, se había considerado la posibilidad de despliegue de sensores remotos lo cual no ha sido posible desplegar, en este caso la principal causa ha sido la imposibilidad de desplazamiento a razón de la pandemia que se vive actualmente. Es en todo caso una línea que se pretende desarrollar en el futuro.

Del punto de vista personal la gestión de este proyecto, que se ha mostrado como un mixto de investigación, desarrollo e implementación, se debe considerar como muy provechoso y gratificante.

Por un lado, la necesidad de presentar los resultados en Elastic ha exigido aprender con un grado de sofisticación importante a configurar, programar y sobre todo entender el modelo de la solución. En este aspecto una de las principales amenazas de este proyecto, la pandemia que vivimos se ha mostrado también parte de la solución porque Elastic ha hecho disponible de forma gratuita mucha formación en todos los componentes de su solución.

Por otro lado, la capacidad que ofrece este sistema es inmensa y su potencial para el desarrollo de soluciones en mi entorno profesional es muy grande, por lo cual todo el conocimiento adquirido podrá a partir de ahora ser utilizado.

Por último, no quedaría una imagen completa si no fueran identificadas las dificultades más relevantes, más allá de aquellas ya referidas relacionadas con los productos a desplegar.

Desde las primeras semanas del proyecto que la crisis de salud que se ha verificado ha condicionado todos los aspectos de la vida, y en mi caso personal, sobre todo en los meses iniciales ha llevado a un incremento considerable de la carga de actividad profesional.

La imposibilidad de desplazamiento a los centros de empleo ha llevado a casi todos los clientes a optaren por soluciones de teletrabajo con las consecuentes solicitudes de despliegue de soluciones, pero también con inmensas incidencias posteriores por dificultades técnicas o de infraestructura.

Se ha generado por lo tanto una especie de debito de calidad al lago del proyecto que exigió una revisión global en las ultimas semanas y se espera terminada en tiempo.

## 6. Bibliografía

Mike O’Leary - **Cyber Operations: Building, Defending, and Attacking Modern Computer Networks**. 2nd Ed. US: Apress, 2019.

Clinton Gormley and Zachary Tong - **Elasticsearch: The Definitive Guide**. US: Elasticsearch, 2015.

Morey J. Haber, Darran Rolls - **Identity Attack Vectors: Implementing an Effective Identity and Access Management Solution**. US: APress 2020.

Scott J Roberts and Rebekah Brow - **Intelligence-Driven Incident Response**. US: O’Reilly Media, 2017.

Steve Suehring - **Linux firewalls: enhancing security with nftables and beyond**. - Fourth edition. Indiana: Pearson Education, 2015.

Matthew Monte - **Network Attacks & Exploitation, A Framework**. Indiana: John Wiley & Sons, Inc, 2015.

VA - **Security Information and Event Management (SIEM) Implementation**. US: McGraw-Hill, 2011.

Andrew Coburn, Eireann Leverett, Gordon Woo - **Solving cyber risk: protecting your company and society**. New Jersey: John Wiley & Sons, Inc, 2018.

Peter N.M. Hansteen - **The book of PF. A No-Nonsense Guide to the OpenBSD Firewall**. 3rd Ed. San Francisco: No Starch Press, 2015.

Richard Bejtlich - **The Practice of Network Security Monitoring. Understanding Incident Detection and Response**. San Francisco: No Starch Press, 2013

Michael Gregg - **The Network Security Test Lab: A Step-by-Step Guide**. Indianapolis: Wiley, 2015.

## 7. Anexos 1 – Instalación ELK

(Nota posterior al desarrollo de la fase de configuración del proyecto)  
ver punto 7.9 – Instalación ELK – proceso alternativo.

### 7.1 - Instalación y configuración de Elasticsearch

En todos los componentes se procede de la misma forma, se descargan los ficheros para Debian y se efectúa su instalación.

```
curl -L -O
https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-
7.6.2-amd64.deb
sudo dpkg -i elasticsearch-7.6.2-amd64.deb
```

La configuración del servicio, por defecto, no registra en journalctl el log de actividad, para activarlo mientras se hacen pruebas de validación del sistema se modifica el fichero elasticsearch.service:

```
sudo nano /usr/lib/systemd/system/elasticsearch.service
#ExecStart=/usr/share/elasticsearch/bin/systemd-entrypoint -p ${PID_DIR}/elasticsearch.pid --quiet
ExecStart=/usr/share/elasticsearch/bin/systemd-entrypoint -p ${PID_DIR}/elasticsearch.pid
```

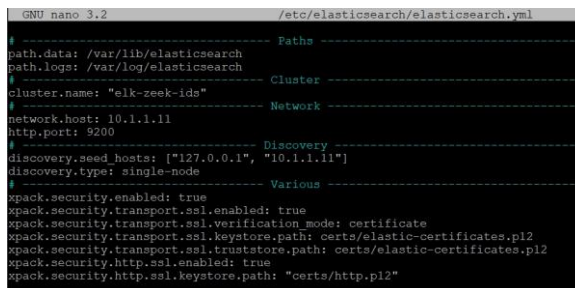
Ilustración 70 - /usr/lib/systemd/system/elasticsearch.service

Se arranca el servicio y se comprueba que se está ejecutando sin errores

```
sudo /bin/systemctl daemon-reload
sudo service elasticsearch start
sudo service elasticsearch status

curl -X GET "localhost:9200/?pretty"
```

Se procede en seguida a la configuración de Elasticsearch y de las opciones de seguridad. Después del proceso documentado, el fichero de configuración quedará así:



```
GNU nano 3.2 /etc/elasticsearch/elasticsearch.yml
# Paths -----
path.data: /var/lib/elasticsearch
path.logs: /var/log/elasticsearch
# Cluster -----
cluster.name: "elk-zeek-ids"
# Network -----
network.host: 10.1.1.11
http.port: 9200
# Discovery -----
discovery.seed_hosts: ["127.0.0.1", "10.1.1.11"]
discovery.type: single-node
# Various -----
xpack.security.enabled: true
xpack.security.transport.ssl.enabled: true
xpack.security.transport.ssl.verification_mode: certificate
xpack.security.transport.ssl.keystore.path: certs/elasticsearch-certificates.p12
xpack.security.transport.ssl.truststore.path: certs/elasticsearch-certificates.p12
xpack.security.http.ssl.enabled: true
xpack.security.http.ssl.keystore.path: "certs/http.p12"
```

Ilustración 71 - /etc/elasticsearch/elasticsearch.yml

## 7.2 – Configuración de Seguridad TLS

Se crea un certificado<sup>46</sup> que es dependiente del nombre del equipo y de su dirección IP, por lo cual se debe validar la configuración.

```
cd /usr/share/elasticsearch/bin/  
sudo ./elasticsearch-certutil ca
```

Por simplificación, se acepta el nombre de CA por defecto y no se define la clave del certificado, en entorno de producción ambos valores deberían definirse adecuadamente. De la ejecución del comando anterior resulta el fichero de Autoridad de Certificación (CA):

```
/usr/share/elasticsearch/elastic-stack-ca.p12
```

A continuación, se genera el certificado privado

```
sudo ./elasticsearch-certutil cert --ca elastic-stack-ca.p12 --dns  
elk.ampaulo.lan --ip 10.1.1.11
```

Se aceptan los valores propuestos y se genera el certificado también sin asignar una clave. Se genera el fichero:

```
/usr/share/elasticsearch/elastic-stack-ca.p12
```

Se configurarán los restantes módulos para que comuniquen con Elasticsearch por HTTPS, por lo cual tendremos que generar una versión distinta del mismo certificado. Elasticsearch lo hace también de forma automática:

```
sudo ./elasticsearch-certutil http
```

En este proceso, a la pregunta si se va a generar un CSR se contestará que no (N), después, a la pregunta de si se va a usar un certificado CA existente se contestará que si (Y). De seguida, hay que pasarle la ruta completa del certificado CA, generado en el primero paso. No se ha definido una clave, por

---

<sup>46</sup><https://www.elastic.co/guide/en/elasticsearch/reference/7.6/configuring-tls.html#node-certificates>

lo cual avanzamos, y debemos elegir la duración temporal del certificado. Se acepta el valor propuesto (5y) - 5 años.

A continuación, se decide si se creará un certificado por nodo o, en alternativa, uno general, en este caso solo hay 1 nodo y se contesta por defecto N. Se definen ahora los datos que se certifican, se introduce el hostname (elk.ampaulo.lan) o varios si coexisten en el servidor y se confirma que son correctos.

En el paso siguiente se introduce el IP asignado al servidor (10.1.1.11) o varios si existen y se confirman los datos. Para finalizar se aceptan los parámetros y los valores propuestos en las preguntas. Como resultado se ha generado un archivo que contiene los certificados que se requieren para configurar Elasticsearch y los demás módulos.

**`/usr/share/elasticsearch/elasticsearch-ssl-http.zip`**

Por último, se procede a activar la seguridad en Elasticsearch, la cual, por defecto, esta desactivada cuando el sistema no tiene una licencia comercial. Se han eliminado todos los parámetros y comentarios no relevantes. Como anexo se incluirá el contenido del fichero yaml en formato texto:

**`sudo nano /etc/elasticsearch/elasticsearch.yml`**

Se crea una carpeta 'certs' en la ruta de configuración de Elasticsearch y se copian ambos ficheros que se encuentran en el archivo .zip generado antes. Es esencial asignar los permisos a la carpeta y ficheros.

**`sudo mkdir /etc/elasticsearch/certs`**

El archivo generado contiene dos conjuntos de ficheros, los que corresponden a Elasticsearch y los que corresponden a la configuración de Kibana. Extraemos solo el fichero http.p12 y copiamos el fichero del certificado privado, ambos para la carpeta 'certs'.

**`cd /usr/share/elasticsearch`**

```
sudo unzip -j elasticsearch-ssl-http.zip elasticsearch/http.p12 -d
"/etc/elasticsearch/certs"
sudo cp elastic-certificates.p12 /etc/elasticsearch/certs
sudo chmod 755 /etc/elasticsearch/certs
sudo chmod 644 /etc/elasticsearch/certs/*
```

Se arranca el servicio y se comprueba que se está ejecutando sin errores

```
sudo service elasticsearch start
sudo service elasticsearch status
```

Si se producen errores se pueden identificar mediante el journal de sistema

```
sudo journalctl -u elasticsearch.service | tail -f -n 50
```

Elasticsearch escucha en el puerto TCP 9200 lo cual hay que añadir a la lista de reglas de nftables

```
sudo nano /etc/nftables.conf
# elastic
tcp dport { 9200 } counter log prefix "[nftables]: " accept
```

Se validan las reglas y se reinicia el servicio nftables.

```
sudo nft -f /etc/nftables.conf
sudo service nftables restart
```

### 7.3 – Configuración de claves de usuarios de sistema

El servicio está configurado y accesible pero solo se permite el acceso de usuarios registrados.

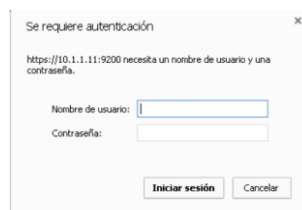


Ilustración 72 - Elasticsearch, autenticación de usuario

Para concluir la configuración de seguridad es necesario asignar claves a los usuarios de sistema y crear los usuarios necesarios. La asignación de claves a los usuarios predefinidos se hace con `elasticsearch-setup-passwords`:



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

```
Changed password for user [apm_system]  
Changed password for user [kibana]  
Changed password for user [logstash_system]  
Changed password for user [beats_system]  
Changed password for user [remote_monitoring_user]  
Changed password for user [elastic]
```

**Ilustración 73 - ELK, usuarios de sistema**

Por simplificar, se asigna como clave el mismo nombre de usuario, aunque en esta fase para todas las funciones se va a usar el usuario elastic que es un super-user. Haciendo login se verifica que el sistema está a funcionar correctamente.

```
curl -k -u elastic:elastic -X GET "https://10.1.1.11:9200"
```

```
{  
  "name" : "elk.ampaulo.lan",  
  "cluster_name" : "elk-zeek-ids",  
  "cluster_uuid" : "0qOtwlG9T56ykYUPbWyLcw",  
  "version" : {  
    "number" : "7.6.2",  
    "build_flavor" : "default",  
    "build_type" : "deb",  
    "build_hash" : "ef48eb35cf30adf4db14086e8aabd07ef6fb113f",  
    "build_date" : "2020-03-26T06:34:37.794943Z",  
    "build_snapshot" : false,  
    "lucene_version" : "8.4.0",  
    "minimum_wire_compatibility_version" : "6.8.0",  
    "minimum_index_compatibility_version" : "6.0.0-beta1"  
  },  
  "tagline" : "You Know, for Search"  
}
```

**Ilustración 74 - Elasticsearch, configuración**

## 7.4 – Instalación y Configuración de Kibana

Se descarga la versión actual desde el repositorio de downloads y se procede a la instalación y configuración.

```
curl -L -O https://artifacts.elastic.co/downloads/kibana/kibana-7.6.2-  
amd64.deb
```

```
sudo dpkg -i kibana-7.6.2-amd64.deb
```

Se arranca el servicio para verificar que se ha instalado correctamente

```
sudo service kibana start
```

```
sudo service kibana status
```

La configuración de seguridad aplicada a Elasticsearch conlleva a la necesidad de ejecutar una configuración más amplia de lo que sería el caso si no se implementara seguridad. Además de definir la conexión a Elasticsearch se requiere configurar las credenciales de acceso y el certificado de CA.

```
GNU nano 3.2 /etc/kibana/kibana.yml
server.port: 5601
server.host: "10.1.1.11"
server.name: "elk.ampaulo.lan"
elasticsearch.hosts: ["https://10.1.1.11:9200"]
elasticsearch.username: "kibana"
elasticsearch.password: "kibana"
elasticsearch.ssl.certificateAuthorities: ["/home/ac/certs/elasticsearch-ca.pem" ]
```

**Ilustración 75 - Kibana, configuración de conexión**

En relación con el certificado de CA se ha optado por crear una localización que será común a Kibana y demás módulos. El certificado para Kibana es de tipo PEM y no es válido el certificado de tipo p12 usado en Elasticsearch. Se procede a copiar el certificado para después arrancar y validar el servicio.

```
sudo mkdir /home/ac/certs
sudo unzip -j elasticsearch-ssl-http.zip kibana/elasticsearch-ca.pem -
d "/home/ac/certs"
```

Se arranca el servicio y se comprueba que se está ejecutando sin errores

```
sudo service kibana start
sudo service kibana status
```

Si se producen errores se pueden identificar mediante el journal de sistema

```
sudo journalctl -u kibana.service | tail -f -n 50
```

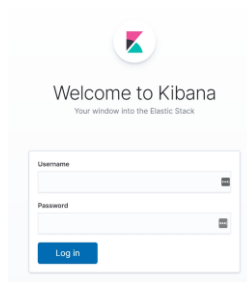
Por último, es necesario configurar el puerto TCP 5601 en el firewall

```
sudo nano /etc/nftables.conf
# kibana
tcp dport { 5601 } counter log prefix "[nftables]: " accept
```

Se validan las reglas y se reinicia el servicio nftables.

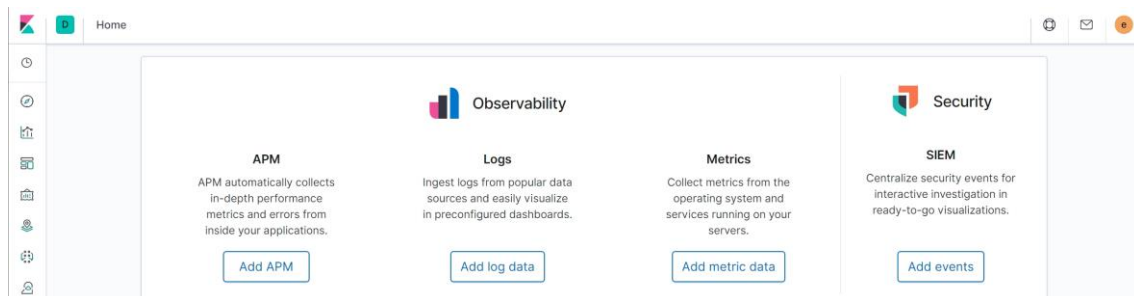
```
sudo nft -f /etc/nftables.conf
sudo service nftables restart
```

La verificación de la configuración se hace mediante un navegador, accediendo a la URL: <http://10.1.1.11:5601>



**Ilustración 76 - Kibana, acceso seguro**

Como resultado de la activación de seguridad solo es posible acceder mediante presentación de usuario y clave. Se accede usando el super-user elastic con clave elastic.



**Ilustración 77 - Kibana, portada**

Se comprueba que el acceso es válido por lo cual la configuración de Kibana e Elasticsearch están validadas.

## 7.5 – Instalación y Configuración de Logstash

Logstash es la L de 'ELK', el módulo que permite normalizar y filtrar la información que se ingesta en Elasticsearch. En una primera fase se opta por tratar la información mediante los Beats que se configuran adelante, pero se instala y configura Logstash para que quede disponible.

```
curl -L -O https://artifacts.elastic.co/downloads/logstash/logstash-7.6.2.deb
```

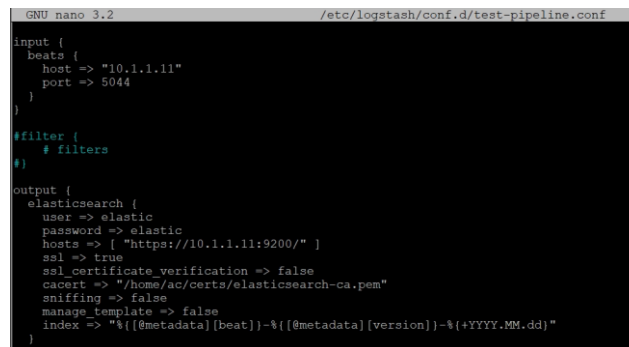
```
sudo dpkg -i logstash-7.6.2.deb
```

La funcionalidad de Logstash se implementa mediante pipelines. Un pipeline corresponde a un fichero de configuración donde se define el tipo de origen de datos y la forma de conexión, es el extremo de entrada.

En el otro extremo, el de salida, se configuran los parámetros de conexión a Elasticsearch, los cuales deben ser conformes a la configuración de seguridad seleccionada. En este caso, se requiere comunicación con encriptación. Se pueden definir filtro para reducir o normalizar la información a enviar a Elasticsearch, la creación de filtros se verá más adelante. Si no se define el apartado filter se envía el documento tal cual se recibe.

```
sudo nano /etc/logstash/conf.d/test-pipeline.conf
```

```
sudo chmod 755 /usr/share/logstash/data
```



```
GNU nano 3.2 /etc/logstash/conf.d/test-pipeline.conf
input {
  beats {
    host => "10.1.1.11"
    port => 5044
  }
}

#filter {
  # filters
}

output {
  elasticsearch {
    user => elastic
    password => elastic
    hosts => [ "https://10.1.1.11:9200/" ]
    ssl => true
    ssl_certificate_verification => false
    cacert => "/home/ac/certs/elasticsearch-ca.pem"
    sniffing => false
    manage_template => false
    index => "%{[@metadata][beat]}-%{[@metadata][version]}-%{+YYYY.MM.dd}"
  }
}
```

**Ilustración 78 - Logstash, configuración pipeline**

Se arranca el servicio y se comprueba que se está ejecutando sin errores

```
sudo service logstash start
```

```
sudo service logstash status
```

Si se producen errores se pueden identificar mediante el 'journal' de sistema

```
sudo journalctl -u logstash.service | tail -f -n 50
```

Por último, es necesario configurar el puerto TCP 5044 en el firewall

```
sudo nano /etc/nftables.conf
```

```
# logstash
```

```
tcp dport { 5044 } counter log prefix "[nftables]: " accept
```

Se validan las reglas y se reinicia el servicio nftables.

```
sudo nft -f /etc/nftables.conf
service nftables restart
```

La validación de la configuración se hace en el punto siguiente.

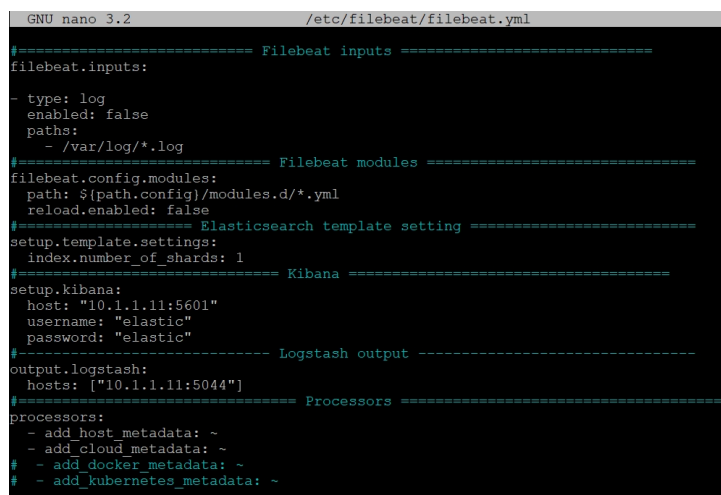
## 7.6 – Instalación y Configuración de Filebeat System

El objetivo de instalar este beat en la VM de Elasticsearch tiene la función de validar la instalación del Logstash.

```
curl -L -O
https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.6.2-
amd64.deb
sudo dpkg -i filebeat-7.6.2-amd64.deb
```

El módulo System de Filebeat envía a Elasticsearch las métricas del sistema donde está configurado. Procedemos a su activación y configuración de las propiedades de conexión.

```
sudo filebeat modules enable system
sudo nano /etc/filebeat/filebeat.yml
```



```
GNU nano 3.2 /etc/filebeat/filebeat.yml
#----- Filebeat inputs -----
filebeat.inputs:
- type: log
  enabled: false
  paths:
    - /var/log/*.log
#----- Filebeat modules -----
filebeat.config.modules:
  path: ${path.config}/modules.d/*.yml
  reload.enabled: false
#----- Elasticsearch template setting -----
setup.template.settings:
  index.number_of_shards: 1
#----- Kibana -----
setup.kibana:
  host: "10.1.1.11:5601"
  username: "elastic"
  password: "elastic"
#----- Logstash output -----
output.logstash:
  hosts: ["10.1.1.11:5044"]
#----- Processors -----
processors:
- add_host_metadata: ~
- add_cloud_metadata: ~
# - add_docker_metadata: ~
# - add_kubernetes_metadata: ~
```

Ilustración 79 - /etc/filebeat/filebeat.yml

Se puede observar que la conexión de salida es Logstash y por lo tanto es una conexión sin encriptación del tráfico. La configuración del módulo no requiere parametrización

```
sudo nano /etc/filebeat/modules.d/system.yml
```

Se procede a la reconfiguración, inicio y comprobación del estado del servicio.

```
sudo filebeat setup -e
sudo filebeat setup --dashboards
sudo service filebeat start
sudo service filebeat status
```

Si se producen errores se pueden identificar mediante el journal de sistema

```
sudo journalctl -u filebeat.service | tail -f -n 50
```

Después de la configuración del clúster y probada la comunicación entre sus módulos se configuran los elementos que harán la ingesta de los logs. Se configura el Metricbeat de forma a poder monitorizar la infraestructura

## 7.7 – Instalación y Configuración de Metricbeat

```
curl -L -O
https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-
7.6.2-amd64.deb
sudo dpkg -i metricbeat-7.6.2-amd64.deb
```

Terminada la instalación se procede a la configuración, la cual es similar al visto anteriormente, pero en este caso la salida se hace directamente hacia Elasticsearch, por lo cual, es necesario asignar la información de seguridad.

```
sudo nano /etc/metricbeat/metricbeat.yml
```

```
GNU nano 3.2 /etc/metricbeat/metricbeat.yml

#==== Modules configuration ====
metricbeat.config.modules:
  # Glob pattern for configuration loading
  path: ${path.config}/modules.d/*.yaml
  reload.enabled: false
  #reload.period: 10s

#==== Elasticsearch template setting ====
setup.template.settings:
  index.number_of_shards: 1
  index.codec: best_compression
  #_source.enabled: false

#==== Kibana ====
setup.kibana:
  host: "10.1.1.11:5601"

#----- Elasticsearch output -----
output.elasticsearch:
  hosts: ["https://10.1.1.11:9200"]
  ssl.certificate_authorities: ["/home/ac/certs/elasticsearch-ca.pem"]
  username: "elastic"
  password: "elastic"

#==== Processors ====
processors:
  - add_host_metadata: ~
  - add_cloud_metadata: ~
  #- add_docker_metadata: ~
  #- add_kubernetes_metadata: ~

xpack.security.enabled: true
```

Ilustración 80 - /etc/metricbeat/metricbeat.yml

A continuación, se activan y configuran los módulos que se pretenden monitorizar, el listado de módulos se puede encontrar en la página del producto, los módulos disponibles en la versión instalada se encuentran en la carpeta: /etc/metricbeat/modules.d/ y se pueden listar con el comando

```
sudo metricbeat modules list
```

Cada módulo permite monitorizar un servicio, contenedor, etc. el objetivo que se pretende es monitorizar los elementos del clúster. Así, se procede a la activación y configuración de los módulos system y elasticsearch-xpack. Después, desde Kibana el sistema nos indicará otros que pueden ser de interés.

```
sudo metricbeat modules enable system
```

```
sudo metricbeat modules enable elasticsearch-xpack
```

De seguida se procede a la configuración de cada módulo, el fichero yaml de configuración de cada uno corresponde al nombre del módulo con extensión yaml.

**sudo nano /etc/metricbeat/modules.d/system.yml**

```
GNU nano 3.2 /etc/metricbeat/modules.d/system.yml
# Module: system
# Docs: https://www.elastic.co/guide/en/beats/metricbeat/7.6/metricbeat-module-system.html
- module: system
  period: 10s
  metricssets:
    - cpu
    - load
    - memory
    - network
    - process
    - process_summary
    - socket_summary
    #- entropy
    #- core
    #- diskio
    #- socket
    #- service
  process.include_top_n:
    by_cpu: 5 # include top 5 processes by CPU
    by_memory: 5 # include top 5 processes by memory
```

**Ilustración 81 - /etc/metricbeat/modules.d/elasticsearch-xpack.yml**

Se pueden activar/desactivar las métricas y se puede consultar la documentación a partir de la URL al inicio de los ficheros de configuración. Algunos de los módulos requieren además de la definición de métricas de interés, la especificación de los datos de conexión a Elasticsearch.

**sudo nano /etc/metricbeat/modules.d/elasticsearch-xpack.yml**

```
GNU nano 3.2 /etc/metricbeat/modules.d/elasticsearch-xpack.yml
# Module: elasticsearch
# Docs: https://www.elastic.co/guide/en/beats/metricbeat/7.6/metricbeat-module-elasticsearch.html
- module: elasticsearch
  metricssets:
    - ccr
    - cluster_stats
    - enrich
    - index
    - index_recovery
    - index_summary
    - ml_job
    - node_stats
    - shard
  period: 10s
  hosts: ["https://10.1.1.11:9200"]
  username: "elastic"
  password: "elastic"
  ssl.certificate_authorities: ["/home/ac/certs/elasticsearch-ca.pem"]
  xpack.enabled: true
```

**Ilustración 82 - /etc/metricbeat/modules.d/elasticsearch-xpack.yml**

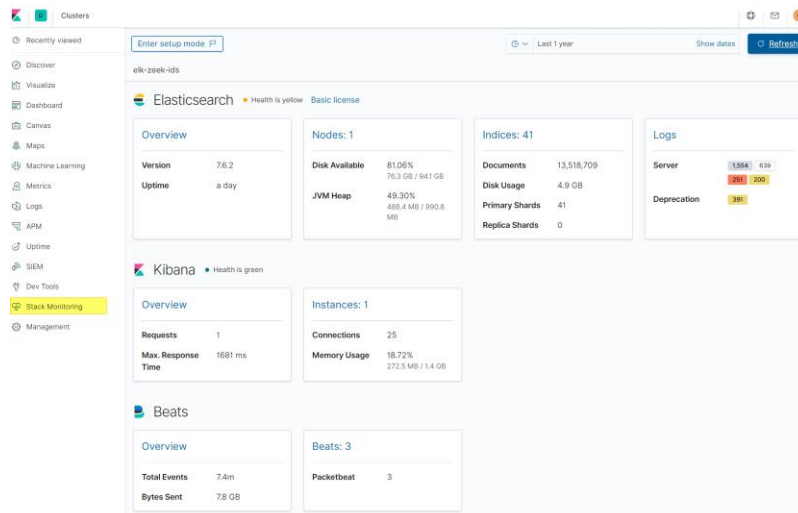
Se comprueba la configuración y se inicia el beat.

**sudo metricbeat test config**

**sudo service metricbeat start**

En el menú Stack Monitoring podemos verificar que todo está correctamente configurado y empezar a ajustar la monitorización de los recursos. Se verifica que Elasticsearch está ahora configurado para ser monitorizado y se procede a realizar la configuración de los restantes módulos.





**Ilustración 83 - Dashboard, Stack Monitoring**

El proceso es similar al anterior por lo cual únicamente se listan los pasos a ejecutar.

```
sudo metricbeat modules enable kibana-xpack
```

```
sudo nano /etc/metricbeat/modules.d/kibana-xpack.yml
```

Se requiere la información de conexión a Kibana, incluyendo usuario y clave.

```
sudo metricbeat setup -e
```

```
GNU nano 3.2 /etc/metricbeat/modules.d/kibana-xpack.yml
# Module: kibana
# Docs: https://www.elastic.co/guide/en/beats/metricbeat/7.6/metricbeat-module-kibana.html

- module: kibana
  metricsets:
    - stats
    period: 10s
    hosts: ["10.1.1.11:5601"]
    #basepath: ""
    username: "kibana"
    password: "kibana"
    xpack.enabled: true
```

**Ilustración 84 - /etc/metricbeat/modules.d/kibana-xpack.yml**

La supervisión de los Beats es una de las funcionalidades que se pretenden tener disponibles para poder desplegar la solución en VPNs conectadas al clúster central. La monitorización es la forma elegida para controlar la disponibilidad y los recursos de los sensores desplegados.

Status	Nodes	Indices	JVM Heap	Total shards	Unassigned shards	Documents	Data
● Yellow	1	41	625.5 MB / 990.8 MB	56	15	13,529,447	4.9 GB

Name ↑	Status	Shards	CPU Usage	Load Average	JVM Heap	Disk Free Space
★ elk.ampaulo.lan 10.1.1.11:9300	● Online	41	27% ↓ 84% max 27% min	8.13 ↑ 10.06 max 4.26 min	81% ↓ 82% max 80% min	77.9 GB ↑ 79.3 GB max 70.0 GB min

Rows per page: 20 ▾ < 1 >

Ilustración 85 - Dashboard, Elasticsearch

## 7.8 – Instalación y Configuración de Packetbeat y HTTP endpoint

Se instalará Packetbeat en todos los nodos y sensores del clúster con el objetivo de monitorizar los demás beats instalados.

```
curl -L -O
```

```
https://artifacts.elastic.co/downloads/beats/packetbeat/packetbeat-7.6.2-amd64.deb
```

```
sudo dpkg -i packetbeat-7.6.2-amd64.deb
```

Se edita el fichero de configuración a ejemplo de los anteriores y se crea un HTTP endpoint (solo local) que el beat Metricbeat usara como endpoint. La definición se debe copiar desde /etc/packetbeat/packetbeat.reference.yml

```
sudo nano /etc/packetbeat/packetbeat.yml
```

```
##### HTTP Endpoint #####
# Each beat can expose internal metrics through a HTTP endpoint. For security
# reasons the endpoint is disabled by default. This feature is currently experimental.
# Stats can be access through http://localhost:5066/stats . For pretty JSON output
# append ?pretty to the URL.
#
# Defines if the HTTP endpoint is enabled.
http.enabled: true
#
# The HTTP endpoint will bind to this hostname, IP address, unix socket or named pipe.
# When using IP addresses, it is recommended to only use localhost.
http.host: localhost
#
# Port on which the HTTP endpoint will bind. Default is 5066.
http.port: 5066
#
# Define which user should be owning the named pipe.
#http.named_pipe.user:
#
# Define which the permissions that should be applied to the named pipe, use the Security
# Descriptor Definition Language (SDDL) to define the permission. This option cannot be used with
# 'http.user'.
#http.named_pipe.security_descriptor:
```

Ilustración 86 - /etc/packetbeat/packetbeat.yml

Después de inicializado el servicio se puede probar mediante el comando siguiente y si todo está correcto se recibe la información del endpoint. Al tratarse de un servicio local no se requieren cambios en el firewall.

```
sudo service packetbeat start
```

```
curl -XGET 'localhost:5066/?pretty'
```

```
"beat": "packetbeat",  
"hostname": "zeek.ampaulo.lan",  
"name": "zeek.ampaulo.lan",  
"uuid": "cda85231-5d81-47d2-9074-3f5b2a097c17",  
"version": "7.6.2"
```

**Ilustración 87 - comprobación del HTTP endpoint**

Necesariamente hay que configurar este servicio en cada nodo en que se pretende monitorizar los elementos del clúster. En metricbeat se activa el control de beats. La configuración se deja por defecto y se reinicia el servicio

```
sudo metricbeat modules enable beat-xpack
```

```
sudo nano /etc/metricbeat/modules.d/beat-xpack.yml
```

```
GNU nano 3.2 /etc/metricbeat/modules.d/beat-xpack.yml  
# Module: beat  
# Docs: https://www.elastic.co/guide/en/beats/metricbeat/7.6/metricbeat-module-beat.html  
  
- module: beat  
  metricsets:  
    - stats  
    - state  
  period: 10s  
  hosts: ["http://localhost:5066"]  
  #username: "user"  
  #password: "secret"  
  xpack.enabled: true
```

**Ilustración 88 - /etc/metricbeat/modules.d/beat-xpack.yml**

```
sudo service packetbeat restart
```

La configuración de Heartbeat y Auditbeat es similar a las anteriores y su funcionalidad en el ecosistema desarrollado ya se ha justificado en el contexto de implementación del SIEM. Se listan los recursos, y ficheros de configuración relevante pero no se procederá a documentar la instalación. Para las opciones de supervisión de servicios y otros dispositivos se puede consultar:

<https://www.elastic.co/guide/en/beats/heartbeat/current/heartbeat-getting-started.html>

```
curl -L -O
```

```
https://artifacts.elastic.co/downloads/beats/heartbeat/heartbeat-  
7.6.2-amd64.deb
```

```
sudo dpkg -i heartbeat-7.6.2-amd64.deb
```

```
nano /etc/heartbeat/heartbeat.yml
```

El dashboard para heartbeat a fecha del proyecto se tiene que configurar manualmente. Se descarga la configuración del dashboard desde el proyecto

uptime-contrib<sup>47</sup> y se añade a Kibana mediante la importación de objetos (Menu Management, Saved Objects, Import).

```
sudo service heartbeat-elastic start
```

Auditbeat, el módulo especializado en la recolección e ingesta de información de auditoría de los nodos, se instala y configura siguiendo el mismo proceso.

```
curl -L -O
https://artifacts.elastic.co/downloads/beats/auditbeat/auditbeat-7.6.2-amd64.deb
sudo dpkg -i auditbeat-7.6.2-amd64.deb
nano /etc/ auditbeat / auditbeat.yml
./auditbeat setup --dashboards
sudo service auditbeat start
```

Se terminará la configuración de ELK activando los servicios de los módulos instalados de forma a que se inicien de forma automática.

```
sudo systemctl enable <nombre del módulo>
```

---

<sup>47</sup> <https://github.com/elastic/uptime-contrib>

## 7.9 – Instalación ELK – proceso alternativo y actualización de versión

Posteriormente al desarrollo del proyecto sobre la versión 7.6.2 de ELK, Elastic ha liberado la versión 7.7 con algunas correcciones y mejoras, de entre ellas se destaca la opción de generar y gestionar casos en el SIEM.

Incluir la integración de una herramienta externa con esa funcionalidad estaba fuera del alcance del proyecto por viabilidad de tiempo, pero su necesidad es indiscutible. Así, se ha procedido a la actualización de la infraestructura para poder disponer de ella. En este proceso se ha verificado que los pasos seguidos de acuerdo con la documentación de instalación han hecho el proceso de actualización más complejo que lo necesario.

Como forma alternativa de instalación, con la salvaguarda que solo es viable en las distribuciones que mantienen los repositorios actualizados, se ha optado por añadir el repositorio a lista de fuentes. Añadida la firma de seguridad y el repositorio la instalación se puede hacer normalmente mediante apt-get.

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo
apt-key add -
sudo apt-get install apt-transport-https
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" |
sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
```

```
sudo apt-get update && sudo apt-get install <aplicación>
```

Donde <aplicación> es cada uno de los módulos de Elastic y todos los pasos de configuración necesarios son los mismos indicados anteriormente. En este caso es importante confirmar el nombre de las aplicaciones, por ejemplo, heartbeat es un paquete distinto aquel que se pretende configurar y que hay que seleccionarlo en el repositorio mediante **heartbeat-elastic**.

```
sudo apt-get update && sudo apt-get install heartbeat-elastic
sudo systemctl enable heartbeat-elastic
```

```
sudo nano /etc/heartbeat/heartbeat.yml
sudo heartbeat setup
sudo service heartbeat-elastic start
```

## 8. Anexos 2 – Instalación YARA

### 8.1 - Instalación y configuración de Yara

Se procede a la instalación siguiendo los pasos indicados en la documentación oficial<sup>48</sup>. El fichero de instalación se descargará desde VirusTotal<sup>49</sup> en GitHub, la versión disponible a fecha de proceso es la reléase 4.0.1.

En el servidor zeek y usando el usuario root se ha procedido a ejecutar los comandos normales para descargar las fuentes y compilar la aplicación.

```
curl -L https://github.com/VirusTotal/yara/archive/v4.0.1.tar.gz -o yara-4.0.1.tar.gz
```

```
apt-get install automake libtool make gcc pkg-config libmagic-dev
tar -zxf yara-4.0.1.tar.gz
cd yara-4.0.1
./bootstrap.sh
./configure --enable-magic
make
make install
```

Ejecutamos también el comando siguiente para evitar errores de ejecución

```
sudo sh -c 'echo "/usr/local/lib" >> /etc/ld.so.conf'
sudo ldconfig
```

---

<sup>48</sup> <https://yara.readthedocs.io/en/stable/gettingstarted.html>

<sup>49</sup> <https://github.com/VirusTotal/yara/releases>

## 8.2 – Yara, configuración de integración con Zeek

Se añade un script para selección de los ficheros a extraer por Zeek, además se formatea el nombre de los ficheros extraídos para facilitar su identificación visual.

```
sudo nano \  
/opt/zeek/share/zeek/policy/frameworks/files/extract-some-files.zeek
```

```
global ext_map: table[string] of string = {  
  ["application/x-dosexec"] = "exe",  
  ["text/plain"] = "txt",  
  ["text/html"] = "html",  
  ["application/zip"] = "zip",  
  ["application/x-7z-compressed"] = "7z",  
  ["application/x-rar"] = "rar",  
  ["application/x-rar-compressed"] = "rar",  
  ["application/xdmg"] = "dmg",  
  ["application/msword"] = "doc", ["application/msexcel"] = "xls",  
  ["application/mspowerpoint"] = "ppt",  
  ["application/vnd.openxmlformats-officedocument.wordprocessingml.document"] = "docx",  
  ["application/vnd.openxmlformats-officedocument.spreadsheetml.sheet"] = "xlsx",  
  ["application/vnd.openxmlformats-officedocument.presentationml.presentation"] = "pptx",  
  ["application/pdf"] = "pdf", ["text/rtf"] = "rtf",  
} &default = "";  
  
event file_sniff(f: fa_file, meta: fa_metadata)  
{ if (! meta?$mime_type )  
  return;  
  
  if ( ! ( meta?$mime_type == "application/x-dosexec"  
  || meta?$mime_type == "text/plain"  
  || meta?$mime_type == "text/html"  
  || meta?$mime_type == "application/xdmg"  
  || meta?$mime_type == "application/zip"  
  || meta?$mime_type == "application/x-7z-compressed"  
  || meta?$mime_type == "application/x-rar"  
  || meta?$mime_type == "application/x-rar-compressed"  
  || meta?$mime_type == "application/msword"  
  || meta?$mime_type == "application/msexcel"  
  || meta?$mime_type == "application/mspowerpoint"  
  || meta?$mime_type == "application/vnd.openxmlformats-officedocument.wordprocessingml.document"  
  || meta?$mime_type == "application/vnd.openxmlformats-officedocument.spreadsheetml.sheet"  
  || meta?$mime_type == "application/vnd.openxmlformats-officedocument.presentationml.presentation"  
  || meta?$mime_type == "text/rtf"  
  || meta?$mime_type == "application/pdf") )  
    return;  
  
  local ext = "";  
  
  if ( meta?$mime_type )  
    ext = ext_map[meta?$mime_type];  
  
  local fname = fmt("/opt/zeek/extracted/%s-%s.%s", f$source, f$id, ext);  
  
  Files::add_analyzer(f, Files::ANALYZER_EXTRACT, [$extract_filename=fname]);  
}
```

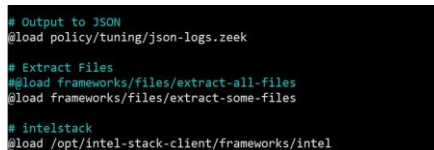
### Ilustración 89 - script de extracción de ficheros

Los ficheros extraídos se guardarán en **/opt/zeek/extracted** por lo cual se crea y se asignan los permisos a esa carpeta.

```
mkdir /opt/zeek/extracted  
chown -R zeek:zeek /opt/zeek/extracted  
chmod 740 /opt/zeek/extracted
```

A continuación, hacemos la modificación necesaria en el fichero de configuración de Zeek, sustituyendo el script genérico por el que se ha creado anteriormente.

```
nano /opt/zeek/share/zeek/site/local.zeek
```



```
# Output to JSON
@load policy/tuning/json-logs.zeek

# Extract Files
#load frameworks/files/extract-all-files
#load frameworks/files/extract-some-files

# intelstack
@load /opt/intel-stack-client/frameworks/intel
```

Ilustración 90 - Yara, modificación en local.zeek

Se procede a la configuración del script de Python que hará la verificación de los ficheros extraídos, incluyendo la clase necesaria para generar él log en formato json.

```
mkdir /opt/zeek/yara
```

```
nano /opt/zeek/yara/yaramatch.py
```

En la carpeta `/opt/zeek/yara/` se añade también el script **yaraProcessor.py** que procederá al procesamiento de los ficheros extraídos, para ello se le asignan los permisos correspondientes.

```
nano /opt/zeek/yara/yaraProcessor.py
```

```
chmod u+x /opt/zeek/yara/yaraProcessor.py
```

El resultado de la ejecución del script, cuando se produce un positivo es el registro del evento en el fichero de log **/var/log/yara/yara.log**, lo cual posteriormente se procesa mediante Filebeat.

Con el objetivo de validar todo el proceso se va a crear una regla que evalúa el tamaño de un fichero cualquiera y si el fichero superar los 200 Kb se genera la alerta correspondiente

```
mkdir /opt/zeek/yara/rules
```

```
nano file_over200kb.yar
```

```
rule fileover_200kb {
    condition: filesize > 200KB
}
```



Se generan dos peticiones sobre las cuales Zeek extrae los ficheros correspondientes lo que permite verificar también que la configuración es correcta. En /opt/zeek/extracted se han creado dos ficheros:

```
-rw-r--r-- 1 zeek zeek 12522 May 30 19:56 HTTP-F0zgwu4EetK43cXq19.html
-rw-r--r-- 1 zeek zeek 433994 May 30 20:24 HTTP-FvZK5P3YyeoMnHCzf4.pdf

yara /opt/zeek/yara/rules/file_over200kb.yar HTTP-F0zgwu4EetK43cXq19.html
yara /opt/zeek/yara/rules/file_over200kb.yar HTTP-FvZK5P3YyeoMnHCzf4.pdf
fileover_200kb HTTP-FvZK5P3YyeoMnHCzf4.pdf
```

### Ilustración 91 - Validación de reglas Yara

Se procede a ejercitar manualmente la regla creada sobre cada uno de ellos y se puede verificar que solo es activada en el caso del fichero PDF pues es el único que cumple la condición de tamaño > 200 kb.

## 8.3 – Yara, automatización mediante script de Python

La ejecución periódica de las reglas Yara sobre los ficheros extraídos por Zeek se hace mediante un script explorado en el capítulo 3. Se indican los pasos de configuración a efectuar para proceder a su automatización. La instalación de p7zip es obligatoria pues se usa en el script desarrollado.

```
mkdir /var/log/yara
mkdir /opt/zeek/yara/alertedFiles
apt-get install p7zip-full
```

Concluido el proceso de configuración se definen los permisos y se activa el script de ejecución.

```
sudo chown -R zeek:zeek /var/log/yara
sudo chmod 740 /var/log/yara
sudo chown -R zeek:zeek /tmp/yaraAllRules
sudo chmod 740 /tmp/yaraAllRules
```

```
crontab -e
*/1 * * * * cd /opt/zeek/yara/ && /usr/bin/Python
/opt/zeek/yara/yaraProcessor.py
```

## 8.4 – Yara, compilación de reglas

El script para compilación de las reglas Yara requiere Python3 y pip, Python ya se encuentra instalado en el sistema, pero es necesario proceder a la instalación de pip3.

```
sudo apt-get install python3-pip -y
sudo pip3 install "setuptools>=11.3" PyYAML
sudo pip3 install pytest
```

El acceso a Yara desde Python requiere la instalación de yara-python<sup>50</sup> se instalará mediante pip3 y posteriormente se descargará el script pyarascanner<sup>51</sup> que se usará para compilar las reglas. Se opta por usar Git y hacer clone del proyecto.

```
pip3 install yara-python
cd /opt/zeek/yara
git clone https://github.com/nogoodconfig/pyarascanner.git
```

A continuación, se descarga un conjunto de reglas<sup>52</sup>, también en este caso haciendo el clone del repositorio. Aunque existan varios repositorios de reglas se opta por el oficial.

```
git clone https://github.com/Yara-Rules/rules.git
cd rules
cp -R * /opt/zeek/yara/rules/
cd /opt/zeek/yara/rules/
```

Se procede después a la compilación de las reglas mediante el script descargado antes.

```
cd /opt/zeek/yara/pyarascanner
python3 pyarascanner.py /opt/zeek/yara/rules/ .
```

---

<sup>50</sup> <https://github.com/VirusTotal/yara-python>

<sup>51</sup> <https://github.com/nogoodconfig/pyarascanner>

<sup>52</sup> <https://github.com/Yara-Rules/rules>

El proceso genera un fichero con el nombre **compiled\_yara\_rules.rules** que se copiará a la ruta donde el script yaraProcessor.py lo referencia, se ha optado por copiarlo a la carpeta rules.

```
cp compiled_yara_rules.rules /opt/zeek/yara/rules/
```

Es importante no olvidar que las reglas compiladas solo son validas para la versión de Yara con la cual se han compilado, se puede comprobar que el resultado es valido ejecutando yara manualmente pasando las reglas como parámetro y un eventual fichero a evaluar.

```
yara -r -C compiled_yara_rules.rules /opt/zeek/extracted/HTTP-FWHg7238jM5ROfNUS5.exe
```

Por último, se va a modificar el script yaraProcessor.py para que haga la verificación usando las reglas compiladas comentando la opción existente y añadiendo la instrucción con el modificador -C y el fichero de reglas a usar

```
start = time.time()
#scanOutput = subprocess.check_output("yara -r /tmp/yaraAllRules " + extractedFilePath + " -d
extension=\"noext\" -d filename=\"nofilename\" -d filepath=\"nofilepath\" -d filetype=\"nofiletype\"",
shell=True)
scanOutput = subprocess.check_output("yara -r -C /opt/zeek/yara/rules/compiled_yara_rules.rules " +
extractedFilePath , shell=True)
end = time.time()
```

#### Ilustración 92 - YaraProcessor.py opción de uso de reglas compiladas