Trabajo Fin de Grado
Ingeniería de las Tecnologías de Telecomunicación

Análisis de la capacidad de detección e identificación de Malware basado en tráfico de red mediante IoC

Autor: F. Javier Ros Raposo

Tutor: F. Javier Muñoz Calle

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

Sevilla, 2023

Trabajo Fin de Grado

Ingeniería de las Tecnologías de Telecomunicación

Análisis de la capacidad de detección e identificación de Malware basado en tráfico de red mediante IoC

Autor:

F. Javier Ros Raposo

Tutor:

F. Javier Muñoz Calle

Profesor Titular

Dpto. de Ingeniería Telemática

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2023

Trabajo Fin de Grado: Análisis de la capacidad de detección e identificación de Malware basado en tráfico de red mediante IoC

Autor: F. Javier Ros Raposo
Tutor: F. Javier Muñoz Calle

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Mis agradecimientos van dirigidos a todo mi entorno cercano y a mis profesores, por el gran apoyo y fuente de conocimiento que han sido para mi.

Con mención especial a mis padres y mi hermano, que nunca dudaron de mi.

Gracias a todos de corazón.

Francisco Javier Ros Raposo

Sevilla, 2023

Resumen

En la memoria se trata la capacidad de detección de malware que proporcionan los indicadores de compromiso. En primer lugar se aporta una base teórica necesaria, comenzando por teoría general sobre la ciberseguridad, amenazas informáticas, malware y estudio de malware, y se continúa revisando la teoría de inteligencia sobre amenazas y de indicadores de compromiso.

Tras revisar la teoría se proponen unas determinadas herramientas de software libre para realizar el estudio práctico en el sistema operativo linux, orientado a detectar muestras malware dirigidas dicho sistema.

En este estudio se ha procedido a utilizar indicadores de compromisos, tanto obtenidos de repositorios, como generados de manera manual para algunas muestras, utilizando para todo las herramientas Cuckoo Sandbox y Loki ioc-scanner. Además se ha utilizado la plataforma MISP para el intercambio de inteligencia de amenazas para gestionar los Indicadores de Compromiso y los análisis realizados con Cuckoo Sandbox.

Tras exponer todo lo realizado en el estudio, se comentan las conclusiones extraídas, tanto sobre el uso de los indicadores de compromiso, así como sobre las herramientas utilizadas, las cuales reflejan que la importancia de contar con indicadores de compromiso de calidad, y los adecuados, para realizar con éxtio la detección e identificación de malware. Además se hace notar la falta de herramientas de software libre para linux de análisis de malware, que generen indicadores de compromiso, y que los utilicen para detección de malware.

Al final de la memoria se encuentran unos apéndices que muestran como instalar y configurar las herramientas empleadas en el estudio.

Abstract

In this memory it is discussed the capability of the malware detection using indicators of compromise (IoC). Firstly, a theoretical basis is provided, composed by general theory of cybersecurity, computer threats, malware and malware study, and then it is profundiced at the threat intelligence and indicators of compromise theory.

After reviewing the theory, certaing open source tools for linux are proposed to carry out the practical study, which intention is to detect malware samples directed to the linux system using indicators of compromise. At the study have been used indicators of compromise that have been obtained from online repositories, and hasve been generated manualy for some specific malware samples. The tools used for the detection are Cuckoo Sandbox and Loki ioc-scanner. In addition, the MISP platform has been used managing the Indicators of Compromise and the analyzes carried out with the Cuckoo Sandbox.

After exposing everything done in the study, the conclusions drawn are discussed, both on the use of indicators of compromise, as well as on the tools used, which reflect the importance of having quality commitment indicators, and the appropriate ones, to successfully detect and identify malware. In addition, the lack of open source tools for linux that analyze malware to generate indicators of compromise and use them to detect it is noted.

At the end of the memory there are some appendices that show how to install and configure the tools used in the study.

Índice Abreviado

Glosario

amenaza informática una amenaza informática es una actividad, actor u elemento que tiene la capacidad de dañar la seguridad de un sistema y que supone un riesgo para el al poder generar un un ciberataque y comprometer su seguridad. 4

análisis forense Es el tipo de análisis que se aplica después de que se haya producido un incidente, no antes ni durante. Trata de averiguar que es lo que ha ocurrido, para ver como ha accedido la amenaza y que daños ha causado. 49

ciberataque un ciberataque es una acción que se produce contra un sistema informático y que atenta contra su seguridad. 4

ciberseguridad la ciberseguridad es el conjunto de elementos, medidas y equipos destinados a conservar la seguridad informática de sistemas y redes. 3

CTI Inteligencia sobre Amenazas (Cyber Threat Intelligence (CTI) en inglés) es el nombre que reciben los datos sobre amenazas de seguridad informática una vez se han recopilado, procesado y analizado en conjunto. 10, 11, 24

DIT Departamento de Ingeniería Telemática. 1

ETSI Escuela Técnica Superior de Ingeniería. 1

firma una firma digital (signature en inglés) es un elemento o patrón concreto que forma parte de del malware o amenaza de manera unívoca, actuando así como una huella digital y que se relaciona de manera directa con una amenaza informática. 16, 17, 23

heurística es el conjunto de técnicas que permiten resolver un problema. En el ámbito de la detección de malware, son el conjunto de técnicas que permiten descubrir malware en un sistema, sin que este sea conocido y se encuentre en la base de datos de la herramienta. 18

IDS un Sistema de Detección de Intrusos (Intrusion Detection System en inglés) es un sistema que detecta accesos no autorizados a una red o equipo, actuando de manera reactiva. 18

incidente un ciberataque que se ha llegado a realizar y que ha causado daños. 10, 23

Indicador de Compromiso un Indicador de Compromiso (Indicator of Compromise en inglés) es una evidencia digital de que se ha producido un ataque informático en el equipo, siendo un elemento o patrón que puede ser identificado en el sistema tras el ataque. Entre los artefactos que pueden ser identificados como un IoC se encuentran firmas de amenazas, direcciones IP, dominios y hashes de muestras malware. 11, 23, 26

intrusión una instrusión en el ámbito de la informática se refiere a actividades que comprometen la seguridad de un sistema o red. 19

IoA un Indicador de Ataque (Indicator of Attack en inglés) hace referencia unicamente a las técnicas utilizadas por las amenazas de seguridad informática para lograr su objetivo. 27

IPS un Sistema de Prevención de Intrusos (Intrusion Prevention System en inglés) es un sistema que protege a los sistemas informáticos de amenazas, actuando de manera preventiva. 19

malware software malicioso diseñado para acceder a un sistema informatico sin permiso y causar daños. 4, 6

muestra malware una muestra malware (sample en inglés) hace referencia al código o software del malware específico que se está tratando. 15, 25

proactivo que toma el control o actúa de manera activa y decide qué hacer en cada momento, anticipándose a los acontecimientos. 10, 19, 27

reactivo que toma el control o actúa como consecuencia de un acontecimiento que ya ha ocurrido. 10

servidor C2&C un servidor C2&C (Command and Control & Communications) es el servidor que da órdenes a malware y que recibe la información que estos recopilan. 24

TI Feed una Feed de Inteligencia (Threat Intelligence Feed en inglés) es un flujo de CTI sobre amenazas actuales y potenciales, provenientes de una organización u herramienta, ofreciendo datos que permiten actuar de manera proactiva ante amenazas informáticas. 13

TIP una Plataforma de Inteligencia de Amenazas (Threat Intelligence Platforms en inglés) se encarga de reunir y agregar CTI proveniente de diferentes organizaciones de seguridad, analistas o feeds de inteligencia. 12

TTP las Tácticas, Técnicas y Procedimientos (Tactics, Techniques, and Procedures en inglés) son los tres niveles en los que se puede analizar el comportamiento de un actor malicioso. La táctica describe al más alto nivel, mientras que las técnicas detallan en profundidad la táctica, y los procedimientos a su vez las técnicas. Las tácticas son la estrategia, las técnicas los métodos empleados en un ataque y los procedimientos los pasos del ataque. 11, 25

US Universidad de Sevilla. 1

vector de ataque un vector de ataque informático es un procedimiento que permite evadir la seguridad de sistema para acceder a el sin permiso. 6

vulnerabilidad de día cero vulnerabilidad que no se conoce y no ha sido reparada. 18

Índice de Figuras

Índice de Tablas

Índice de Códigos

Índice

1 Introducción

En este proyecto se va a analizar la capacidad de detección de malware mediante el uso de Indicadores de Compromiso (IoC) en el sistema operativo Linux, además de proponerse un entorno de trabajo, constituido por software libre, para realizar dichos análisis.

Para realizar los análisis será necesario realizar las siguientes tareas:

- Obtener las muestras malware que se van a analizar, para catalogarlas como malware.
- Conseguir los IoC que se van a utilizar en la detección.
- Preparar el entorno de trabajo, el cual estará formado por las herramientas:
  - Cuckoo Sandbox
  - Loki IoC scanner
  - Malware Information Sharing Platform (MISP)

Con la evolución de la tecnología y las comunicaciones, así como el aumento de la presencia de internet en nuestras vidas, cada vez son mayores las distintas amenazas informáticas que aparecen, y que suponen pérdidas económicas cada vez más significantes, haciendo en consecuencia que aumente la importancia y necesidad de la ciberseguridad.

El estudio realizado por la empresa Check Point Software Technologies revela que en el año 2021 el aumento de ciberataques hacia redes corporativas fue del 50 % respecto a los que hubo en el 2020 a nivel mundial [1].

La aparición de nuevos malware es una tendencia que lleva al alza durante años, y que no parece que vaya a cesar, por lo que cada vez es mayor objeto de estudio [2]. Además de aumentar la cantidad de malware, cada vez se presentan con un funcionamiento y técnicas más avanzadas, dando lugar a que las técnicas de detección de malware vayan quedando obsoletas. Estos eventos hacen que el estudio del malware tienda a un cambio de paradigma, al igual que lo hacen las tecnologías que emplean los desarrolladores del código malicioso. En Virustotal se pueden consultar las estadísticas de los últimos días, pudiendo ver el nivel de enlaces y archivos maliciosos que son detectados cada día [3].

En la figura Figura 1.1 perteneciente al instituto de investigación independiente en materia de seguridad informática de Alemania [4], se recoge la cantidad de malware dirigido al sistema operativo Windows que se ha registrado cada año, en la que se aprecia la tendencia alcista que se comentaba sobre las amenazas informáticas.

El objetivo del proyecto no es otro que mostrar la utilidad del uso de Indicadores de Compromiso en la detección de malware en el sistema operativo Linux. En adición el análisis se pretende que se desarrolle en un entorno de software libre. Para lograr el objetivo se llevarán a cabo las tareas mencionadas al comienzo de la introducción.

La memoria comienza con una parte introductoria con conocimientos que después permitirán desarrollar la parte práctica del final, en la que se desarrollará el objetivo. Los contenidos se estructuran de la siguiente manera:

En el capítulo 2 se tratará sobre el estado del arte y la teoría base de la Inteligencia de Amenazas e Indicadores de Compromiso, como son conceptos como la ciberseguridad, amenazas informáticas, malware y su análisis.

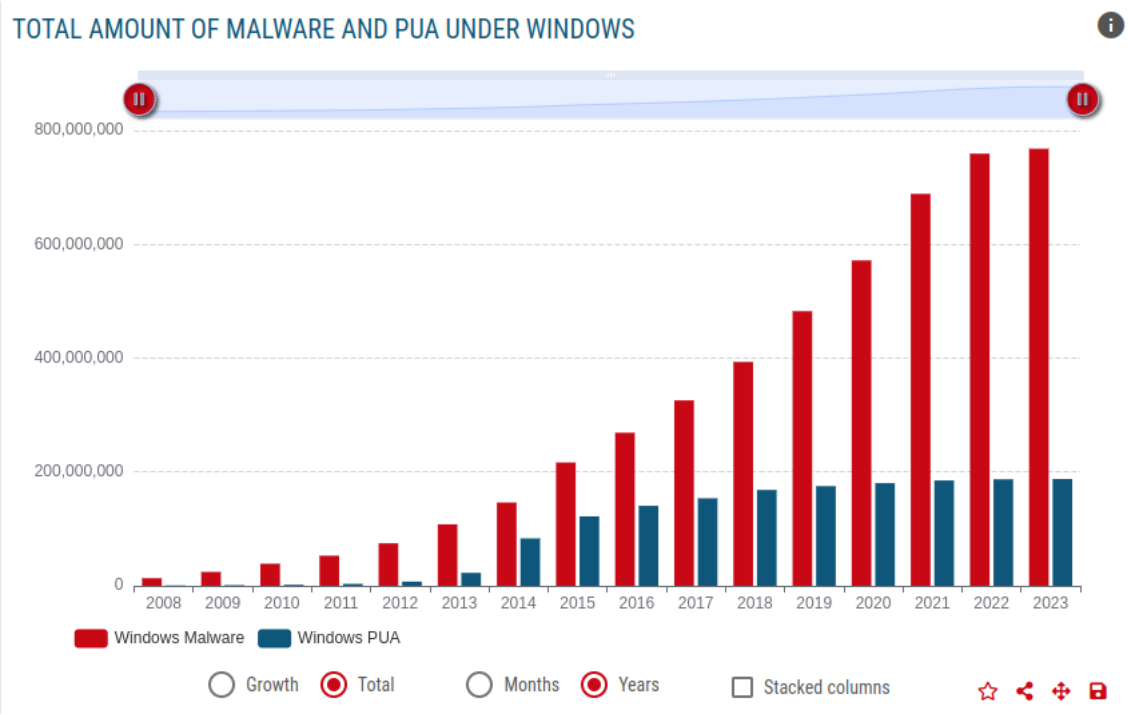

Figura 1.1 Cantidad de malware para windows registrado por AV-ATLAS.

Se van a introducir aspectos generales del malware que van a permitir comprender todo lo que los rodea, viéndose las técnicas de ataque y evasión de detección que emplean los desarrolladores de malware para diseñarlos, así como la taxonomía por las que se los puede clasificar en grupos. Además en la sección 2.7 se contempla la visión ofrecida en el artículo [5], en el que se habla sobre las distintas áreas del estudio de malware, comentando las técnicas y procedimientos para su análisis y detección.

En el capítulo 3 se trata la teoría sobre Indicadores de Compromiso, incluyendo además el concepto de Indicador de Ataque.

En el capítulo 4 se va a desarrollar como se han empleado los IoC para la detección de malware en el sistema operativo Linux. A lo largo del capítulo se comentará el entorno de trabajo propuesto, así como su uso, y la la obtención de muestras malware e IoC; tareas necesarias para llevar a cabo el uso de IoC en la detección de malware.

Finalmente en el capítulo 5 se hace una recapitulación sobre lo realizado en el proyecto.

Al final de la memoria se encuentran apéndices correspondientes a las herramientas empleadas, en los que se indica como se ha procedido para llevar a cabo su instalación. Tanto los Indicadores de Compromiso empleados, como los scripts de instalación, configuración y uso de las herramientas se pueden obtener en un repositorio creado para el proyecto [6].

2 Fundamentos: conceptos relacionados con los indicadores de compromiso

En este apartado se van a revisar conceptos de importancia sobre la ciberseguridad que son necesarios comprender para poder desarrollar el proyecto y alcanzar los objetivos. Se va a tratar que es la seguridad de un sistema informático, las amenazas a las que se enfrenta, y los ciberataques que reciben. Además se introducirá la Inteligencia sobre Amenazas(CTI), un concepto muy empleado en la ciberseguridad, y que dará paso a la teoría de Indicadores de Compromiso.

2.1 Ciberseguridad

La ciberseguridad es el conjunto de medidas y equipos destinados a conservar la seguridad de sistemas y redes. Dentro de la ciberseguridad se pueden realizar diferentes clasificaciones según el discriminante, como puede ser según el nivel en el que se trabaja (usuarios, aplicaciones, equipos, redes, planificación), o según el objetivo que tienen las acciones [7, 8].

Seguridad de red	Es el conjunto de técnicas que protegen una red de equipos de intrusiones y ataques.
Seguridad de información	Se protege la información, ya se encuentre en estado digital o analógico, y esté almacenado o en tránsito.
Seguridad de aplicación	Tiene el objetivo de que el software se mantenga limpio de amenazas, protegiendo así cualquier sitema y usuario que lo utilice. Existen diferentes etapas en las que se toman medidas para esto, empezando en un diseño adecuado de la aplicación y que existan el menor número de vulnerabilidades posible.
Seguridad operativa	Abarca procesos y decisiones que controlan y protegen la información y recursos de un sistema, tales como quien puede acceder y qué puede hacer.
Entrenamiento del usuario	Se enseñan a los usuarios las buenas prácticas de ciberseguridad medidas tienen el objetivo de evitar problemas de seguridad generados o que involucren a los usuarios de los sistemas enseñándoles las prácticas de ciberseguridad que deben seguir.

Tabla 2.1 Tipos de Ciberseguridad según el nivel de aplicación.

2.2 Seguridad

En el ámbito digital que se está tratando, los elementos que son objetivo de ataques, y que corren riesgos son la información y los sistemas informáticos, cuyo funcionamiento es vital para las organizaciones que hay detrás.

La seguridad, de manera genérica, es un estado en el que se está a salvo de peligros, encontrándose todos los riesgos controlados. La seguridad de la información, de un sistema informático o de una organización se basa en tres principios, la confidencialidad, integridad y autenticación [8]. Adicionalmente, también se puede identificar la disponibilidad.

- La confidencialidad sostiene que únicamente se puede acceder a la información y sistemas con el permiso adecuado.
- La integridad defiende que la información y sistemas no han sido manipulados por terceros ni se encuentran bajo errores.
- La autenticación se refiere a que la información proviene de una fuente fiable y conocida, así como que los sistemas son de confianza y verificables.
- La disponibilidad se define como la capacidad de sistemas o de información de encontrarse accesibles sin problemas cuando se requiere.

Son estas las cualidades que definen el perfecto funcionamiento de un sistema o estado de la información, y que peligran ante diferentes amenazas.

2.3 Amenazas Informáticas

Una amenaza informática es una actividad, actor u elemento que tiene la capacidad de dañar la seguridad de un sistema u organización, de manera directa o indirecta, y que en definitiva supone un riesgo para su seguridad; entendiéndose así que una amenaza es aquello que posibilita o que está detrás de la ejecución de un ciberataque. Las amenazas digitales se pueden agrupar según diferentes aspectos como su objetivo, origen, impacto o agente detrás de la amenaza.

2.4 Ciberataques

Un ciberataque son una o múltiples acciones que atentan contra un sistema, comprometiendo cualquiera de los pilares de su seguridad y alterando cualquier funcionamiento habitual. Generalmente los ciberataques se aprovechan de vulnerabilidades existentes en el sistema, pero se pueden agrupar según su funcionamiento.

Existen numerosos tipos de ciberataques, los cuales pueden compartir el mismo objetivo, causando un daño similar, aunque el camino sea diferente. Los principales tipos de ciberataques se pueden ver en la Tabla 2.2 [9].

2.5 Malware

El software se puede agrupar en subconjuntos según las intenciones que tienen u objetivos que persiguen [2], distinguiéndose los términos goodware, grayware y malware.

De manera general, el software se identifica con el término Goodware, tratándose de aplicaciones que no suponen ningún riesgo para el usuario ni el equipo, que únicamente llevan a cabo las acciones legitimas que se supone que deben realizar, siempre bajo el conocimiento del usuario.

Por otro lado están las aplicaciones y software malicioso, conocidos como malware, creadas para lograr un objetivo concreto que daña al usuario o equipo; no suelen llegar al usuario bajo su conocimiento, si no a través de diferentes técnicas y ataques.

Por ultimo, entre goodware y malware, se identifican aplicaciones de dudosa procedencia e intenciones, agrupadas como Grayware o PuP (Potencially Unwanted Applications). Este tipo de aplicaciones no son deseables, aunque no son malintencionadas, si no que funcionan de una manera que no es la ideal, llevando a cabo acciones sin el conocimiento del usuario, alterando el funcionamiento habitual del sistema en el que se encuentran, e incluso creando algún problema de seguridad, pero sin causar daños más allá de eso [10].

Una vez el equipo contenga el malware, el software hará aquello para lo que ha sido diseñado. Algunas de las intenciones que puede tener un malware son [11]:

Tabla 2.2 Principales tipos de ciberataques.

Nombre	Definición	Técnicas
Ataque de denegación de servicio (DoS o DDoS)	Tiene el objetivo de inhabilitar el sistema u organización al que va dirigido, impidiendo que pueda funcionar correctamente y deje de estar disponible.	• Volumétrico • De protocolo • A capa de aplicación
Man in the Middle (MITM)	Tiene el objetivo de interceptar la comunicación entre dos dispositivos sin tener autorización.	• Servidor DNS • Envenenamiento ARP • Servidor DHCP • Man in the Browser • Simulación de Punto de Acceso
Inyección SQL	Se inyecta código SQL en un servidor para conseguir información a la que no se tiene acceso.	• Ataque por unión • Ataque por error • Ataque Ciego
Ataque por ingeniería Social	Se manipulan a los usuarios para conseguir información confidencial.	• Phishing • Baiting • Tailgating
Ataque de contraseña	El objetivo es obtener la contraseña de acceso a un servicio o sistema. Hay diversos métodos para onseguirlo.	• Fuerza bruta • Ataque por diccionario
Ataque DNS	Está destinado a modificar el funcionamiento de un servidor DNS.	• Bloqueo DNS • Inundación • Dominio fantasma
Ataque mediante Malware	Se emplea software malicioso diseñado para llevar a cabo acciones en un sistema informático que causan daños y comprometen la seguridad del sistema.	• Gusano • Virus • Troyano • Ramsomware

- Interrumpir la actividad del equipo
- Robar información privada
- Conseguir acceso no autorizado al equipo
- Tomar el control del equipo
- Emplear el equipo como herramienta para llevar a cabo otros ataques
- Secuestrar el equipo así como la información que contiene, solicitando dinero por su liberación

2.5.1 Vectores de Ataque

Un vector de ataque es un método o procedimiento que permite conseguir acceso no autorizado a un equipo o una red en un ataque informático, aprovechando una vulnerabilidad existente para lograrlo. En el caso del malware, los vectores de ataque serán la vulnerabilidad empleada por los atacantes para que el malware acabe infectando un equipo.

Los vectores de ataque se pueden clasificar en pasivos y activos, entrando las técnicas empleadas por malware dentro de los activos. Los vectores de ataque pasivos son aquellos que no alteran ni afectan a los recursos del sistema ni a su funcionamiento, como phishing o cualquier método basado en ingeniería social. Por otro lado los activos son los que modifican el funcionamiento del sistema para conseguir su objetivo. Entre los vectores de ataque más comunes están:

- Fuerza bruta
- Phishing
- Correo electrónico malicioso
- Explotación de vulnerabilidades
- Unidades de memoria externa
- Main in the Middle

El propio malware puede ser un vector de ataque para otra amenaza digital o para que otro malware infecte el equipo.

2.5.2 Evasión de análisis

Con el tiempo el malware ha ido empleando diferentes técnicas para evadir los análisis, y según la forma de actuar para evitarlos se puede denominar al malware de una manera u otra [12, 13, 14].

Encriptación

Esta fue de las primeras técnicas de camuflaje de malware. El malware encriptado cuenta con un módulo que se encarga de encriptar y desencriptar el cuerpo principal del software malicioso, de forma que al ejecutarse el módulo de encriptación desencripta el cuerpo principal para que este se ejecute. El cuerpo principal hasta puede cambiar de clave según el equipo.

Con esta técnica el código malicioso se encuentra encriptado, y hace mucho más complicada una detección basada en firmas. Además se dificulta el análisis estático del código al no poder acceder al cuerpo principal. El punto débil que tiene es que el módulo de encriptación se mantiene intacto siempre, de modo que la detección por firmas es posible.

Oligomorfismo

Esta técnica es una evolución de la anterior. El malware oligomorfico al igual que el encriptado tiene un módulo de encriptación y un cuerpo con el código malicioso, pero el módulo puede cambiar entre una lista de desencriptadores, de forma que cada instancia del malware emplea uno de manera aleatoria, y así el módulo de encriptación no siempre es el mismo en los equipos infectados.

Presenta los mismos inconvenientes que el malware encriptado a la hora de ser estudiado, con la diferencia de que la existencia de varios desencriptadores hace que el tiempo aumente mucho.

Polimorfismo

El malware polimórfico supera la limitación presente en el encriptado y oligomorfico, además del módulo de encriptación y el cuerpo principal cuenta con un módulo de mutación que permite generar ilimitados módulos de encriptación diferentes y así en cada infección siempre es diferente.

El motor de mutación emplea numerosas técnicas de ofuscación para conseguir que el malware nunca parezca el mismo.

Esta técnica de desarrollo de malware hace que las instancias del malware que se van generando siempre sean diferentes, haciendo imposible que sean detectados por métodos basados en firmas, puesto que no hay

nunca una instancia con una firma que se haya podido recolectar previamente.

El problema que tiene esta técnica es que al final el malware acaba siendo desencriptado, y con una emulación del malware es posible ver el código malicioso.

Metamorfismo

Esta técnica deja de lado la encriptación y pasa directamente a mutar completamente el propio código malicioso. El malware metamórfico cuenta con un motor de mutación y con el cuerpo principal, de forma que en cada equipo infectado se ejecuta una instancia completamente diferente (tamaño, estructura, secuencia de código...) que hace lo mismo.

Este malware es mucho mas sofisticado que los anteriores, teoricamente nunca habrá dos instancias con firmas y cadenas comunes a causa de su capacidad de modificarse para generar diferentes instancias ilimitadas, y para detectarlos es necesario emplear técnicas basadas en heurística.

Técnicas de Ofuscación

Las técnicas de ofuscación son aquellas que hacen que un código sea más difícil de entender, dificultando su lectura y comprensión.

Son empleadas por los desarrolladores de malware para hacer que el código sea más difícil de analizar, además de por los desarrolladores de software para proteger la propiedad intelectual de los programas, impidiendo que puedan ser reproducidos [2].

Las técnicas mas comunes se agrupan en:

- Inserción de código muerto
- Reasignación de registros
- Reordenación de subrutinas
- Sustitución de instrucciones
- Integración de código
- Trasposición de código

Para profundizar en estas técnicas se pueden consultar [12, 13].

2.5.3 Taxonomía Malware

Tal y como se comenta en la introducción, la cantidad de malware que hay crece cada vez más [11], y es por eso por lo que la clasificación y correcta asignación de nombres es tan importante a la hora de hacer análisis y responder ante un incidente, para poder identificar correctamente el problema y actuar con la mayor rapidez posible [15, 2]. Además de hacer necesaria la taxonomía, la cantidad de malware existente la complica, pues son tantas ramas y variantes las que van apareciendo que en ciertas ocasiones diferentes organizaciones llegan a clasificar una misma muestra en diferentes categorías o a identificar de la misma manera a diferentes muestras.

Para profundizar más sobre la taxonomía de malware, y el esquema de nombrado de CARO se recomienda el artículo [16] y el libro [2] que se ha utilizado como base para la sección.

Categorizar el malware no es una tarea fácil, tal y como se comenta en [11, 15], pues existen numerosos aspectos que se pueden emplear para realizar la clasificación, muchos de los cuales pueden repetirse en diferentes muestras, haciendo que pertenezcan a distintas categorías, por lo que es necesario emplear más de una dimensión para poder diferenciarlos.

Tipología

El tipo de malware es una categoría general que los agrupa en función del medio que utiliza para conseguir acceso ilegal a un equipo, lo que hace en el equipo infectado y como funciona [2]. Entre los diferentes tipos que existen podemos distinguir los clásicos y más generales. A pesar de que otras organizaciones pueden considerar tipos más concretos.

- **Virus**: son código malicioso que cuando se ejecuta, intenta replicarse en otro ejecutable, y cuando lo

consigue el ejecutable queda infectado con el virus, permitiéndole continuar replicándose. Este tipo de malware se replican localmente, a través de archivos, o unidades de almacenamiento, pero no a través de la red. Además, la muestra del virus que observemos podrá clasificarse según el estado de replicación al que pertenezca [15]:

  - Germ: Es el código original, que precede a todas las réplicas.
  - Intended: Es el código que se ha intentado replicar sin conseguir funcionar correctamente.
  - Dormant: el código que se ha replicado, correctamente y está presente pero no infecta, como puede ser el caso de que sea específico para una arquitectura o SO y esté en otro en la que no tiene efecto.

- **Gusano**: es muy similar al virus, pero a diferencia de este, los gusanos únicamente se replican por red, y sin acoplarse a otro software como hace el virus [15], si no que son software independiente, lo que les permite replicarse sin necesidad de que el usuario intervenga [11].
- **Troyano**: es un tipo de malware que aparenta ser goodware, realizar las tareas que tiene que hacer, pero además realiza otras tareas maliciosas [15].
- **Ransomware**: malware que encripta los archivos de un equipo, y evita que el usuario pueda acceder al equipo. Solicita un rescate para devolver el acceso y los archivos [11].
- **Backdoor**: backdoor o puerta trasera es un tipo de malware que permite acceder al equipo esquivando las medidas de seguridad usuales. Pueden ser independientes o ir dentro de goodware [15].
- **RAT(Remote Administration Tool/Remote Access Trojan)**: es un tipo concreto de backdoor que permiten monitorizar el equipo infectado de manera remota [15].
- **Adware**: tipo de malware enfocado al marketing, que muestra anuncios en el dispositivo, redirige en el navegador a ciertas web, y además en algunas ocasiones pueden controlar el comportamiento del usuario para ofrecer anuncios específicos. Este tipo de malware no se autoreplica [17].
- **Spyware**: tipo de malware que se oculta en su dispositivo, controla su actividad y roba información. Entre la información que suele ser objetivo se encuentran contraseñas, direcciones de correo, cuentas bancarias, códigos de licencias,... A diferencia de otros tipos no se autoreplica.
- **Botnet**: Otro tipo de malware es el que convierte a los ordenadores que infecta, llamados **zombies** o bots, en parte de una Botnet, un grupo de ordenadores infectados que ejecutan los comandos que recibe de un servidor. Las botnets se convierten en una herramienta que los desarrolladores del malware utilizan para otras actividades, como ataques DDos o envío de spam [15, 11].
- **Dropper**: se dedican a descargar e instalar en el equipo otros malware [11].

Otros aspectos

Existen numerosos aspectos que pueden diferir entre los malware, y que se podrían utilizar para distinguirlos, algunos son atributos muy concretos.

- **Riesgo**: en función del riesgo que supone el software, mezclando greyware y malware, se pueden distinguir diferentes niveles:
  - Low: greyware.
  - Medium: software dudoso que pone en peligro la privacidad del usuario.
  - High: software que producen daños en el equipo y al usuario.
  - Severe: malware identificado.
- **Trazas**: el malware cuando se ejecuta en un equipo deja a su paso diferentes tipos de trazas en el sistema, de modo que se puede distinguir el malware en función de las trazas que genera; además, según el tipo de trazas que genere variará la capacidad de los antivirus de detectar esos malware a partir de las firmas que dejen, lo cual hace interesante tener presente este aspecto:
  - Trazas en disco: operaciones con archivos, cambios en los registros.
  - Trazas de red: escaneo de puertos, interacciones con C&C centers
  - Trazas de memoria: actividades de memoria, lectura/escritura, nuevos procesos,..
- **Objetivo**: se puede emplear el sistema operativo, el formato, o la plataforma a la que va dirigido el malware para distinguirlo de otros. Es algo que tiene mucho sentido, pues cada sistema tiene sus vulnerabilidades y modos de funcionamiento, de forma que en numerosas ocasiones el malware destinado a un sistema operativo no pueda funcionar en otros. En Microsoft distinguen:

  - Sistemas operativos
  - Scripts
  - Macros para Microsoft Office
  - Tipos de archivos

- **Fecha**: la fecha de creación es imposible de averiguar, pero la fecha en la que este se identifica y registra por primera vez si es posible tenerla.

2.5.4 Esquemas de Nombrado

Debido a la complejidad que supone la clasificación de malware, hasta el momento no existe un estándar que defina su clasificación y nombrado, lo que deriva en que cada organización clasifique y nombre el malware como crea oportuno, complicando la identificación de malware, y con las consecuencias que ello conlleva, como que las bases de datos de muestras malware sean un caos, y tengan muestras repetidas o falten otras. Desde 1990 han aparecido diferentes modelos, fruto de intentos de de la creación de un estándar para la identificación de malware.

Computer Antivirus Research Organization (CARO)

Computer Antivirus Research Organization (CARO) se formó en 1990, con el objetivo de facilitar el estudio e investigación en los análisis de malware. Fue la primera organización que trató de elaborar un convenio estándar para identificar malware, para lo que creó un esquema de asignación de nombres en 1991. Según [2] fue adoptado por algunas organizaciones, aunque a día de hoy Microsoft [18] y Trend Micro [19] son las únicas que continúan utilizando el convenio de nombres de CARO.

La idea tras CARO consiste en tener claro para cada muestra el tipo, plataforma, familia, variante e información adicional, aunque lo estrictamente necesario son la familia y la variante. Para conocer en profundidad los detalles sobre los tipos, familias, variantes y campos de información adicional existentes, así como las reglas para su uso y creación, puede verse el artículo [16].

El principal problema que presentaba este esquema de nombres es que pretendía incluir toda la información importante en el identificador, para lo que se necesitaban todos los atributos mencionados anteriormente, y debido a la cantidad de aspectos similares que pueden presentar diferentes muestras de malware, en ocasiones el identificador tendría que ser muy largo para poder diferenciar dos muestras muy similares. La solución a este problema es el uso de identificadores no basados en atributos del malware [20].

```
\vfill [<type>://][<platform>/]\textbf{<family>}[.<group>][.<length>]\textbf
    {.<variant>}[<modifiers>][!<comment>]
```

- El **tipo** de malware los agrupa de manera general como se ha visto en el apartado anterior, mientras que la **familia** es un grupo de malware del mismo tipo, que tienen en común un aspecto concreto, el cual puede ser:
  - Uso de una vulnerabilidad particular.
  - Comportamiento en cuanto a la memoria.
  - La carga en la que se basan.
  - El objetivo que tienen.
  - Las técnicas de ofuscación en la que se basa.

  Todos estos aspectos que caracterizan las familias están muy relacionados con el funcionamiento del malware, y ayudan a investigar un malware pudiendo observar otros de su familia [2].
- Además, en el esquema CARO se pueden distinguir **grupos** dentro de una familia, lo que permite organizar mejor familias muy amplias [16].
- Las **variantes** se emplean para referir a versiones de un mismo malware, que hacen lo mismo pero se presentan de manera diferente, generalmente con diferente encriptación, estructura y técnicas de ofuscación para evitar ser detectados.
- La **Plataforma** permite especificar la plataforma concreta en la que el malware trabaja.

Common Malware Enumeration (CME)

Common Malware Enumeration (CME) fue desarrollado en 2005 por United States Computer Emergency Readiness Team (US-CERT) en colaboración con MITRE [21].

La idea consistía en unir los enfoques utilizados por diferentes organizaciones a través de un único identificador que no estuviera basado en atributos, el cual permitiría conservar y unificar todos los estudios realizados.

Durante un tiempo se utilizó, pero al final, dado el aumento de apariciones de malware, y el cambio de paradigma que sufrieron, apareció la necesidad de tener claras las características del malware. Para suplir esa necesidad más adelante se creó un lenguaje de descripción de malware [20].

Malware Attribute Enumeration and Characterization (MAEC)

Malware Attribute Enumeration and Characterization (MAEC) es un lenguaje estructurado, desarrollado por la comunidad y mantenido por MITRE, que permite codificar y compartir información sobre las muestras de malware, formada por diferentes aspectos como el funcionamiento, las técnicas empleadas y la relación con otras muestras [22, 2].

Las técnicas de detección de malware de la actualidad normalmente caracterizan y determinan el funcionamiento de las muestras de malware, de modo que este lenguaje basado en ese tipo de atributos es perfecto para su identificación [22].

2.6 Inteligencia sobre Amenazas (CTI)

Cuando se produce un incidente de seguridad, en el sistema afectado se han tenido que realizar ciertas tareas para poder llevar a cabo el ataque, esas acciones, entre las que se encuentran por ejemplo el uso de vulnerabilidades, son las que caracterizan entre otras cosas a la amenaza que hay detrás del incidente.

La Inteligencia sobre Amenazas (Threat Intelligence(TI) o Cyber Threat Intelligence (CTI)) es el resultado de agrupar el conocimiento basado en evidencias sobre amenazas existentes o potenciales, incluyendo toda clase de detalles, como contexto, mecanismos implicados, indicadores,... [23, 24, 25, 26, 27].

Diferentes organizaciones pueden encontrarse ante una misma amenaza o ciberataque, de modo que tras producirse un incidente, este es estudiado y documentado, y se genera la información sobre amenazas correspondiente para ser distribuida y que pueda ser empleada por las herramientas de seguridad. Puede ser utilizada de manera proactiva, para prevenir incidentes, y de manera reactiva, para detectarlos, identificarlos y tratarlos correctamente. La CTI permite que se pueda llevar a cabo el intercambio de conocimientos de ciberseguridad de manera automatizada y eficiente, ayudando a conocer las amenazas informáticas a las que se puede enfrentar cualquier sistema digital, por lo que su uso está muy extendido en el mundo de la ciberseguridad.

Tal y como se comenta en [27], durante los últimos años ha estado aumentando considerablemente el número de amenazas y ciberataques, mientras que la investigación y estudios, concretamente por parte de la CTI y de los IoC ha sido muy escasa. Existen muchos problemas que necesitan ser solucionados. El hecho de ser tan utilizado es algo que le da y le quita fuerza al mismo tiempo, porque cuanto mayor es la cantidad de información que se intercambia, más efectivo es su uso, pero aparecen datos menos útiles y anticuados; además existen demasiados formatos y esquemas de nombrado, que hacen que las tareas de agrupación y procesamiento de la información sean más complejas, incluso llegando a situaciones en las que se pierde información por superposición o falta de profundidad de los estándares. Otro problema que afecta al intercambio de información sobre amenazas entre organizaciones es que se puede considerar una violación de la privacidad, lo cual hace que ciertas organizaciones no compartan ciertos datos, a pesar de ser una buena práctica muy beneficiosa para todos que permite aumentar el conocimiento general de las amenazas.

De manera general la CTI se encuentra descrita a partir de los elementos de la tabla Tabla 2.3.

Tabla 2.3 Atributos de la Inteligencia sobre Amenazas.

Identidad	Identifica el actor que hay detrás de la amenaza. Aunque no se conozca realmente el individuo u organización que hay detrás se relaciona con otros ataques que sean similares.
Motivación	Identifica la motivación que hay detrás de los objetivos que se persiguen.
Objetivos	Es aquello que se pretende conseguir con el ataque y es lo que produce el impacto que se genera.
Estrategia	Identificala metodología empleada para realizar el ataque y conseguir los objetivos.
TTPs	Las tácticas, técnicas y procedimientos (TTP) caracterizan el comportamiento del actor tras la amenaza para identificar que hacen y que harán. Algunos tipos de TTP son patrones de ataque específicos, malware y recursos empleados en el ciberataque.
Herramientas	Herramientas empleadas por los atacantes. El malware es un medio por el que se realizan los ciberataques de modo que queda dentro de las herramientas, pero en general no se identifica software malicioso, si no herramientas técnicas.
Indicadores de Compromiso(IoC)	Estos indicadores permiten describir comportamientos y detalles que permiten detectar TTPs, patrones de ataque, malware, herramientas, amenazas...
Indicadores atómicos	Son indicadores que tienen un ciclo de vida muy reducido. Pueden ser dominios, direcciones IP, hashes de archivos...
Víctimas	Es el objetivo al que va dirigido el ataque.
Respuesta	Medidas que se pueden tomar para responder o prevenir el ataque.

2.6.1 Ciclo de inteligencia

La CTI se produce en los pasos definidos en el conocido como ciclo de inteligencia. En la Figura Figura 2.1 se pueden apreciar todos los pasos del ciclo [28].

Figura 2.1 Ciclo de vida de la Inteligencia sobre Amenazas.

1. **Planificación**: Se identifican los objetivos y requisitos que debe cumplir la inteligencia que se quiere desarrollar, los cuales dependerán de la situación. Normalmente lo que se pretende averiguar con la

inteligencia es quienes son los actores tras los ataques, cual es su motivación, su modus operandi, y cuales son las medidas para prevenir sus ataques.

2. **Recolección de datos**: Se recolectan los datos (entre ellos IoC) a partir de los cuales se creará la inteligencia. La decisión de la etapa anterior condicionará los datos que se deben conseguir, y por tanto la fuente de la que provienen.
3. **Procesamiento de datos**: una vez se tienen los datos, estos se agrupan y de ellos se descartan los que no sean relevantes según lo establecido en la primera fase. Por último se expresan en formatos que permitan manejarlos.
4. **Análisis y producción de inteligencia**: Se analizan los datos ya procesados de la etapa anterior para crear la CTI, añadiéndoles detalles contexto y otros datos. La inteligencia será la que se utilice a la hora de tomar decisiones y planificar las medidas para conservar la seguridad, permitiendo responder a las preguntas planteadas en la primera fase.
5. **Difusión e integración de la inteligencia**: La CTI se integra en las diferentes herramientas de seguridad y se distribuye e intercambia en las Plataformas de Inteligencia de Amenazas (TIP). Se generan informes con los resultados, que son repartidos en el equipo de seguridad.
6. **Evaluación y feedback**: Esta fase retroalimenta al resto, en ella se analizan los resultados y la efectividad de la inteligencia con el objetivo de mejorar y crear la inteligencia que mejor se ajuste a la situación.

2.6.2 Tipos

Tal y como se explica en el artículo [27] es conveniente distinguir subtipos dentro de la CTI, para poder trabajar agilmente con ella, pues se abarcan muchas cosas y no siempre se busca lo mismo. Se pueden distinguir diferentes áreas siguiendo distintas filosofías de división, pero la forma de agrupamiento más corriente deriva en los siguientes grupos:

- **Información de Amenaza Estratégica**: es información de alto nivel que emplean los estrategas para tomar decisiones y entender y analizar riesgos y amenazas. Suele presentarse como informes que tratan sobre el contexto de los posibles ataques, cual es la motivación que tienen, que buscan, las tendencias de metodología, perfiles de las amenazas.... Responde a quien hay detrás del ataque y porqué.
- **Información de Amenaza Operacional**: trata información sobre posibles ataques que se pueden producir en un tiempo próximo o se están produciendo.
- **Información de Amenaza Táctica**: son las TTPs tras una amenaza; pretende dar a conocer la formas de actuar y realizar ataques por parte de los actores tras la amenaza. Esta categoría de información de amenazas es empleada por los equipos de respuesta para prepararse correctamente frente a las amenazas y localizar posibles brechas de seguridad. Responde a cómo y dónde se ha producido el ataque.
- **Información de Amenaza Técnica (Technical Threat Intelligence, TTI)**: la información de esta categoría se produce mayoritariamente a través de los IoC recolectados en un equipo en el que se ha producido un incidente, permitiendo así conocer que es lo que ha ocurrido. Es consumida por recursos técnicos de una organización como equipos de respuesta o herramientas de análisis.

  Es la categoría más fácil de cuantificar dada la naturaleza de los datos que procesa, lo que deriva en que sea fácil de estandarizar, y rápida y sencilla de utilizar por las herramientas de seguridad. Por estas razones es el tipo de información sobre amenazas que más se distribuye y más se emplea. Sin embargo, a pesar de su popularidad, la TTI tiene ciertas desventajas, pues el ciclo de vida que tienen es en ocasiones limitado dada la naturaleza de los IoC, y los IoC inservibles pueden eclipsar a los útiles haciendo menos efectiva la información.

2.6.3 Plataformas de Inteligencia de Amenazas (TIP)

Las Plataformas de Inteligencia sobre Amenazas (TIP) tienen el objetivo de administrar la información sobre amenazas informáticas. En estas plataformas se llevan a cabo los procesos ontológicos de los datos

(tareas de agrupamiento y enriquecimiento de datos provenientes de distintas fuentes de información), el almacenamiento y el intercambio de CTI, especialmente de la categoría TTI (los IoC) [29, 30].

Suelen soportar un gran número de estándares, formatos y esquemas para dar servicio al mayor número de organizaciones y herramientas de seguridad. Las técnicas ontológicas normalmente emplean machine learning o big data para realizar la reunión de los datos.

Existen numerosas TIPs, algunas de las que son software libre son:

- Malware Information Sharing Platform (MISP)
- Collaborative Research into Threats (CRITs)
- Collective Intelligence Framework (CIF)
- GOSINT
- Model-based Analysis of Threat Intelligence Sources (MANTIS)
- Repositorio común y estructurado de amenazas y código dañino (REYES)
- MineMeld
- Yeti
- Open CTI

Las TIPs son un concepto diferente a las feeds de inteligencia, que son flujos constantes de datos sobre amenazas, como indicadores o CTI, pero sin realizar ningún tipo de agrupamiento.

Open CTI

Open CTI fue creada por la Agencia Nacional Francesa de Ciberseguridad (ANSII), junto al Equipo de Respuesta a Emergencias Informáticas de la Unión Europea (CERT-EU). La plataforma se basa en el estándar Structured Threat Information Expression (STIX). Además permite integrarse con herramientas como MISP y MITRE ATT&CK [31].

Yeti

Yeti es una plataforma que se creo para organizar CTI, TTPs, IoCs y observables en un mismo repositorio. Además de unificarlos, los enriquece a través de diferentes fuentes. Cuenta con una API para que pueda ser utilizada por otra herramientas [32].

2.6.4 Estándares

Entre los estándares más relevantes se encuentran los siguientes:

- RID
- CIF (Collective Intelligence Framework)
- IODEF (Incident Object Description Exchange Format)
- STIX (Structured Threat Information eXpression)
- TAXII (Trusted Automated eXchange of Indicator Information)
- OpenIOC (Open Indicator of Compromise)
- CyBox (Cyber Observable Expression)
- VERIS (Vocabulary for Event Recording and Incident Sharing / Verizon)
- CAPEC (Common Attack Pattern Enumeration and Classification)
- MAEC (Malware Attribution and Enumeration Characterization)
- ATT&CK (Adversarial Tactics, Techniques & Common Knowledge)

Para entrar en más detalles sobre los estándares de CTI se puede ver [27]. De manera adicional a estos estándares, se pueden encontrar algunos formatos en los que se puede recoger la CTI, los cuales son más "prácticos" al formar parte de herramientas de detección e identificación de malware, como son las reglas

YARA y SNORT.

Structured Threat Information Expression (STIX)

Structured Threat Information Expression (STIX) es un estándar abierto (un lenguaje) para definir CTI con el objetivo de que esta pueda ser administrada y compartida de manera eficiente. El estándar es mantenido por la organización OASIS Cyber Threat Intelligence, que además es responsable de:

- Trusted Automated Exchange of Indicator Information (TAXII): un protocolo de aplicación diseñado para el intercambio de CTI sobre HTTPS [33].
- Cyber Observable Expression (CybOX): es un estándar destinado a la caracterización y comunicación de eventos relacionados con la seguridad en un sistema de información, que son observables en todas las operaciones de los sistemas y de las redes de comunicaciones. Este esquema de definiciones ha sido integrado en versiones más modernas de STIX [34, 35].

STIX es un estándar modular muy flexible, con estructuras basadas en XML, que a pesar de su complejidad es el más empleado en la actualidad. Puede incluir extensiones CybOX, IODEF, OpenIOC, reglas YARA, reglas SNORT, y esquemas de nombrado XML, como CAPEC, MAEC, ATT&CK,... Emplea el protocolo de transporte TAXII, y es el sucesor de IODEF, el cual empleaba como protocolo de transporte RID.

Está organizado de forma que se distinguen 8 elementos: Campaigns, Indicators, Observables, TTP (Tactics, Techniques and Procedures), Incidents, ThreatActors, ExploitTargets and Courses of Action.

Yara

Yara es un lenguaje creado por VirusTotal que permite detectar, identificar y clasificar malware entre otras cosas. Permite describir malware en lo que se denominan reglas Yara, permitiendo cada una identificar el malware que se describe en ella. Las reglas contienen expresiones lógicas que determinan si se ha identificado el malware o no según se hayan localizado en el supuesto archivo malware los rasgos que lo describen.

La extensión de los archivos en los que se escriben las reglas Yara es ".yar". En las reglas se distinguen un nombre, etiquetas que permiten identificar su contenido y filtrarlas, y 3 secciones: "meta", "strings" y "condition". Lo único obligatorio son el nombre y la sección "condition" [36].

- El nombre distingue mayúsculas y minúsculas, y debe empezar por un carácter.
- La sección "meta" permite añadir metadatos sobre la regla, pero no se pueden utilizar en la condición.
- En la sección "strings" se definen las cadenas que puede utilizar la sección de condición para identificar el malware.
  - Los identificadores de esas cadenas deben empezar por '$'.
  - El valor de la cadena puede estar en ASCII "texto" o en hexadecimal 4D 5A.
  - Además el valor de las cadenas hexadecimales permite el uso de operadores para que se pueda adaptar a diferentes situaciones, diferente longitud, indiferencia sobre el valor de ciertos bytes, diferentes valores en ciertos bytes...
  - En el caso de las cadenas ASCII se puede indicar si se distingue entre mayúsculas o minúsculas.
- En la sección "condition" se expresa la condición que se debe cumplir para que se considere que la regla ha dado positivo y que ha identificado y detectado al malware al que hace referencia.

Además de la sintaxis básica de yara, existen módulos oficiales y personalizados, que amplían las capacidades de las reglas y el alcance de análisis que tienen, siempre haciendo más fina la detección, como por ejemplo:

- PE
- ELF
- Cuckoo
- HASH

El módulo HASH permite utilizar la regla YARA como una firma de un archivo, a través de su HASH.

2.7 Estudio del Malware

En el estudio de malware se pueden distinguir dos procesos diferentes, el análisis y la detección [5]. Por un lado la detección de malware pretende conocer si hay malware en un equipo, mientras que el análisis busca estudiar la muestra malware de la que se disponga para conocer con detalle su funcionamiento. Ambos procesos se realimentan mutuamente, de forma que el análisis brinda información para realizar una detección más eficiente y la detección de malware ofrece nuevas muestras malware para analizar, sin embargo el flujo habitual es realizar primero análisis para después poder detectar [13].

El análisis de malware es el proceso en el que se estudia una muestra malware para conocer como funciona y como se comporta, permitiendo así, a través de los resultados [11]:

- Conocer el objetivo que persigue el ataque y poder ver el impacto que ha causado, y así poder mitigar sus efectos y evolución en la medida de lo posible.
- Conocer las técnicas y vulnerabilidades empleadas para saber que fallos en la seguridad le han permitido infectar el equipo, y poder así mejorar la seguridad del equipo y de la red.
- Extraer CTI para alimentar al proceso de detección de malware y desarrollar mejores herramientas de seguridad.

Por otro lado, la detección de malware es el proceso mediante el cual se estudia si hay presente malware en un sistema; y en este proceso se distinguen 3 etapas [37].

- En primer lugar es necesario contar con CTI para alimentar la herramienta de detección y que pueda funcionar de manera eficiente, por lo que la primera etapa consiste en obtener la CTI, la cual es fruto de realizar análisis de malware a muestras que se han ido recolectando en incidentes de seguridad.
- Una vez se tiene la información que alimenta la herramienta, el siguiente paso para llevar a cabo la detección es la recolección de rasgos o artefactos del equipo en el que se busca la presencia de malware, para posteriormente contrastar esos datos para verificar la existencia de malware en el equipo.
- Por último, en caso de haber localizado malware, se procede a identificarlo para poder actuar de manera pertinente.

2.7.1 Técnicas de Análisis de Malware

Las técnicas de análisis de malware se pueden clasificar según su metodología en estáticas, dinámicas o de comportamiento, e híbridas. También se pueden agrupar de forma mas específica, según si se centran en el análisis del código, o por el contrario se centran en analizar la memoria del equipo.

El malware por norma general emplea diversas técnicas de evasión de análisis, por lo que se suelen emplear técnicas de distintos tipos y metodologías para analizar las muestras de malware y poder conseguir un mayor nivel de detalle; lo habitual es realizar técnicas de análisis estático y dinámico, dando lugar a lo que se conoce como un análisis híbrido.

Estático

Las técnicas de análisis estático se caracterizan por analizar las muestras malware sin ejecutarlas; emplean ingeniería inversa para entender que es lo que hace el código y poder determinar si es malicioso o no. Muchas de esas muestras son ejecutables portátiles (Portable Ejecutables, PE) que normalmente van empaquetados en binario por lo que será necesario emplear herramientas para desempaquetarlos y poder analizarlos.

Las herramientas que pueden ser utilizadas en estos análisis son depuradores, desensambladores, descompiladores y analizadores de código fuente. Y algunas de las técnicas que se pueden ver [11]:

- Identificar el formato del archivo, y el sistema operativo y arquitectura para la que está diseñado el malware.
- Creación del hash que identifica de manera unívoca esa instancia del malware. Para obtener el hash se emplean algoritmos criptográficos.
- Escaneo de la supuesta muestra son herramientas anti virus para averiguar si existen firmas de la muestra malware.

- Extracción de cadenas, funciones y metadatos asociados con el archivo del malware, para conseguir identificadores (firmas) asociados con la muestra y detalles sobre el funcionamiento.
- Identificación de técnicas de ofuscación empleadas para evadir el análisis, como empaquetar o encriptar el código malicioso.
- Clasificación de la muestra para obtener información general de la familia y tipos a los que pertenezca.

De estos análisis se extraen metadatos asociados al malware así como patrones de detección, en general información que permite decidir hacia donde enfocar el resto de análisis que se vayan a hacer. Ejemplo de la información que se puede encontrar en un análisis de este tipo son:

- Llamadas a APIs de Windows
- Firmas de cadenas
- Graficos del Flujo de Ejecución (control flow graph, CFG)
- Frecuencia del uso de Opcode (códigos de operación)
- Secuencias de bytes n-grams
- Tamaños de archivos
- Longitudes de funciones
- Aspectos de Red, como puertos, direcciones IP y peticiones HTTP

La metodología de estos análisis es rápida y segura, además de presentar pocos falsos positivos, pero el problema que presenta es que puede ser evitada con técnicas de evasión de análisis. La encriptación evita el acceso al código, y la ofuscación, que se centra en modificar el código del malware evita que las firmas puedan ser de utilidad para detectarlo en el futuro, pues el malware mutará y no será exactamente igual aunque el comportamiento continúe siendo el mismo.

Dinámico o de comportamiento

Las técnicas de análisis dinámico o de comportamiento se centran en ejecutar las muestras de malware en un entorno controlado para monitorizar su comportamiento y conocer en detalle como funcionan.

El entorno de ejecución no puede ser cualquiera ya que el malware puede estar preparado para no actuar en caso de detectar que está siendo ejecutado en un entorno seguro para ser estudiado, por ello el entorno en el que se vaya a realizar el estudio debe estar preparado para ser invisible para el malware. Existen diferentes tipos de entornos controlados:

- Emulador: es un entorno controlado que se utiliza para controlar la ejecución el malware.
- Depurador: es un programa que observa y recopila información sobre la ejecución de otro programa.
- Simulador: simulan la ejecución de las acciones que llevaría a cabo el malware, pero sin que realmente se ejecuten.
- Máquina virtual: es un programa que corre un sistema operativo aislado del sistema operativo del equipo en el que se instala.

Cuando el malware se ejecuta este interactúa con el sistema y lleva a cabo diversas tareas para cumplir su objetivo; son esas interacciones las que se pretenden monitorizar con este tipo de análisis, y así poder explicar el funcionamiento del malware. Así lo que se hace una vez está preparado el entorno de ejecución con todas las herramientas necesarias, es ejecutar el malware y monitorizarlo, para después analizar los datos recolectados y sacar los resultados [11]. Algunos de los procesos de monitorización que se cursan son [11]:

- Monitorización de procesos
- Monitorización de la actividad del sistema de ficheros
- Monitorización del registro de Windows, observando las entradas accedidas y modificadas por el malware.
- Monitorización de la red, concretamente del tráfico entrante y saliente del equipo infectado.

Estas técnicas tienen algunas desventajas, como su incapacidad para analizar malware de kernel, y que en comparación con las técnicas estáticas requieren de más tiempo y recursos, y conllevan un mayor riesgo al precisar la ejecución del malware. El mayor problema que presentan que al trabajar observando la ejecución del malware, si este no llega a ejecutarse o lo hace de manera parcial, no todas las funciones del malware se detectarán, y el malware no será completamente analizado.

Las técnicas de evasión de análisis aprovechan esta limitación para evitar los análisis dinámicos, por ejemplo como la ejecución del malware únicamente en ciertas condiciones. Estas condiciones pueden tener detrás razón directa, como que el entorno de ejecución no sea controlado, que no haya programas de análisis de malware, que haya ciertos programas de los que se busca información; o simplemente para evitar que el malware se ejecute siempre y lo haga solo de manera aleatoria, para complicar así el análisis dinámico, puesto que si el malware no se ejecuta cuando se realiza el análisis pues no se podrá analizar.

Híbrido

Este tipo de análisis mezclan técnicas estáticas y dinámicas, obteniéndose así lo mejor de ambos para reducir el tiempo del análisis y así disminuir el número de falsos negativos. Normalmente se comienza realizando análisis estático, y a través de los resultados se redirige el análisis dinámico [37]. El beneficio de combinar análisis estático y dinámico es que se obtienen las ventajas de los dos, logrando una mayor eficiencia de detección y realizando un estudio más detallado.

Memoria

Este conjunto de métodos consisten en analizar la memoria RAM del equipo afectado para localizar actividades maliciosas, así como para ver la forma de actuar que tiene el malware. Estas técnicas son cada vez más populares, dada la precisión y el nivel de detalle que ofrecen sus resultados. En primer lugar es necesario guardar una imagen de la memoria RAM del equipo para después localizar actividades maliciosas y ver la forma de actuar que tiene el malware.

Entre las técnicas que tienen esta forma de trabajar se encuentran:

- API Hooking
- DLL Injection
- Hidden processes

Código

Este conjunto de técnicas se centran en analizar el código para entender el funcionamiento interno del malware. Siguiendo la clasificación general se distinguen:

- Análisis de Código Estático: Estos se centran en desarmar el binario y analizar detalladamente el código.
- Análisis de Código Dinámico: Estas técnicas consisten en depurar el código de las muestras.

2.7.2 Métodos de Detección de Malware

Existen diversas técnicas para detectar malware, y principalmente se pueden agrupar según en lo que se basan para detectar, destacando los métodos basados en firmas, y los basados en heurística. El enfoque más tradicional es el basado en firmas, pero a causa de las limitaciones que presenta y la evolución que ha sufrido el malware a lo largo de los años, se han comenzado a utilizar nuevos enfoques: técnicas de comportamiento, heurísticas , de comprobación de modelos, y las más novedosos, basadas en deep learning [37].

Ninguna técnica de detección es perfecta, no hay ningún algoritmo o herramienta capaz de detectar todo el malware existente. Las causas de que esto sea así son las limitaciones que presentan las técnicas de detección y las técnicas anti-análisis empleadas por el malware que dificultan la detección. Por estas razones la detección y análisis de malware se suelen abordar desde diferentes enfoques para obtener el mayor éxito posible [37].

Técnicas basadas en Firmas

Una firma es un rasgo/artefacto que permite identificar una amenaza digital de forma unívoca, como una huella digital. En el malware son cadenas de código o hashes de muestras, en general información y patrones. Las firmas que se tienen registradas se han recolectado en un análisis que se ha realizado a la muestra malware a la que identifican, de modo que esa muestra malware en concreto se considera conocida, pues ya ha aparecido y se ha podido estudiar.

Las técnicas de detección de malware basadas en firmas son muy utilizadas por los antivirus, son muy rápidas

y eficaces para detectar muestras malware conocidas, pues al haber sido previamente analizadas y recogidas las firmas que la identifican, únicamente es necesario tratar de localizar las mismas firmas para detectar su presencia, consiguiendo un buen ratio de detección sobre estos. Sin embargo, por como funciona esta forma de detección, trabaja de manera ineficiente contra malware cuyas firmas no se conocen, ya se deba esto a que se trate de un malware hasta hora nunca analizado, se emplee una vulnerabilidad de día cero, que aún no se ha descubierto, o que emplee técnicas de evasión de análisis, haciendo que sus firmas no sean siempre las mismas. Esta es la principal limitación que tiene este tipo de análisis, pues cualquier malware que emplee técnicas de evasión de análisis puede cambiar, cambiando con el sus firmas e impidiendo que se detecten.

Estas técnicas también pueden ser conocidas como detección basada en cadenas o patrones.

Técnicas basadas en Heurística

La heurística aplicada a la detección de malware son el conjunto de técnicas que permiten detectar la presencia de malware desconocido (no se tienen las firmas que lo identifican) en un equipo. Estas técnicas también se conocen por basarse en el análisis de comportamiento, pues se dedican a monitorizar un equipo para detectar malware [37].

Este tipo de procesos de detección se desarrollan en dos fases. Primero se lleva a cabo el entrenamiento o aprendizaje, en el que se analizan las actividades que se llevan a cabo en el equipo de manera habitual, y después en la etapa de monitorización, mientras el equipo está en funcionamiento, se van analizando los programas y acciones y se determina sin son benignas o maliciosas, según se encuentren patrones y formas de actuar catalogadas como maliciosas o que difieran de lo observado en la primera fase.

Estas técnicas son capaces de detectar malware cuyas firmas no se poseen, pues ya sea desconocido o la forma del malware haya cambiado es el comportamiento lo que hace al programa maligno, y eso es lo que se detecta con estos métodos [37]. Sin embargo, a pesar de que el ratio de detección es bastante mejor que el de las técnicas basadas en firmas, tienen algunos inconvenientes, no solo requieren de muchos recursos y tiempo para escanear y monitorizar los equipos, si no que presentan un alto ratio de falsos positivos.

Entre los procedimientos disponibles para obtener la información para llevar a cabo este tipo de detección [37]:

- Análisis automático en sandbox
- Monitorización de llamadas de sistema
- Monitorización de cambios en archivos
- Comparación de snapshots de registros
- Monitorización de actividad en la red
- Monitorización de procesos

Técnicas basadas en especificación

Estas técnicas se monitorizan los programas, comprobando que se están comportando como se supone que deben hacerlo. Derivan de las técnicas basadas en heurística, con la diferencia de que no emplean machine learning e inteligencia artificial para detectar comportamientos anómalos, si no que directamente comparan el funcionamiento con lo que se supone que tiene que hacer el programa.

Por un lado suavizan la cantidad de falsos positivos, pero por otro aumentan los falsos negativos.

2.7.3 Implementación de las técnicas

Sistemas de Detección de Intrusos

Los Sistemas de Detección de Intrusos (IDS) tienen el objetivo de detectar actividades intrusivas, amenazas contra la seguridad de un sistema o red; y pueden encontrarse implementados en un equipo físico dedicado o en software. Se encargan de ejecutar de manera automática el proceso de detección de amenazas informáticas, para lo que trabajan monitorizando y registrando eventos, y en caso de detectar un ataque producen los avisos y alertas oportunos [38]. Los IDS se pueden clasificar en función del nivel al que actúan, la metodología que emplean y la forma en la que actúan contra las amenazas [39].

En función de los eventos que se controlan se tendrá información de diferentes fuentes, pudiendo distinguir entre IDS basados en Red (NIDS), que monitorizan el tráfico de red, y los IDS basados en Host (HIDS), que monitorizan la actividad interna que tiene lugar en el equipo, observando procesos y el sistema de ficheros.

Según la metodología empleada para detectar la actividad intrusa, sin importar los eventos que se monitoricen, se pueden distinguir los IDS basados en la detección de anomalías (AIDS), y los basados en la detección de firmas (SIDS). Los SIDS emplean técnicas de detección basadas en firmas, de forma que se monitorizan los eventos y se comparan sus firmas (patrones y cadenas de texto o binarias) con las que se encuentran en la base de datos. Esta metodología precisa conocer firmas del ataque antes de que se lleve a cabo, y como todas las técnicas basadas en firmas, la principal desventaja que tiene es que no puede detectar ataques desconocidos (cuyas firmas no se encuentren en la base de datos). A pesar de la limitación que tiene esta metodología es la más implementada en los IDS. Los AIDS trabajan con perfiles realizados con estadísticas de comportamiento que se obtienen observando los eventos a lo largo del tiempo. Primero se establece lo que se conoce como comportamiento habitual, y después se monitoriza el comportamiento y se compara con el clasificado como normal para detectar anomalías. La ventaja que presenta la metodología de los AIDS es que permiten detectar cualquier ataque que se salga de las actividades habituales, incluso ataques desconocidos hasta el momento. Sin embargo, estas técnicas presentan muchos falsos positivos y negativos, al darse eventos lícitos que se pueden clasificar como intrusos y viceversa.

Por último, en función de la respuesta que dan ante la detección de un ataque, se distinguen los IDS con respuesta pasiva, que se dedican a dar alertas y registrarlos sin actuar, y los IDS con respuesta activa, que son conocidos como Sistemas de Prevención de Intrusos.

Sistemas de Prevención de Intrusos

Los Sistemas de Prevención de Intrusos (IPS) llevan a cabo el proceso de detección al igual que los IDS y en caso de detectarse un ataque proceden a bloquearlo con los mecanismos de respuesta pertinentes. Además llevan a cabo acciones de manera proactiva para que los ataques no se puedan llegar a producir. A diferencia de los IDS, el principal objetivo de los IPS es evitar que el ataque se produzca, y en caso de que este ocurra evitar el máximo de daños posibles [40].

Programas

Algunos NIDS basados en la correlación de información son Snort, Suricata y Bro IDS. Un ejemplo de HIDS es Tripwire.

Para llevar a cabo cada una de las técnicas de detección y análisis de malware será necesario emplear herramientas específicas que permitan desempeñar las tareas pertinentes. A continuación se muestra una completa recopilación realizada en [41]:

Herramientas para análisis dinámico:

- Herramientas de red:
  - Wireshark
  - Microsoft Network Monitor
  - Netcat
  - BurpSuite
  - Fiddler
  - DNS Query Sniffer
  - FakeNet-NG
  - INetSim
- Debugueadores:
  - X64dbg
  - Immunity Debugger
  - WinDbg
- Sandboxing:
  - Virus Total

  - Hybrid Analysis
  - Cuckoo
  - Any.run
  - Intezer
  - Joe Sandbox
  - VB Analysis Tools
  - ViperMonkey
  - decode-vbe.py
  - oledump.py
  - CWSandbox
  - NormanSandbox
  - Oracle Virtual Machine
  - VirtualBox
- Ingeniería inversa:
  - IDA Pro
  - Ghidra
  - dotPeek
  - Scylla
  - PdbXtract
- Análisis de cadenas:
  - FLOSS
  - Sysinternals Strings
  - Fireeye stringsifter
- Herramientas para analizar el comportamiento:
  - Process Explorer
  - Process Monitor
  - Process Hacker
  - CaptureBAT
  - Sysmon
  - API Monitor
  - CMD Watcher
  - Autoruns
  - Regshot
  - Flypaper (Password : “rich”)
  - Microsoft ASA (Attack Surface Analyzer)
- Anubis
- Ollydbg
- OllydumpEx
- ImpRec

Herramientas para análisis estático:

- Herramientas para analizar ejecutables portables (PE):
  - PE-bear
  - pev the PE file analysis toolkit
  - PeStudio
  - PEiD
  - Resource Hacker
  - CFF Explorer
  - Exeinfo PE
  - Dependency Walker
- HxD

Máquinas virtuales preparadas para realizar análisis:

- REMnux
- OALabs Malware Analysis VM
- FLARE VM
- Kali Linux

Otras herramientas para análisis:

- Didier Stevens Suite
- Fireeye Market
- ProcDOT
- Malzilla
- Kahu Security Tools
- HashMyFiles
- CyberChe

3 Indicadores de Compromiso

Los incidentes de seguridad que se producen en los sistemas, de manera habitual producen modificaciones en el sistema, las cuales dependerán de la forma de actuar de la amenaza. En el caso del malware, este produce modificaciones en el sistema para aumentar sus privilegios y tratar de pasar desapercibido ante herramientas de detección, sin embargo, estas modificaciones no siempre son iguales, pues dependerán de como se haya desarrollado el código malicioso en cuestión, lo que se introduce en el punto 2.5 de la memoria.

Los equipos de seguridad, además de estudiar la naturaleza de los ataques y amenazas, indagan en estas modificaciones que realizan las amenazas en los sistemas que atacan, en lo que se conoce como informática forense. Así, se analizan esas modificaciones en equipos que hayan sufrido un incidente de seguridad (un ciberataque que se ha llegado a producir), produciendo lo que se conocen como Indicadores de Compromiso (IoC), los cuales interpretan esas modificaciones como indicadores de que el sistema ha sido comprometido, o se están realizando acciones previas a ser comprometido [24, 42, 27].

Los IoC pueden ser artefactos, patrones o comportamientos, cuya presencia se traduce en una cierta probabilidad o certeza de que se esté produciendo o haya sucedido un ataque, aunque puede darse un falso positivo.

Están muy relacionados con la TTI, pero no se deben confundir con inteligencia, pues no son datos procesados, sino simplemente datos, que normalmente contribuyen en la creación de inteligencia y suelen ser incluidos en ella. También se debe entender de manera adecuada su relación con las firmas, pues estas son la huella digital de los archivos empleados en el ataque, como puede ser el hash de un código malicoso, y las firmas pueden ser tratadas como IoC, de esta forma una firma siempre dará una coincidencia 100 % positiva o negativa, mientras que un IoC de manera general, puede contener darse en situaciones y elementos que no sean maliciosos, por ello conllevan tras su coincidencia un cierto peso y probabilidad que hay que valorar.

Por otro lado, los IoC se encuentran en el sistema y red del escenario del incidente de seguridad, haciendo referencia a consecuencias y rastros, mientras que las firmas siempre hacen referencia a una identificación unívoca de la amenaza.

Los IoC recogidos del equipo que ha sido comprometido, permiten relacionar ciertas condiciones del sistema con el ataque, permitiendo describir el incidente y conocer que hace y como funciona la amenaza [43]. En el caso en el que un malware infecte un equipo, se podrá hacer un análisis del sistema, obteniendo IoC del incidente, y estos se podrán comparar con los IoC recogidos en estudios previos para tratar de identificar el malware concreto y actuar como corresponda; y es por esta razón por la que al realizar un análisis, los resultados y los IoC generados son normalmente compartidos con la comunidad.

La base y potencia de los IoC es la distribución masiva de estos, permitiendo que todos los sistemas y equipos de seguridad se puedan beneficiar de los resultados de estudios y herramientas de análisis externos.

El principal problema que presentan es que es muy fácil que se queden anticuados, o que mediante ciertas variaciones en los incidentes, como pueden ser variantes y familias en el caso de malware, hagan que no sean efectivos; por ello el ciclo de vida que tienen suele ser de días.

Algunos ejemplos de lo que pueden ser indicadores de compromiso son: direcciones IP, hash de archivos, nombres de archivos, registros de windows y configuraciones, aumento del volumen de lectura de bases de

datos, actividades poco corrientes en cuentas de usuario con privilegios, alto volumen de tráfico inusual, de una localización ajena determinada...

3.1 Tipos

Los IoC se pueden dividir siguiendo diversos criterios, como el nivel de complejidad, la procedencia o clase de información que contengan.

Según el nivel de complejidad de los IoC, siguiendo lo comentado en el artículo [44], se pueden clasificar en:

- Atómicos: son los indicadores que cuya información no puede dividirse para dar más IoC, y cuyo significado y valor depende del contexto del incidente. Un ejemplo claro de este tipo de IoC son las direcciones IP.
- Computado: estos indicadores son el resultado de procesar información recopilada de un incidente, como pueden ser hashes de malware.
- Comportamiento: estos indicadores son colecciones de indicadores atómicos y computados, .

Los IoC de comportamiento representan las TTP de las amenazas, y están estrechamente relacionados con la CTI táctica . Estos indicadores son más complejos que los otros, y ofrecen resultados más detallados, aunque requieren más tiempo para ser procesados. Se podrían llegar a calificar como inteligencia.

Los IoC atómicos y computados están presentes en la CTI táctica; son más básicos, y dado que son fáciles de evadir, tienen un ciclo de vida más reducido. A pesar de las limitaciones que presentan estos tipos de IoC, permiten gestionar rápidamente un incidente de seguridad, consiguiendo resultados casi inmediatos; por ello son los más empleados y los más distribuidos. Además de que la información que tratan puede ser expresada en formatos que hacen la información mucho más manejables para las herramientas y equipos de seguridad, algo más complicado para los de comportamiento. Por todo esto estos tipos de indicadores son mas apropiados para lidiar con amenazas no muy complejas. Se podría decir que son observables, y en ellos no hay presente inteligencia.

Teniendo en cuenta esto, tal y como se sostiene en el artículo [44], la principal limitación que presenta el uso de IoC es que los que más se comparten y se pueden localizar son los atómicos y computados, que los más fáciles de ser evadidos, y por ello el uso general de los IoC no es tan efectivo como debía serlo.

Por otro lado, siguiendo una clasificación en función del origen del que proceden, se pueden distinguir los siguientes 3 grupos:

- Indicadores de Red: son URL o nombres de dominios empleados por las amenazas, ya sean sobre un servidor C2&C, webs maliciosas o equipos pertenecientes a una botnet. Al conocer que estos indicadores corresponden a IPs y dominios comprometidos son muy empleados en la detección de ataques, aunque presentan una limitación importante, y es que tienen un tiempo de vida muy limitado, pues lo único que es necesario para dejarlo anticuado es cambiar de servidor, dirección IP o dominio. A pesar de esto, el ir recolectándolos puede aportar información como el número de veces que se reutilizan esos parámetros, así como el tiempo entre que se cambian.
- Indicadores basados en Host: son extraídos de equipos mediante técnicas de análisis. Pueden ser nombres de malware, hashes de documentos, hashes de malware, DLLs, claves de registro,... Esta clase de indicadores cambian menos que los de red.
- Indicadores de Email: los atacantes emplean técnicas de ingeniería social a través del correo electrónico para realizar ataques. Estos indicadores recopilan información sobre el email y los servicios y servidores de los que proceden.

3.1.1 Pirámide del dolor

En 2013 el experto en ciberseguridad y experto en amenazas David Bianco creo la pirámide del dolor, en la que se representan los principales IoC que se pueden encontrar, ordenados según la complejidad que presentan, y que de manera proporcional son a su vez más efectivos contra las amenazas (en caso de utilizarse de manera correcta). En la Figura 3.1 se puede ver la pirámide [45, 46, 43].

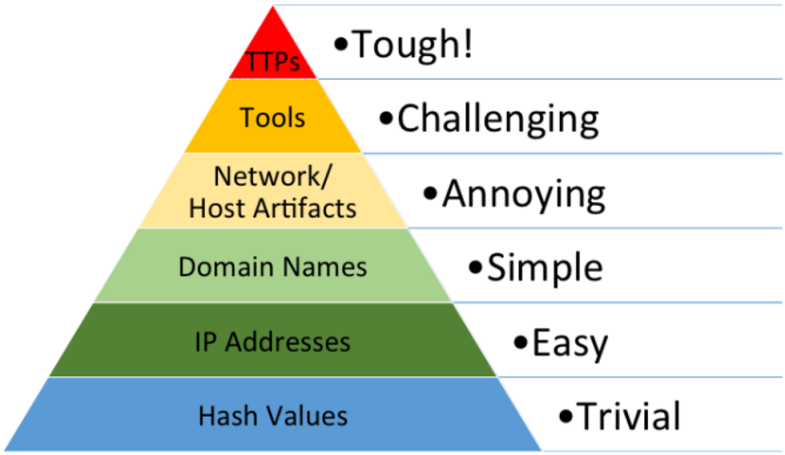

Figura 3.1 Pirámide del Dolor.

La intención de la pirámide es hacer referencia a lo distintos que son los IoC entre sí, ya sea por su efectividad, complejidad o las cantidades de ellos que se pueden encontrar.

- Hashes: El hash de un archivo malicioso hace referencia a él de manera prácticamente unívoca. Se emplea para identificar las diferentes muestras de malware o archivos que se ven involucrados en un ataque. Son muy fáciles de cambiar, pues al variar minimamente el archivo el hash ya no será el mismo.
- Direcciones IP: Todas las direcciones IP que se relacionan con un ataque se catalogan como maliciosas. Estas IP pueden ser la dirección de un servidor de comando y de control (C&C) o de una web maliciosa. El uso de las direcciones es fácil de esquivar simplemente cambiándolas.
- Nombres de Dominio: Hacen referencia y representan lo mismo que las direcciones IP, con la diferencia que son menos dinámicas al ser más difícil de cambiarlas, por ello se cuentan en un apartado diferente.
- Observables de Red o Host: Hacen referencia a todo tipos de detalles que se pueden ver envueltos en las actividades que se llevan a cabo en el ataque, como archivos, procesos, tráfico, librerías DLL.
- Herramientas: Estos indicadores contienen los patrones de comportamiento de estas herramientas, así como el tráfico y las firmas digitales que dejan a su paso.
- TTPs: Hace referencia a como trabaja el actor tras la amenaza en todos los pasos que lleva a cabo, la metodología en general. Si se llega a conseguir detectar y bloquear ataques en función de las técnicas, tácticas y procedimientos que lo describen los atacantes tendrán que crear una nueva estrategia, una nueva forma de atacar para poder conseguirlo.

Los tres primeros indicadores son los tradicionales y aunque son los más empleados y más abundantes son los menos efectivos a causa del tiempo de vida limitado que tienen. Mientras tanto, los otros tres son menos comunes, pero son mucho más efectivos pues el comportamiento del ataque es lo que lo define, siendo más complicado cambiar la forma de hacer lo mismo, porque aunque se pueda hacer lo mismo de numerosas maneras no son infinitas [46].

3.2 Definición y Formatos

Existen varias iniciativas, ya sean proyectos o estándares, que indican como definir los IoC, con que estructura y formato. Entre estas iniciativas hay algunas propias y específicas para definir los IoC, y otras que están orientadas a la CTI de manera general, y que además tratan los IoC. Estas iniciativas orientadas a la Inteligencia de Amenazas se encuentran en el punto 2.6.2.

Estas iniciativas potencian la base de los IoC, dándoles un formato unificado y compatible que no limite el contenido de estos, y que permita una compartirlos administrarlos y distribuirlos, y ser utilizados en herramientas de seguridad [25, 47].

Un IoC es normalmente un esquema, que puede estar escrito en diferentes lenguajes (txt,XML,yaml,...), en el que se especifican detalles sobre el sistema afectado por un ataque. Esos detalles se relacionan con operadores lógicos para poder evaluar si un sistema se ha visto comprometido realmente frente a la amenaza que hace referencia [42].

Algunas características claves de los IoC son[42]:

- Es un documento que está destinado a distribuir la información que contiene.
- El documento no es definitivo, la información que contiene puede cambiar y se puede adaptar.
- El esquema es flexible, se puede recoger cualquier tipo de evidencias sobre ataques, ya sean sobre un equipo determinado o generales para todos los sistemas.

3.2.1 Open Indicators of Compromise (OpenIoC)

Este modelo de definición de IoC fue desarrollado en 2011 por la empresa de ciberseguridad Mandiant, y desde 2013 es un proyecto libre bajo la licencia Apache 2.0. Los IoC son escritos en una estructura XML extensible contenida en un fichero ".IOC" [25].

Los IoC definidos bajo OpenIoC distinguen tres secciones [48]:

- Metadatos: una cabecera que contiene metadatos sobre el IoC en general, como autor y descripción.
- Criterios: esta sección contiene la información que permite hacer una evaluación con lógica booleana para determinar si la amenaza con la que se está tratando es la que referencia este IoC o no.
- Parámetros: sección que fue añadida en una nueva versión del modelo en la que se almacenan metadatos aplicables en la sección de Criterios.

3.3 Obtención de Indicadores de Compromiso

Los IoC se pueden conseguir de dos formas, obteniéndolos como fruto de intercambios, o bien extrayendolos del sistema en el que se haya producido el incidente de seguridad al que hacen referencia.

3.3.1 Distribución

Los IoC se intercambian en las TIPs, vistas en el punto 2.6.3 de la memoria, o además se pueden conseguir en bases de datos, repositorios o plataformas de intercambio específicas de IoC.

En los repositorios o bases de datos se pueden encontrar IoC compartidos por organizaciones de ciberseguridad y por la comunidad de usuarios [47].Estos son algunos repositorios de IoC:

- IOC Bucket
- IOC DB
- Citizen Lab Malware Indicator
- Threat fox

3.3.2 Generación de IoC

Los IoC se generan a partir de la información recolectada de un incidente de seguridad, en el ámbito de la informática forense. Estos pueden ser escritos de manera manual a partir del análisis del equipo, siguiendo el modelo de definición deseado, o pueden ser generados de manera automática, en el formato deseado, por una herramienta de análisis automático.

Además, existen herramientas que por separado permiten escribir IoC, como la herramienta IOC Editor de Fireeye.

En el artículo "Dridex: analysis of the traffic and automatic generation of IOCs"[49] se presenta un esquema de trabajo para generar de manera automática IoC de red a través del análisis dinámico de una muestra malware, mediante Cuckoo Sandbox, pudiendo dejar atrás la generación manual de los IoC, y las limitiaciones que esa

práctica trae consigo. Sin entrar mucho en detalles, se generan a partir de capturas de tráfico realizadas en el análisis dinámico.

En el artículo "A systems aproach to indicators of compromise utilizing graph theory" [50] se introduce una aplicación de la teoría de grafos, que permite crear IoC de host, que describen de manera completa un incidente de seguridad, teniendo en cuenta todo el entorno.

3.4 Empleo de IoC

Los IoC se emplean tanto en detección como en identificación de amenazas en sistemas informáticos, lo cual es clave, pues una rápida identificación de la amenaza permite poder actuar cuanto antes de manera correcta, y reducir el tiempo de respuesta ante el incidente, llegando incluso a poder evitarlo. Son muy comunes en la detección de malware.

Una vez se tienen los IoC, ya se hayan obtenido de repositorios de distribución o se hayan generado a partir de estudios de malware, para trabajar con ellos lo que se hace es recopilar información de un equipo y analizarla en busca de indicadores de compromiso, de forma que se si localizan significará que el equipo ha sido o está siendo comprometido [35].

Para recopilar la información del equipo y localizar si hay IoC, algunas de las herramientas que se pueden encontrar son:

- IOC Finder de Fireeye
- Redline de Fireeye
- Loki
- Splunk

Además existen herramientas de ciberseguridad de diferentes campos que pueden utilizar IoC para funcionar:

- Sistema de detección de intrusos (IDS)
- Sistema de prevención de intrusos (IPS)
- Security information and event management system (SIEM)
- Cortafuegos
- Antivirus

3.5 Indicadores de Ataque

Los Indicadores de Ataque (IoA) hacen referencia a una serie de acciones o pasos que el actor malicioso realiza para generar un incidente de seguridad. Al tener estos indicadores el conocimiento de la forma de actuar de las amenazas, se pueden monitorizar esos pasos, e identificar cuando ocurre un ataque, y poder así pararlo a tiempo para evitar que se llegue a ejecutar con éxito. [51, 52]

La forma de trabajar con los IoA es más proactiva que con los IoC, debido a que los IoC se extraen una vez el incidente se ha producido, y estos pueden estar relacionados con el final del ataque, permitiendo casi unicamente detectar el incidente, siendo de poca ayuda para evitar el incidente, mientras que los IoA se monitorizan en tiempo real en busca de ataques en curso.

Los IoA tienen mayor tiempo de vida que los IoC, porque los IoA hacen referencia al comportamiento de la amenaza, que es más general que un rasgo concreto fruto de hacer las cosas de una determinada manera. Ambos se utilizan de manera simultánea en la detección de ataques e incidentes.

Los IoC entran en los subgrupos de CTI técnica y táctica, mientras que los IoA representan las intenciones y TTPs tras un ataque, por lo que entran dentro de la CTI estratégica. Ejemplos de IoC son direcciones ip, firmas de amenazas; y ejemplos de IoA son robo de credenciales, uso de servidores C&C...

4 Detección real de malware mediante indicadores de compromiso

Para realizar los análisis será necesario realizar las siguientes tareas:

- Obtener las muestras malware que se van a analizar, para identificar si contienen malware, y que malware en particular.
- Preparar el entorno de trabajo para realizar los análisis, instalando y configurando las herramientas.
- Conseguir los IoC que se van a proporcionar a las herramientas seleccionadas para que lleven a cabo los análisis.
- Carga de los IOC en la herramientas Cuckoo, Loki y MISP, según los formatos soportados y disponibles.

4.1 Obtención de muestras de malware

El formato de las muestras malware escogidas es Éxecutable and Linkable Format', que aparece entre otras con la extensión ".elf". Es un formato de archivo estándar utilizado entre otras cosas para ejecutables, códigos de objeto, volcados de núcleo y bibliotecas compartidas. Es muy flexible al no estar limitado a ninguna arquitectura o procesador. Con el tiempo se ha ido adoptando como el formato para archivos binarios en Unix y sistemas basados en Unix, como es el caso de Linux, que es el sistema operativo objetivo para el que queremos analizar el malware.

Algunos de los repositorios de muestras malware que existen son:

- VirusShare
- Contagio
- The Zoo
- Malshare
- Das Malwerk
- Malware bazaar

Además se pueden encontrar repositorios y códigos fuente de malware filtrados en Github. Para el proyecto se han obtenido de Malware Bazaar. La razón de emplear esta web es la posesión de muestras de malware destinado a Linux, que es lo que se busca, y que además permite descargarlas con total libertad sin necesidad de registrarse o tener ningún permiso especial. En la Tabla 4.1 están las muestras que se han obtenido y empleado en la detección.

Tabla 4.1 Muestras malware descargadas de Malware Bazaar.

n.	Malware	SHA256	Tipo
1	BlackMatter	6a7b7147fea63d77368c73cef205eb75d16ef209a246b05698358a28fd16e502	-
2	CobalStrike	6352be8aa5d8063673aa428c3807228c40505004320232a23d99ebd9ef48478a	exploit - evad
3	CoinMiner	4380d5ccae16ac81d05be6b6be20754cb1bbe421929fa8fcefa629ff5878518a	troj.min
4	CoinMiner	bf8dc5eca570a1a0d702303547b736cff9df54c31745dde90dfc429580c0cc28	troj.min
5	CoinMiner	f72babf978d8b86a75e3b34f59d4fc6464dc988720d1574a781347896c2989c7	troj.min
6	Conti	95776f31cbcac08eb3f3e9235d07513a6d7a6bf9f1b7f3d400b2cf0afdb088a7	Ransomware
7	DarkSide	c93e6237abf041bc2530ccb510dd016ef1cc6847d43bf023351dce2a96fdc33b	Ransomware - 64b
8	DarkSide	984ce69083f2865ce90b48569291982e786980aeef83345953276adfcbbeece8	Ransomware - 64b
9	DarkSide	9cc3c217e3790f3247a0c0d3d18d6917701571a8526159e942d0fffb848acffb	Ransomware - 64b
10	eCh0raix	24b5cdfc8de10c99929b230f0dcbf7fcefe9de448eeb6c75675cfe6c44633073	Ransomware - Qnap
11	eCh0raix	3d8d25e2204f25260c42a29ad2f6c5c21f18f90ce80cb338bc678e242fba68cd	Ransomware - Qnap
12	Gafgyt	14d2209984ec4a688c2ea085feaa817bde27d4604aeb4ee607bf0fe43fe7b04e	trojan
13	Gafgyt	2cc82477f23ac0030fe702b27d6cc7cc77671b25e259120a4d44830a3b8be0fe	trojan
14	Mirai	085adbea50f92bcc93970e0078c770fd073f87bec0c2dbed75a7a3b094d8de10	trojan
15	Mirai	44f5b34ef8beffac75f75ac0568a852181557d84e7753326c28ed51a87801597	trojan
16	Sudinokibi	a322b230a3451fd11dcfe72af4da1df07183d6aaf1ab9e062f0e6b14cf6d23cd	Ransomware
17	Sudinokibi	f864922f947a6bb7d894245b53795b54b9378c0f7633c521240488e86f60c2c5	Ransomware
18	Tsunami	41a8fde38a947b392f87960ef6cf6f32c0b56e0348c047dea912ab140aa2a147	Backdoor
19	Tsunami	e088410e995c9ee2f51c45757bec3bb337ecda0b20cab9de1520bf892cd84da3	Backdoor
20	Tsunami	41a8fde38a947b392f87960ef6cf6f32c0b56e0348c047dea912ab140aa2a147	Backdoor
21	Winnti	2f1321c6cf0bc3cf955e86692bfc4ba836f5580c8b1469ce35aa250c97f0076e	Backdoor
22	XorDDos	d920dec25946a86aeaffd5a53ce8c3f05c9a7bac44d5c71481f497de430cb67e	Trojan
23	Panchan	7184b2064739ce5ff09ce5cea8fe0d61dfd0bb3124f35c1ef1f359d6806ab213	Trojan

4.2 Preparación del entorno de trabajo

4.2.1 Herramientas

Las herramientas empleadas son Cuckoo Sandbox, Loki y MISP, todas gratuitas y de software libre. Hay un apéndice para cada una de ellas, en el que se comenta su instalación y uso de ciertas utilidades con las que cuentan. Además se pueden obtener los scripts empleados en su instalación, configuración y uso en un repositorio creado para el estudio [6].

Cuckoo Sandbox

Cuckoo Sandbox es una herramienta que realiza análisis de malware de manera automática y que permite obtener resultados detallados sobre el análisis de un archivo malicioso en Windows, macOS, Linux, y Android, todo realizado en un entorno aislado preparado para ser real y seguro para el equipo. La herramienta cuenta con diferentes módulos que permiten trabajar junto con TIPs y otras herramientas.

El proceso de instalación y configuración de la herramienta se encuentra detallado en el apéndice A.

Una vez se haya instalado y configurado, antes de arrancar Cuckoo es necesario verificar que las interfaces de red con las máquinas virtuales están activas y configuradas, y que haya conexión. Además debe haber conexión con los posibles servicios de terceros que se pueden configurar, en el caso del proyecto el único que se ha configurado es un servidor MISP.

Para arrancar Cuckoo, habiéndolo instalado siguiendo los pasos del apéndice A, se hace a través del nuevo usuario que se ha creado para trabajar, y en cuya carpeta personal se encuentra el directorio de configuración "/home/cuckoo/.cuckoo/".

Se pueden consultar las distintas opciones de arranque y la ayuda sobre todos los comandos que ofrece Cuckoo:

```
cuckoo --help
```

Además en cada comando se pueden ver las opciones que soporta de la misma manera:

```
cuckoo command --help
```

Por ejemplo, en caso de querer visualizar los logs de debug habría que añadir la opción "-d" al comando de arranque.

Cuckoo cuenta con una interfaz web, que se debe arrancar de manera independiente al núcleo del programa. Esta interfaz permite visualizar la información básica sobre los recursos del equipo en el que está instalado, información sobre los análisis, las estadísticas de uso, y realizar los análisis sin necesidad de emplear la terminal. Para ejecutar el servidor web es necesario arrancarlo en una terminal independiente a en la que está arrancado Cuckoo:

```
cuckoo web runserver
```

De forma alternativa al uso de la interfaz gráfica, se pueden realizar los análisis desde la línea de comandos.

Además Cuckoo cuenta con el soporte para numerosas herramientas que emplea para completar y enriquecer los análisis de las muestras malware. Entre estos módulos de procesamiento se encuentran MISP, VirusTotal, Suricata y Snort además de otros. Por defecto cuenta con el soporte de reglas Yara. Además es posible obtener firmas y reglas creadas por la comunidad mediante el comando:

```
cuckoo community
```

MISP

Malware Information Sharing Platform (MISP) es una TIP desarrollada para que profesionales de la ciberseguridad puedan almacenar, correlacionar y compartir CTI e IoC sobre malware y otros incidentes de seguridad [53]. Permite reunir diversas feeds, intercambiar información con otras instancias de la plataforma e interactuar con herramientas de seguridad, tanto para alimentarlas con información como para obtener los resultados que vayan generando. Para poder lograr cualquier interacción con herramientas y otras instancias, así como cualquier uso automatizado que se haga de la plataforma, deberá estar soportado su uso [54, 53].

El funcionamiento de la plataforma, así como la forma de trabajar con ella se puede ver de forma detallada en [55]. La herramienta es una plataforma que puede ser ejecutada por todo el mundo, de forma que cada servidor que se haya habilitado será una instancia única, la cual puede pertenecer a una persona concreta o a una organización. Existen numerosas instancias que pertenecen a organizaciones de seguridad privadas y públicas, en las que se pueden encontrar CTI, IoC e información relacionada con estudios e incidentes de numerosas amenazas.

Loki

Loki es una herramienta que busca las coincidencias de reglas Yara e IoC en un equipo, pudiendo así determinar si se da una coincidencia correspondiente a una amenaza [56]. Además de las reglas Yara, las coincidencias que se pueden localizar con los IoC son nombres de archivo, hashes y conexiones con equipos remotos.

Para consultar las opciones que soporta el programa este se deberá ejecutar con la opción "-h" (con el comando "python loki.py -h"), para lo que el programa devolverá todas las opciones disponibles:

```
optional arguments:
-h, --help   show this help message and exit
-u URL       MISP URL
-k APIKEY    MISP API key
-l tframe    Time frame (e.g. 2d, 12h - default=30d)
-o dir       Output directory
-y yara-dir  YARA rule output directory
--verifycert Verify the server certificate
--debug      Debug output
```

4.2.2 Equipos

El sistema operativo para el que se quieren detectar los malware es Linux, y aunque podrían utilizar herramientas en otros sistemas, las herramientas que se van a utilizar están destinadas a funcionar en Linux.

El entorno de trabajo consta de un ordenador principal, y tres equipos virtuales. El equipo principal es un ordenador físico que tiene un i7-7700HQ, con 256Gb de espacio y 16Gb de RAM, con el sistema operativo Ubuntu 20.04.04 LTS. Los otros tres equipos virtuales están emulados sobre este con VirtualBox, con sistema operativo Ubuntu Mate 20.04.4 LTS, con asignación de 2 núcleos de procesador, 30Gb de espacio y 4Gb de RAM.

En el equipo principal se ha instalado Cuckoo Sandbox, el cual utilizará Virtualbox para analizar las muestras malware en un equipo virtual aislado (uno de los 3 equipos virtuales), permitiendo que el análisis sea seguro y efectivo, haciendo que la muestra vea el equipo virtual como un equipo real y no pueda emplear técnicas de evasión de análisis dinámico.

En otro equipo virtual se instala el servidor MISP, el cual se conectará con Cuckoo Sandbox para recoger los resultados de sus análisis y poder correlacionarlos con la CTI de la que disponga.

En el tercer equipo virtual se instala la herramienta Loki. Esta máquina virtual antes de comenzar a usarse para ejecutar los malware y realizar análisis debe de aislarse de la red.

En la Figura 4.1 se puede ver la configuración de red. La máquina virtual utilizada por Cuckoo está conectada a la red interna de Virtualbox 192.168.56.0/24, y la máquina virtual de MISP está conectada a otra red interna de Virtualbox 196.168.157.0/24. Ambas redes internas de Virtualbox son accesibles por el ordenador principal.

A la hora de trabajar con VirtualBox para realizar análisis es recomendable no instalar las Virtualbox addons para que el acceso al ordenador principal pueda realizarse únicamente por red y no alterar el aislamiento de las máquinas, y que pueda desactivarse cuando quiera crearse un entorno aislado. Además, emplear las snapshots de Virtualbox para guardar el estado de las máquinas virtuales facilita mucho el trabajo para poder modificar configuraciones en caso de error en las instalaciones, y por supuesto para restaurar los equipos virtuales tras ejecutar malware.

4.3 Obtención de indicadores de compromiso

Tal y como se ha comentado en capítulos anteriores, se pueden obtener los IoC o bien a través de un repositorio, o generandolos a partir de una muestra malware.

En la Tabla 4.2 se encuentran fuentes de las que se han obtenido IoC en distintos formatos, los cuales se pueden utilizar con las herramientas anteriormente vistas.

En cuanto a los formatos en los que vienen los IoC obtenidos:

- Lo formatos trasversales ".txt" o ".csv", que no son propios del ámbito de la ciberseguridad, en este caso portan el valor de los indicadores, sin ninguna estructura propia de la CTI.
- Los indicadores que vienen en el formato ".json" contienen la estructura de eventos MISP.

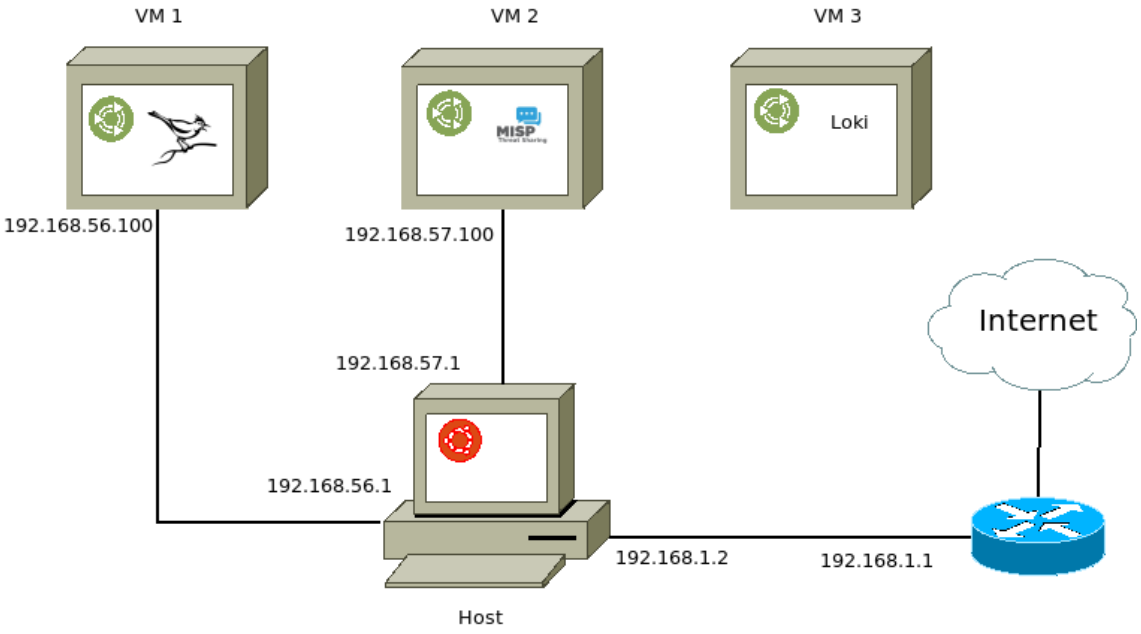

Figura 4.1 Diagrama de red del entorno de trabajo.

- "STIX" y "openioc" son formatos propios de la CTI y los IoC.

Tabla 4.2 Formatos de IoC de los respositorios de descarga.

Repositorios	Formato	Malware	Tipo IoC
thesecmaster	text	Rapperbot	hash, url, ip,
socinvestigation	text	Symbiote	hash, puerto, dominio, nombre proceso, nombre fichero,
cobaltstrike-extraneous-space	csv	CobaltStrike	ip, puerto
threatfox	csv - json	—	hash, ip, domino, url
iocbucket	openioc	—	—
malware-indicators	csv - MISP json - openioc - STIX	—	—

Por otro lado, para generarlos será necesario obtener el valor del campo o valor del IoC (el mismo que portan los ficheros .txt y .csv), el cual definirá el tipo de IoC que se está generando. Ese campo puede ser, entre muchas cosas, el valor del hash del archivo malicioso, el cual sería además de un indicador una firma del propio malware, al ser algo único del malware, o por ejemplo una dirección ip o dominio a los que se conecta.

4.4 Carga de indicadores y reglas

4.4.1 Cuckoo

En Cuckoo se han cargado reglas Yara de forma manual, en el directorio "./.cuckoo/yara/" ubicado en la carpeta personal del usuario. Para poder utilizarlas el volcado de memoria deberá estar activado. Además en MISP se podrán utilizar los IoC y CTI que tenga la plataforma para relacionarlos con los resultados extraídos de los análisis hechos en Cuckoo.

4.4.2 MISP

En la herramienta la información está estructurada en eventos, los cuales se describen mediante una lista de atributos de diferentes tipos. Los eventos se relacionan entre sí mediante los atributos que comparten, lo que permite observar coincidencias y llevar a cabo detección de amenazas y ampliar estudios. Los eventos, así como los atributos, pueden ser importados o creados, ya sea manualmente o por una herramienta que se haya conectado a la plataforma.

En la página principal, la cual se puede ver en la Figura 4.2, se puede ver la lista de eventos existentes, además haciendo clic en su id se podrá acceder a el para ver toda la información y atributos que tiene, como se ve en la Figura 4.3, y poder desde ahí añadirlos.

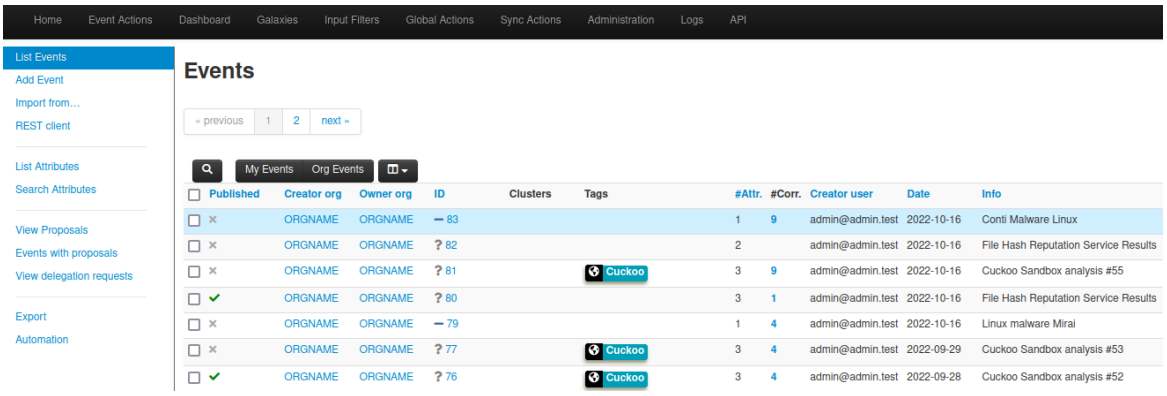

Figura 4.2 Página principal de MISP.

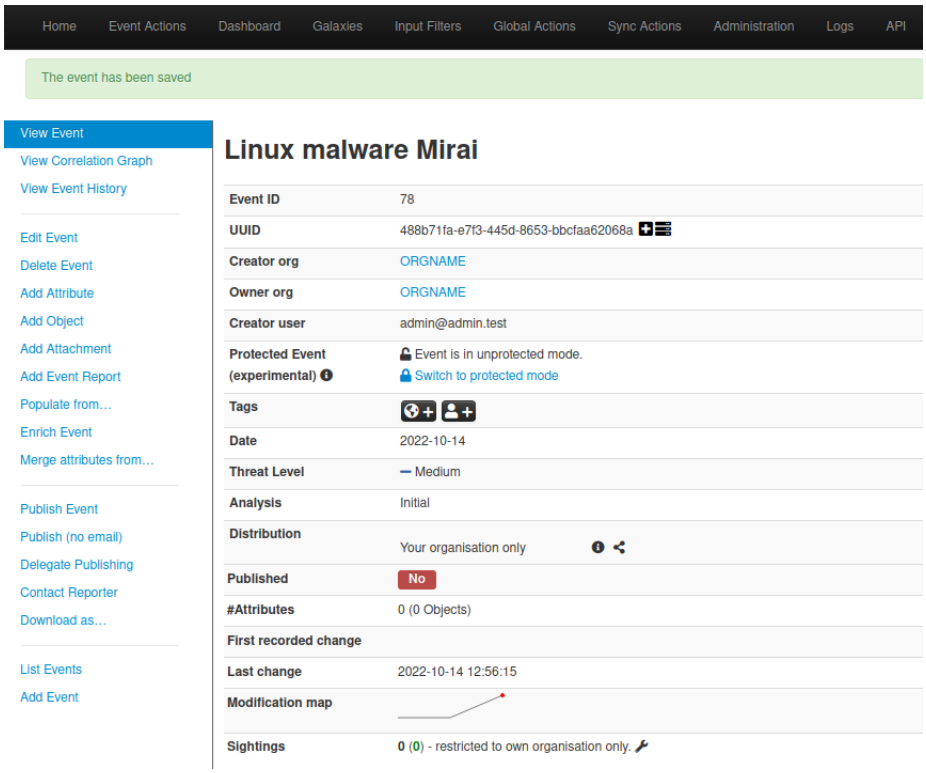

Figura 4.3 Visualización de un evento MISP.

De forma alternativa se podrán ver los atributos en una lista en "List Attributes" desde la página principal, ni necesidad de tener que ver por separados los de cada evento.

En la página principal se encuentra la opción "Import from..." que da la opción a importar información. Por defecto la importación de CTI soporta 3 formatos: STIX 1.1.1 (.XML), STIX 2.0 (.JSON) , y eventos en formato MISP (.JSON). Para que esto sea posible los campos deben ser los que soporta MISP, los cuales se pueden encontrar en la documentación[57, 58]. En el repositorio de github existen ficheros de muestra para importar.

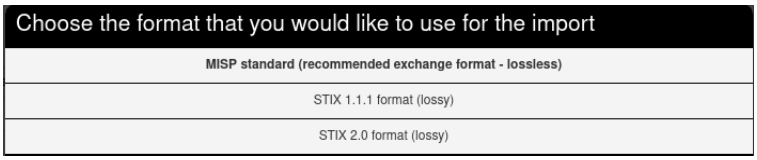

Figura 4.4 Menú de importación de eventos.

Además de manera adicional soporta importación en formato ’.csv’y openioc (.ioc), para lo que es necesario configurar los módulos de importación correspondientes.

Código 4.1 Formato STIX 1.1.1 (.XML) para importar en MISP un IoC hash.

```
<stix:STIX_Package
xmlns:FileObj="http://cybox.mitre.org/objects#FileObject-2"
xmlns:cybox="http://cybox.mitre.org/cybox-2"
xmlns:cyboxCommon="http://cybox.mitre.org/common-2"
xmlns:cyboxVocabs="http://cybox.mitre.org/default_vocabularies-2"
xmlns:example="http://example.com"
xmlns:indicator="http://stix.mitre.org/Indicator-2"
xmlns:stix="http://stix.mitre.org/stix-1"
xmlns:stixCommon="http://stix.mitre.org/common-1"
xmlns:stixVocabs="http://stix.mitre.org/default_vocabularies-1"
xmlns:ttp="http://stix.mitre.org/TTP-1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" id="example:Package-
    bc2955f8-f1bb-4f02-b2ed-339d7daf6d75" version="1.2">
<stix:STIX_Header>
<stix:Title>File Hash Reputation Service Results</stix:Title>
<stix:Package_Intent xsi:type="stixVocabs:PackageIntentVocab-1.0">Indicators
    - Malware Artifacts</stix:Package_Intent>
</stix:STIX_Header>
<stix:Indicators>
<stix:Indicator id="example:indicator-14975dea-86cd-4211-a5f8-9c2e4daab69a"
    timestamp="2015-07-20T19:52:13.853585+00:00" xsi:type=’indicator:
    IndicatorType’>
<indicator:Title>File Reputation for SHA256=
    e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855</
    indicator:Title>
<indicator:Type xsi:type="stixVocabs:IndicatorTypeVocab-1.1">File Hash
    Watchlist</indicator:Type>
<indicator:Observable id="example:Observable-7b97c8a2-2d0b-4af7-bcf0-
    cad28f2fea5a">
<cybox:Object id="example:File-b04bfc7c-04ae-4dfe-ba8e-a297f0717552">
<cybox:Properties xsi:type="FileObj:FileObjectType">
<FileObj:Hashes>
<cyboxCommon:Hash>
<cyboxCommon:Type condition="Equals" xsi:type="cyboxVocabs:HashNameVocab
    -1.0">SHA256</cyboxCommon:Type>
```

```
<cyboxCommon:Simple_Hash_Value condition="Equals">
    e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855</
    cyboxCommon:Simple_Hash_Value>
</cyboxCommon:Hash>
</FileObj:Hashes>
</cybox:Properties>
</cybox:Object>
</indicator:Observable>
<indicator:Indicated_TTP>
<stixCommon:TTP id="example:ttp-23e715a9-24c8-4b21-ba5b-f564d2edc660"
    timestamp="2015-07-20T19:52:13.854415+00:00" xsi:type='ttp:TTPType'>
<ttp:Title>Malicious file</ttp:Title>
</stixCommon:TTP>
</indicator:Indicated_TTP>
<indicator:Confidence timestamp="2015-07-20T19:52:13.854506+00:00">
<stixCommon:Value vocab_reference="https://en.wikipedia.org/wiki/Percentage"
    vocab_name="Percentage">75</stixCommon:Value>
</indicator:Confidence>
</stix:Indicator>
</stix:Indicators>
</stix:STIX_Package>
```

Para añadir de forma manual CTI se puede registrar un evento, y después asociarle atributos. Para crear un evento se hace desde "Add event" en la página principal, y después accediendo al evento, se podrá ver su información y crear y añadir objetos y atributos, tal y como se ver en la Figura 4.3. A la hora de crear los eventos y atributos será necesario indicar los valores correspondientes a los campos que los definen, tal y como se puede apreciar en Figura 4.5 y Figura 4.6.

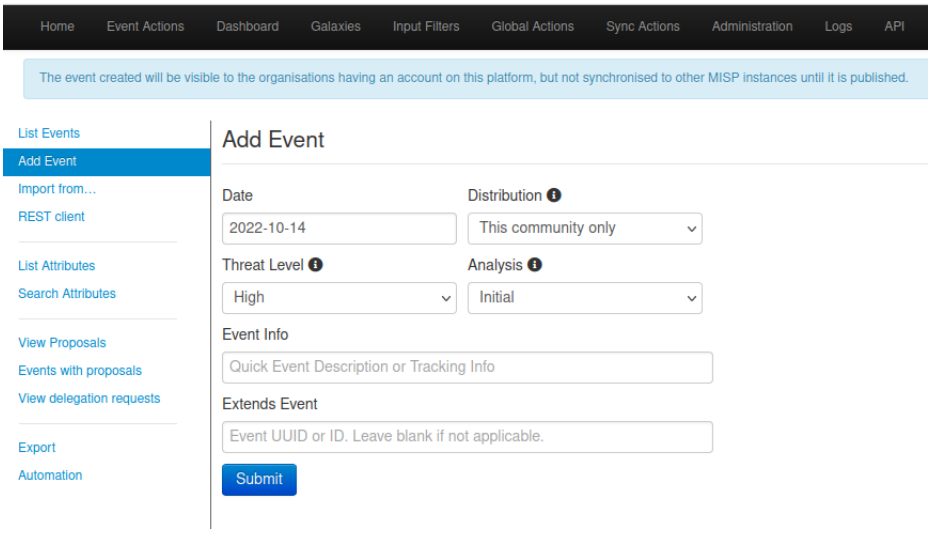

Figura 4.5 Panel MISP para crear un evento.

Los campos que forman los eventos:

- **Date**: La fecha en la que sucedió el incidente que se pretende describir con el evento.
- **Distribution**: Este campo permite configurar el alcance que tendrá a la hora de ser distribuido, configurando quieren podrán verlo, tanto en el propio servidor como entre servidores. Entre las opciones se encuentran:
  - **Your organization only**: únicamente los miembros de la organización a la que pertenece el usuario que crea el evento podrán verlo, ya sea en esta o en otra instancia de MISP.

  - **This Community-only**: los miembros de la comunidad MISP a la que pertenece el usuario que crea el evento serán quienes podrán verlo únicamente. Estos usuarios serán los pertenecientes a todas las organizaciones que se encuentren en la instancia MISP, así como las que que pertenezcan a servidores que sincronicen con este.
  - **Connected communities**: hace referencia a los miembros de la opción anterior y además a los de las comunidades que sincronicen con los servidores de las comunidades presentes en este servidor.
  - **All communities**: el evento podrá ser visto y propagado por todos los usuarios de las comunidades.
  - **Sharing group**: los usuarios que podrán ver este evento deben pertenecer a las comunidades seleccionadas en un grupo personalizado.

- **Threat Level**: Este campo indica el nivel de peligro que supone el incidente. Se puede dejar indefinido o con uno de los siguientes valores:

  - **Low**: malware común.
  - **Medium**: se trata de una APT.
  - **High**: se tratan de APTs sofisticadas y ataques que emplean vulnerabilidades de día 0.

- **Analysis**: Indica el estado actual del análisis y estudio del incidente del evento. Entre los valores que se pueden indicar están:

  - **Initial**: el análisis acaba de empezar.
  - **Ongoing**: el análisis está en progreso.
  - **Completed**: el análisis se ha finalizado.

- **Event Info**: campo en el que indicar toda información relevante sobre el suceso y el malware.

Figura 4.6 Panel MISP para crear un atributo.

Los campos que forman los atributos:

- **Category**: La categoría del atributo indica el aspecto del malware que se está describiendo.

- **Type**: Dependerá de la categoría. Explica el punto desde el que se explica el aspecto del malware. Por ejemplo, una dirección IP origen de un ataque permite describir la carga de envío de un malware. Se pueden ver todos los atributos y categorías que hay en la documentación de la herramienta[53, 57].
- **Distribution**: Este campo es el mismo que el de la creación de eventos, y soporta los mismos valores. Se le aplicará el campo más restrictivo de los dos.
- **Value**: El valor del atributo.
- **For Intrusion Detection System**: Esta opción permite que el atributo pueda ser utilizado como firma para realizar detección automática en IDS.
- **Batch import**: Indica si el valor es una lista de atributos del mismo tipo, cada uno en una línea, permitiendo así importarlos de manera automática. Importante recordar que tendrán la misma categoría y tipo y permanecerán al mismo incidente de seguridad, y por consecuente a la amenaza que esté detrás.
- **Disable Correlation**: Permite desactivar la correlación con el resto de atributos a los que se tiene acceso.
- **Contextual Comment**: Comentario sobre el atributo.

Además de la inserción o importación de CTI de forma manual, es posible importarla de manera automática a través de repositorios y feeds; esas conexiones se pueden configurar en la sección "Sync Actions".

Finalmente se ha optado por por crear los eventos y atributos manualmente (tal y como se ve en la Figura 4.7) para tener un mayor control sobre todos los campos que tiene MISP. Los eventos se corresponden con los 4 malware seleccionados.

Figura 4.7 Inserción de un hash de malware como atributo en un evento en MISP.

4.4.3 Loki

Los IoC y reglas Yara que se emplean en los análisis es necesario añadirlos, para ello, una vez recopilados y escritos según lo requiere la aplicación, se introducen en una carpeta del directorio de trabajo de Loki llamada "./Loki/signature-base/", en el que habrá una carpeta para IoC "./iocs/" y otra para las reglas Yara en la carpeta "./yara/".

Las reglas Yara deben seguir su propia sintaxis, y los IoC no necesitan ningún formato u estructura estandarizada, simplemente deben estar en texto plano, siguiendo una estructura de separación concreta y un sistema de nombrado de archivos que se comenta en la documentación del proyecto de Github "signature-base" [59] del creador del proyecto "Loki" [56]. Los ficheros en que contienen IoC y que están ubicados en la carpeta "iocs" deberán tener en el nombre al menos la cadena "hash", "filename" o "c2" para identificar el tipo de IoC que contendrá el fichero.

La estructura de separación que tendrá cada tipo de fichero es la siguiente:

- fichero de IoC de conexiones:

```
# COMMENT
c2-server.tld
ip-address
```

- fichero de IoC de hash

```
MD5;COMMENT
SHA1;COMMENT
SHA256;COMMENT
```

- fichero de IoC de nombres de ficheros:

```
# COMMENT
REGEX;SCORE
```

4.5 Ejecución y resultados de los Análisis

4.5.1 Cuckoo

Para realizar los análisis desde la línea de comandos se empleará el comando siguiente, el cual cuenta con numerosas opciones:

```
cuckoo submit archivo [opciones]
```

Se pueden indicar diferentes opciones:

- "- -timeout 300": tiempo en segundos que durará el análisis.
- "- -enforce-timeout": evita que el análisis acabe antes de tiempo.
- "- -machine nombre": nombre de la máquina virtual a emplear.
- "- -package tipo": el tipo de archivo que le damos. En caso de no colocarlo el propio programa lo determinará. No vale cualquier valor, de modo que en caso de no conocer el adecuado no se debe indicar para no generar un error.
- "- -options password=contraseña": contraseña del .zip encriptado.

La capacidad de poder trabajar con la terminal ofrece mayor flexibilidad a la hora de automatizar los análisis en caso de querer realizar muchos, permitiendo encolarlos para que la herramienta trabaje a su ritmo sin necesitar atención.

En la interfaz web se pueden ver todos los análisis que se han realizado, a los que se asocian fecha, nombre del archivo, información sobre el archivo como tipo, hash, tamaño, nombre... Después sobre los análisis realizados se pueden ver las firmas y reglas Yara que han coincidido. Además, al emplear MISP se podrán ver en la plataforma(MISP) los atributos que se han generado a partir de los resultados, y observar las

correspondencias que puede haber con otros incidentes, y que puede dar señales sobre la detección de una amenaza.

Los resultados de un análisis de malware en Cuckoo se muestran como en la Figura 4.8. Además, como MISP y Cuckoo están conectados, los resultados de Cuckoo se guardan en MISP, lo que permite realizar correlaciones con los IoC y CTI existente en MISP, como ocurre con los IoC que hemos añadido, se puede ver esa relación en Figura 4.9 y en Figura 4.10, entre los eventos creados por Cuckoo y los creados de manera manual. El orden en el que son creados los eventos manuales o de Cuckoo no altera los resultados salvo por el momento en el que se obtienen las relaciones entre ellos.

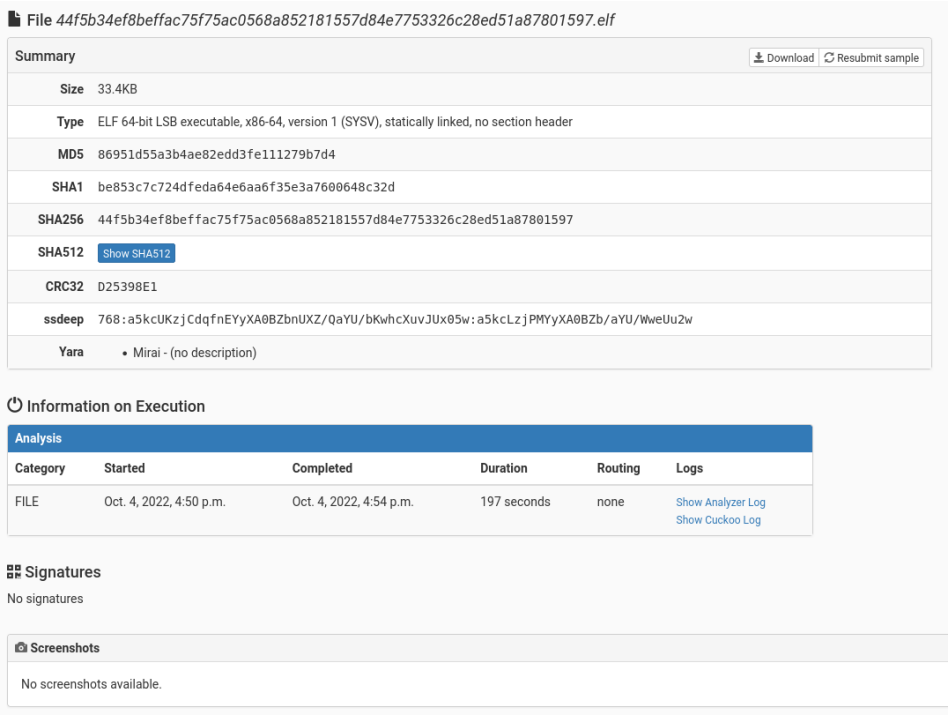

Figura 4.8 Resultado de un análisis de malware con Cuckoo Sandbox.

4.5.2 MISP

Una vez realizados los análisis en Cuckoo, se crean eventos que los identifican, descritos con atributos que Cuckoo recava en el análisis.

Tras incluir IoC sobre malware en distintos eventos, y realizar diferentes análisis en Cuckoo, en la página principal de MISP se observan esos eventos. Como se ve en la Figura 4.9 aparecen el número de correlaciones que hay con el resto de los eventos y el número de atributos que hay.

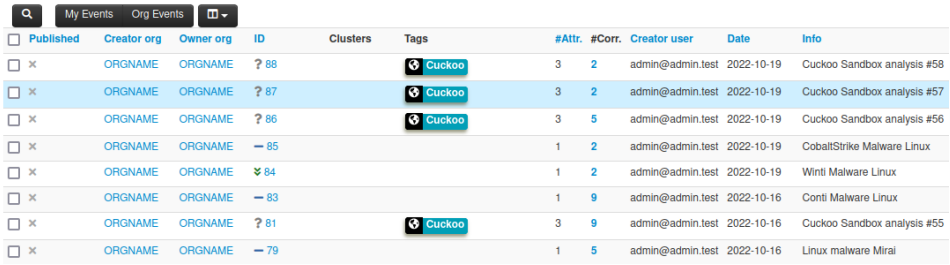

Figura 4.9 Resultado de un análisis de malware con Cuckoo Sandbox visto desde MISP.

Accediendo a los eventos se puede ver la sección de la Figura 4.10, en la que se pueden ver los eventos que

comparten algún atributo, de esta forma se pueden relacionar diferentes incidentes y describir mejor las muestras analizadas, llegando a identificarlas como dañinas en caso de que Cuckoo no las clasificara como tal.

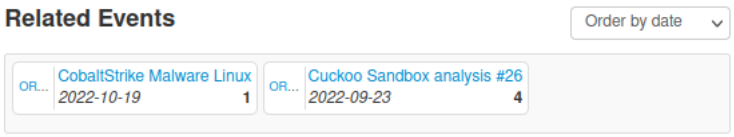

Figura 4.10 Eventos relacionados con el resultado de un análisis de malware con Cuckoo Sandbox en MISP.

4.5.3 Loki

Al realizar las tareas de escaneo y detección se podrán indicar diferentes opciones que modificarán el comportamiento de la herramienta. Se podrá especificar si únicamente se quiere escanear el sistema de archivos o un directorio concreto, o por otro lado simplemente se quieren analizar los procesos y conexiones, por defecto se realizan ambas tareas.

Al arrancar el programa se mostrará como en la Figura 4.11, mostrando logs de información que indican que se han cargado los indicadores y reglas Yara que se emplearán en los análisis.

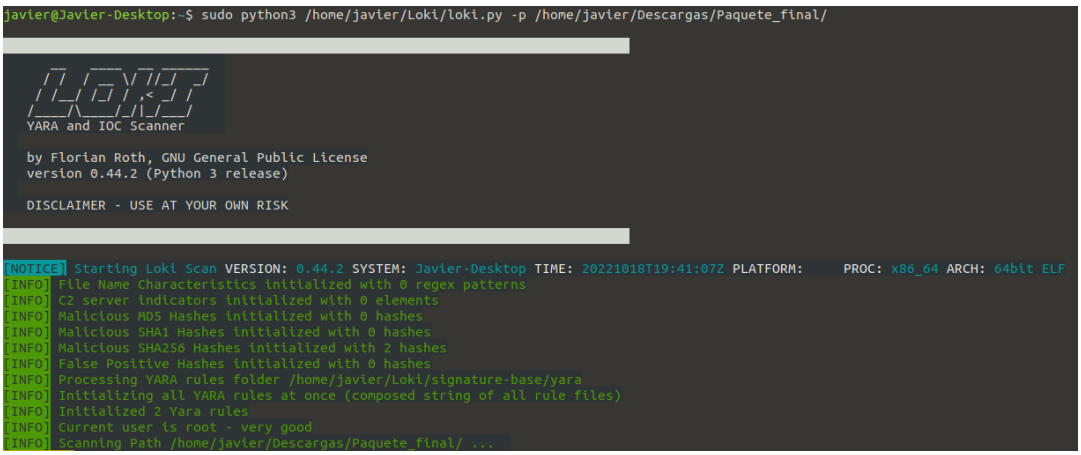

Figura 4.11 Inicio del programa Loki IoC scanner.

Mientras se van realizando los análisis irán apareciendo alertas que pueden ser verdes, para cuando no hay coincidencias de los elementos escaneados, amarillas para avisos, y rojas para coincidencias peligrosas. En la Figura 4.12 se puede apreciar como se visualizan, además del resumen final al terminar de ejecutarse la aplicación.

Los resultados tras los análisis de los 4 malware mediante Loki se pueden ver en la Figura 4.12, en la que se pueden apreciar en como se ven las coincidencias habiendo empleado reglas Yara o IoC.

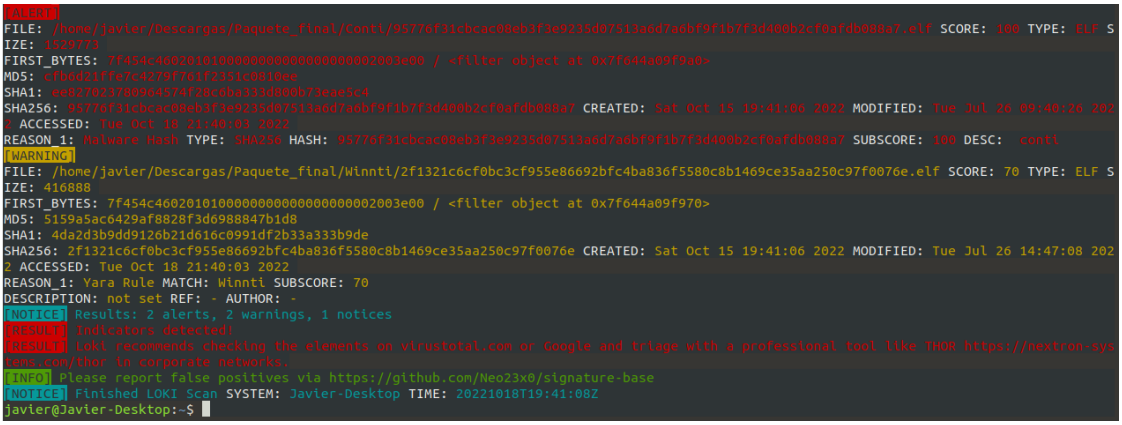

Figura 4.12 Resultados de una detección por hashes en Loki.

5 Validación y análisis de los resultados

5.1 Validación del estudio

La detección de malware se ha llevado a cabo siguiendo lo comentado en el apartado 4, utilizando los IoC que se han obtenido de los repositorios comentados, y que se han agrupado en el repositorio "IoC-linux-malware" [6]. Se han analizado las 23 muestras malware de la tabla Tabla 4.1, habiendo previamente cargado indicadores de compromiso con valores HASH de ficheros, dominios y direcciones IP, en sus correspondientes formatos a las herramientas. La herramienta loki ofrece la siguiente contabilidad de ioc, que indica el número de IoC empleados de cada tipo:

- SHA1 Hashes: 7313
- SHA256 Hashes: 682877
- c2 server indicators: 70540

Concretamente, esas cantidades son las que han sido empleadas por la herramienta loki. Para poder emplear esos indicadores en Cuckoo existen dos opciones, la primera importarlos todos a la herramienta, en su formato propio de indicadores, pudiendo como alternativa importar como reglas YARA únicamente los hash, o la segunda opción, que consisten en utilizar Cuckoo en complemento con una TIP, como MISP que es la que se ha utilizado en este caso, de forma que esta gestiona tanto los IoC como los resultados generados por cuckoo, y de esta forma se pueden localizar coincidencias y gestionar los resultados. Para realizar estas conversiones se han utilizado los scripts de conversión de los anexos, que se encuentran en el repositorio "IoC-linux-malware" [6].

Esos indicadores son los que se obtuvieron de los repositorios, y ninguno presentó coincidencias con los malware analizados. A raíz de esta situación se procedió a generar los indicadores para las muestras de la tabla Tabla 5.1, en este caso si que se cargaron los IoC en MISP, CuckooSandbox y Loki.

De esta situación parten principalmente las conclusiones, y es que en la detección de malware mediante IoC, sin contar con ese indicador que genera una coincidencia, como es natural, no se puede detectar nada.

Tabla 5.1 Muestras malware para las que se han generado indicadores de compromiso.

Nombre	Hash256	Tipo
Mirai	44f5b34ef8beffac75f75ac0568a852181557d84e7753326c28ed51a87801597	Troyano
Winnti	2f1321c6cf0bc3cf955e86692bfc4ba836f5580c8b1469ce35aa250c97f0076e	Backdoor
Conti	95776f31cbcac08eb3f3e9235d07513a6d7a6bf9f1b7f3d400b2cf0afdb088a7	Ransomware
CobalStrike	6352be8aa5d8063673aa428c3807228c40505004320232a23d99ebd9ef48478a	exploit

Los IoC que se han generado para probar la capacidad de detección de malware mediante IoC que tienen las herramientas escogidas son los que contienen el hash de la muestra malware, en distintos formatos en los que se pueden escribir los IoC.

Esta decisión se ha tomado por ciertos motivos. En primer lugar, los IoC correspondientes a una muestra malware deben haber sido extraídos del análisis de la muestra malware o de un equipo infectado por ella previamente, lo cual hace complicado encontrar en los repositorios un IoC que coincida justamente con la muestra malware que se va a analizar. Por otro lado las herramientas presentan ciertas limitaciones, pues Cuckoo es un proyecto abandonado, lo que desemboca en un mal funcionamiento de ciertas capacidades de la herramienta, y Loki tiene capacidades limitadas en cuanto al uso de los IoC. De esta manera la sencillez de los hashes permite emplear con ambas herramientas los IoC sin ningún problema, y que además se pueda confirmar con total certeza si la herramienta está funcionando correctamente, pues los hashes son fáciles de extraer y comprobar para ver si realmente pertenecen a una muestra o no.

Los IoC que contienen el hash de la muestra malware son los más sencillos, pues son los que menor tiempo de vida tienen y más fáciles son de evitar; las técnicas anti-análisis hacen que las muestras cambien y a su vez también lo hace el hash, de este modo son los que menos suelen coincidir, aunque son bastante precisos y son los más empleados pues cuando coinciden no da lugar a dudas. Es por esto último que se consideran firmas los hashes. Este tipo de IoC, tal y como se ha visto en la sección 3 de la memoria son los más básicos en la "pirámide del dolor", y pertenecen al grupo de los atómicos, pues la información que contienen es indivisible y muy básica.

En la Tabla 4.2 se encuentran fuentes de las que se pueden obtener IoC en distintos formatos, los cuales se pueden utilizar con las herramientas anteriormente vistas.

Los formatos en los que se han definido los IoC son los siguientes:

- STIX - destinado a MISP
- Regla Yara - destinado a Cuckoo y Loki
- Txt - destinado a Loki y MISP

Para utilizar las reglas Yara como IoC de tipo hash es necesario utilizar el módulo para reglas Yara hash. De forma que la regla resultante sería la siguiente:

Código 5.1 Formato de regla Yara que contiene un IoC hash.

```
import "hash"

rule nombre: tag1 tag2
{
  condition:
  hash.sha256(0,filesize) == ""
}
```

Finalmente, tras cargar los IoC en las herramientas, generados a partir de las muestras comentadas en la Tabla 5.1, se ha obtenido un resultado de acierto del 100 %.

5.2 Conclusiones

5.2.1 Detección de malware mediante indicadores de compromiso

El proceso de detección de malware mediante el uso de indicadores de compromiso entra en el grupo de las técnicas de detección basadas en firmas, las cuales se desarrollaban anteriormente en el apartado 2.7.2 de la memoria. Este tipo de técnicas, tal y como se ha visto en el apartado 4, son efectivas y rápidas, siempre y cuando se disponga de los indicadores adecuados, (en el caso desarrollado son IoC o reglas Yara con el hash de la muestra malware presente en el equipo). Sin embargo, en caso de no disponer de los indicadores correctos, la herramienta no detectará la presencia de malware.

De esta manera, resalta que la obtención de los IoC sin importar como se realice, ya sea por intercambio de CTI o generándose a partir de análisis de muestras malware, es la parte más importante en las detecciones de malware basadas en firmas, pues sin el indicador adecuado no se detectará nada.

5.2.2 Indicadores de compromiso

El formato más completo para utilizar los IoC es STIX, por que no solo permite definir IoC, si no que sirve para definir todo tipo de CTI, de forma que es muy flexible y personalizable, lo cual lo hace muy valioso pues permite añadir mucha información de contexto al valor del indicador, lo que enriquece su uso. Esto se puede ver en la herramienta MISP, la cual cuenta con muchos campos para añadir información a los indicadores (identificados como atributos en MISP) y los estudios (eventos en MISP). Todo lo que se almacena en MISP se puede exportar al formato STIX, para ser posteriormente importado en una instancia MISP. Sin embargo, su aspecto más valioso es a su vez su mayor debilidad, y es que es muy complejo de utilizar, y en ocasiones en las que únicamente se quieren utilizar IoC, o en las que se tienen herramientas que no aprovechan su potencial, es ineficiente utilizarlo, como se ha visto al utilizar la herramienta Loki, que no lo soporta y únicamente necesita los IoC en texto plano.

Las reglas Yara, que también se han utilizado en el apartado 4 no son un formato para CTI o IoC como tal, si no que son un lenguaje que se utiliza para la descripción de malware, permitiendo la identificación y a su vez detección de malware, y que además de patrones permite utilizar incluir los valores de los IoC en las descripciones de malware definidas en las reglas. De esta forma para utilizar un valor de un indicador concreto para detectar malware con Yara, pues hace falta crear una regla Yara con el formato visto.

Las reglas Yaras están más orientadas al malware, mientras que STIX lo está a las amenazas informáticas de forma general, permitiendo la definición de información sobre amenazas (CTI) de sin particularizar para el malware.

Es importante recalcar que los IoC no tienen porque ser firmas como lo son el hash, que únicamente se corresponden con una muestra malware determinada, si no que se pueden dar casos en los que un mismo IoC pueda llegar a identificar observables que provienen de muestras malware similares, pero diferentes. Esto se puede dar por ejemplo, al utilizar un mismo dominio o ip para un servidor de C&C, o cuando se quiere crear una nueva muestra malware, para lo que se modifican algunas partes del código para evitar mantener el mismo hash, pero al continuar existiendo bloques de código iguales puede haber IoC de cadenas que pueden continuar identificando la muestra tras haberse modificado. Además, las firmas hacen referencia unicamente al malware, mientras que los IoC pueden hacer referencia a la red y sistemas afectados por el malware.

Estas conclusiones se pueden ver reforzadas en el artículo [44], concretamente en el apartado "3.2. Real-World IOC".

5.2.3 Herramientas

En la Tabla 5.2 se pueden ver las diferencias y la información básica más relevante de las herramientas.

Tabla 5.2 Comparación de las características de las herramientas.

	Cuckoo Sandbox	Loki	MISP
Fecha última versión	26/04/2021	4/10/2022	26/9/2022
s.o	Linux Windows Mac OS X	Linux Windows Mac OS X	Linux
Licencia	GPL-3.0	GPL-3.0	AGPL-3.0
Formatos de CTI soportados	Yara, formato txt propio	Yara, formato txt propio	STIX 1.1.1, STIX 2.0, txt, MISP
Tipo de herramienta	Análisis de Malware	Detección de Malware	Plataforma de Intercambio de información (TIP)

Las tres herramientas vistas tienen cada una un propósito diferente, siendo más o menos adecuadas según la situación y entorno. Si se agrupan las herramientas utilizadas en función del propósito para el que fueron diseñadas, se distinguen dos grupos, uno que busca la gestión de resultados y CTI, la TIP MISP, y otro que busca el estudio del malware. Dentro de las herramientas que buscan el estudio del malware, Loki está únicamente destinada a la detección de malware, mientras que Cuckoo Sandbox está enfocada al análisis de

malware, más que a su detección.

Cuckoo Sandbox es una herramienta profesional para el estudio de muestras malware, pudiendo automatizarse y personalizarse de numerosas maneras, y permitiendo emplear herramientas y motores de detección externos para generar los resultados más completos posibles. El mayor problema que presenta esta herramienta es que se encuentra sin soporte desde el año 2021, y habiendo pasado un pequeño período de abandono previo, de modo que actualmente ya presenta numerosos problemas de compatibilidad con las versiones más modernas de linux, lo cual no hará más que aumentar si continúa abandonado. Al igual que el malware continua evolucionando, las herramientas que lo estudian también deben hacerlo.

Por otro lado, Loki está más orientado al análisis de máquinas que podrían estar infectadas, tratando de detectar malware, y ofreciendo unos resultados que están más limitados que los de Cuckoo al no contar con muchas de las capacidades que Cuckoo sí que tiene. Esta herramienta no soporta ningún formato específico de IoC, está mas orientada al uso de reglas yara.

Al margen de las herramientas empleadas, apenas hay alternativas de software libre para linux, que permitan estudiar el malware mediante el uso de indicadores de compromiso.

5.3 Líneas de continuación

Para continuar el estudio sobre la detección de malware mediante IoC habría que empezar a analizar el malware para extraer información que pueda ser empleada y así poder generar y utilizar diferentes tipos de IoC. Concretamente el siguiente paso sería analizar con éxito el uso de red de un malware, y generar los IoC adecuados para emplear esa información para detectar.

En caso de querer avanzar en la detección de malware en general y querer explorar otras metodologías, el estudio de técnicas de detección heurísticas serían el siguiente paso.

Por supuesto cualquier continuación requeriría profundizar en técnias de estudio de malware para extraer información que pueda ser utilizada por los procesos de detección, ya sean métodos basados en detección heuristica o firmas, pues contar con información correcta es vital, sobre todo en la detección de malware basada en firmas, en la que se ha visto que si no se cuenta con el IoC adecuado la detección no se llegará hacer.

Otro estudio que sería interesante sería comprobar si el uso de IoC para la detección de malware es más exitoso en otros sistemas operativos como Windows, en comparación con los resultados que se han llevado para linux en este proyecto.

Apéndice A
Cuckoo Sandbox

En las siguientes secciones del apéndice aparecen los pasos que se han seguido en el proyecto para la instalación y configuración de la herramienta. Para poder visualizar al completo todas las opciones se debe consultar la documentación oficial [60]. Los archivos de configuración y scripts necesarios se encuentran en el repositorio creado para el proyecto [6].

A.1 Instalación

La instalación se ve dividida en dos equipos, el equipo principal que usualmente será un ordenador físico, y el equipo de pruebas que la herramienta empleará para realizar los análisis de manera aislada, que usualmente será un entorno virtual, aunque también podrá ser un equipo físico.

La herramienta está escrita en Python 2, y por ello se puede instalar en los sistemas Windows, Ubuntu y Mac OS X; aunque la documentación ofrecida por los creadores trata específicamente para Windows 7 y Ubuntu 16.

La instalación que se ofrece a continuación se ha empleado en un ordenador real con Ubuntu 20.04 para el equipo principal, y una máquina virtual en Virtualbox con Ubuntu Mate 18.04 para el equipo de pruebas.

A.1.1 Equipo Ubuntu principal

En primer lugar se prepara el sistema actualizando los paquetes y repositorios, e instalando algunos básicos que va a hacer falta:

```
sudo apt update
sudo apt upgrade
sudo apt install -y net-tools curl git autoconf
```

Se añade un usuario para no trabajar con el usuario personal, y como este nuevo usuario no estará en el archivo ’/etc/sudoersño tendrá permisos de administrador:

```
sudo adduser cuckoo
cd /home/cuckoo/
```

Se instala Python 2.7 y pip2:

```
curl https://bootstrap.pypa.io/pip/2.7/get-pip.py -o get-pip.py
sudo apt install python2
```

```
sudo python2 get-pip.py
```

Se instalan las dependencias de python necesarias:

```
sudo apt-get install -y python-dev libffi-dev libssl-dev libfuzzy-dev libtool
    flex libjansson-dev
sudo apt-get install -y python-setuptools
sudo apt-get install -y libjpeg-dev zlib1g-dev swig systemtap
```

Se instala mongodb:

```
sudo apt-get install -y mongodb
```

Se instala PostgreSQL:

```
sudo apt-get install -y postgresql libpq-dev
```

Se instala el entorno de virtualización VirtualBox:

```
  curl https://download.virtualbox.org/virtualbox/5.2.44/VirtualBox
      -5.2.44-139111-Linux_amd64.run --output virtualbox_5.2.44.run
  chmod +x ./virtualbox_5.2.44.run
  ./virtualbox_5.2.44.run
  sudo apt install virtualbox-dkms
```

Se añade al usuario cuckoo al grupo de virtualbox, y será desde el usuario cuckoo desde el que habrá que configurar y utilizar virtualbox:

```
sudo usermod -a -G vboxusers cuckoo
```

Se instala tcpdump:

```
sudo apt-get install -y libcap2-bin tcpdump apparmor-utils
sudo aa-disable /usr/sbin/tcpdump
```

Se crea un grupo de captura, añadimos al grupo al usuario cuckoo, y hacemos dueños de tcpdump a ese grupo

```
sudo groupadd pcap
sudo usermod -a -G pcap cuckoo
sudo chgrp pcap /usr/sbin/tcpdump

sudo setcap cap_net_raw,cap_net_admin=eip /usr/sbin/tcpdump
getcap /usr/sbin/tcpdump #Para verificar los resultados del comando anterior
```

Se instalan librerías de interés general para la herramienta:

```
sudo -H pip install distorm3==3.4.4
```

```
sudo -H pip install yara-python==3.6.3
sudo apt-get install -y ssdeep
ssdeep -V
sudo -H pip install pydeep
pip show pydeep
sudo -H pip install openpyxl
sudo -H pip install ujson
sudo -H pip install jupyter
```

Se instala la librería M2Crypto:

```
sudo apt-get install -y swig
sudo pip install m2crypto==0.24.0
```

De manera adicional, en caso de querer realizar análisis forense de la memoria, es necesario el módulo opcional Volatility:

```
git clone https://github.com/volatilityfoundation/volatility.git
cd volatility
sudo python setup.py build
sudo python setup.py install
cd ..
```

Por último, una vez se han terminado de instalar los requisitos necesarios para Cuckoo y para los módulos se procede a instalar Cuckoo:

```
sudo -H pip install -U pip setuptools
sudo -H pip install -U cuckoo
```

A.1.2 Ubuntu Guest

Antes de nada es necesario crear la máquina virtual e instalar el sistema operativo. Se debe colocar al comienzo un adaptador de red en NAT para tener acceso a internet y poder instalar y añadir librerías y programas sin problemas.

En primer lugar descargamos el script "agent.py" que podemos encontrar en la carpeta "/home/cuckoo/.cuckoo/agent/".

Se prepara el sistema:

```
sudo apt update
sudo apt upgrade -y
sudo apt install curl
```

Se instala Python 2.7 y pip2.

```
curl https://bootstrap.pypa.io/pip/2.7/get-pip.py -o get-pip.py
sudo apt install python2
sudo python2 get-pip.py
```

Para utilizar python y pip los comandos serán python2 y pip2.

Se configura el sistema para que al arrancar el agente se inicie automáticamente. Es necesario añadir rutas absolutas.

```
whereis python2
sudo crontab -e
```

La línea que habrá que añadir será como esta:

```
@reboot /usr/bin/python2.7 /path/to/agent.py
```

Se instalan dependencias necesarias:

```
sudo apt-get install systemtap gcc patch linux-headers- $(uname -r)
```

Se instalan símbolos de debuggeo:

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys C8CAB6595FDFF622

codename=$(lsb_release -cs)
sudo nano /etc/apt/sources.list.d/ddebs.list

deb http://ddebs.ubuntu.com/ ${codename} main restricted universe multiverse
deb http://ddebs.ubuntu.com/ ${codename}-updates main restricted universe
    multiverse
deb http://ddebs.ubuntu.com/ ${codename}-proposed main restricted universe
    multiverse

sudo apt-get update
sudo apt-get install linux-image-$(uname -r)-dbgsym

wget https://raw.githubusercontent.com/cuckoosandbox/cuckoo/master/stuff/
    systemtap/expand_execve_envp.patch
wget https://raw.githubusercontent.com/cuckoosandbox/cuckoo/master/stuff/
    systemtap/escape_delimiters.patch
sudo patch /usr/share/systemtap/tapset/linux/sysc_execve.stp < expand_execve_
    envp.patch
sudo patch /usr/share/systemtap/tapset/uconversions.stp < escape_delimiters.
    patch
```

Se compila la extensión del kernel:

```
wget https://raw.githubusercontent.com/cuckoosandbox/cuckoo/master/stuff/
    systemtap/strace.stp
sudo apt-get remove systemtap
sudo apt-get install gcc g++ elfutils libdw-dev elfutils build-essential
#Descargamos la más reciente https://sourceware.org/systemtap/wiki/
    SystemTapReleases (4.7 yo)
tar xf systemtap-x.x.tar.gz
cd systemtap-*
./configure
sudo make
sudo make install
```

```
cd ../
sudo stap -p4 -r $\mbox{\$}$(uname -r) strace.stp -m stap_ -v
```

Se testa que la extensión se ha instalado correctamente:

```
sudo staprun -v ./stap_.ko
```

La salida deberá tener una estructura similar a:

```
staprun:insert_module:x Module stap_ inserted from file path_to_stap_.ko
```

Se coloca en su sitio la extensión:

```
sudo mkdir /root/.cuckoo
sudo mv stap_.ko /root/.cuckoo/
```

Se desactiva el cortafuegos de la máquina virtual

```
sudo ufw disable
```

Se desactiva NTP:

```
sudo timedatectl set-ntp off
```

Se desactiva software pre-instalado:

```
sudo apt-get purge update-notifier update-manager update-manager-core ubuntu-
    release-upgrader-core
sudo apt-get purge whoopsie ntpdate cups-daemon avahi-autoipd avahi-daemon
    avahi-utils
sudo apt-get purge account-plugin-salut libnss-mdns telepathy-salut
sudo apt autoremove
```

Por último, una vez que se haya terminado de instalar los requisitos, se debe apagar la máquina virtual, y cambiar el adaptador de red para que sea solo anfitrión y que pertenezca a la red "vboxnet0" que hemos creado.

Después hay que encender la máquina y configurar los parámetros de red en el sistema operativo

```
IP = 192.168.56.101
Subnet = 255.255.255.0
Gateway = 192.168.56.1
DNS = 8.8.8.8 / 8.8.4.4
```

Finalmente, con la máquina activa y mientras esté ejecutándose el agente, hay realizar una snapshot que se debe llamar como hemos indicado en la configuración de Cuckoo (Final), quedando todo listo para que la máquina virtual pueda ser utilizada por Cuckoo.

A.2 Configuración

Una vez que se ha instalado Cuckoo, es necesario proceder a configurarlo. Primero se va a configurar VirtualBox y después Cuckoo.

A.2.1 VirtualBox

En primer lugar hay que configurar Virtualbox, lo cual hay que hacer desde el usuario Cuckoo, desde la interfaz gráfica, o desde la terminal. Hay que añadir una red interna con nombre "vboxnet0", y con la red 192.168.56.0/24.

```
su cuckoo
vboxmanage hostonlyif create
vboxmanage hostonlyif ipconfig vboxnet0 --ip 192.168.56.1
VBoxManage modifyvm cuckoo1 --nested-hw-virt on
exit

sudo mkdir /opt/systemd/
sudo nano /opt/systemd/vboxhostonly
# !/bin/bash
# hostonlyif create
# vboxmanage hostonlyif ipconfig vboxnet0 --ip 192.168.56.1

sudo chmod a+x /opt/systemd/vboxhostonly

sudo touch /etc/systemd/system/vboxhostonlynic.service
sudo nano /etc/systemd/system/vboxhostonlynic.service
# Description=Setup VirtualBox Hostonly Adapter
# After=vboxdrv.service
#
# [Service]
# Type=oneshot
# ExecStart=/opt/systemd/vboxhostonly
#
# [Install]
# WantedBy=multi-user.target

systemctl daemon-reload
systemctl enable vboxhostonlynic.service
```

Se configuran las reglas del cortafuegos para que no haya internet en la máquina virtual y el host no reenvíe tráfico:

```
sudo apt-get install -y iptables-persistent
sudo iptables -P FORWARD DROP
iptables-save > /etc/iptables/rules.v4
```

A.2.2 Cuckoo

Para configurar Cuckoo se van editar los ficheros de configuración que encontramos dentro del directorio de trabajo de Cuckoo, en la carpeta "conf" (en nuestro caso se encuentra en "/home/cuckoo/.cuckoo/conf/").

En el archivo "**cuckoo.conf**" se pueden configurar los aspectos generales del sandbox, y temas relacionados con los análisis. Concretamente deberán estar estos valores:

```
machinery = virtualbox #por defecto
memory_dump = no #por defecto
ip = 192.168.56.1 #por defecto
max_vmstartup_count = 2
```

En caso de querer utilizar las reglas Yara es necesairo activar el volcado de memoria 'memory_dump = yes'.

En el archivo "**auxiliary.conf**" se pueden configurar las diferentes opciones que tienen los scripts que se emplean para analizar el malware:

```
sniffer enabled = yes #por defecto
```

En el archivo "**virtualbox.conf**" se puede configurar la forma de interactuar con el software de virtualización escogido:

```
virtualbox mode = gui
machines = cuckoo1 #por defecto
platform = linux
ip = 192.168.56.101 #por defecto
snapshot = Final
```

En el archivo "**memory.conf**":

```
delete_memdump = yes
```

En el archivo "**reporting.conf**":

```
feedback enabled = no #por defecto
html = yes
mongodb enabled = yes

[misp]
enabled = yes
url = https://192.168.1.146/
```

En el archivo "**processing.conf**":

```
[misp]
enabled = yes
url = https://192.168.1.146/
apikey = WtSH3J2GBfcM8IQEYcIYathrylggnMADUYYxIZQS
```

Apéndice B
Loki

En este apéndice se comentan la instalación y el uso de ciertas utilidades de la herramienta [56, 59]. Los archivos de configuración y scripts necesarios se encuentran en el repositorio creado para el proyecto [6].

La herramienta está basada en python 3. Para instalarla será necesario clonar el repositorio e instalar las dependencias necesarias. Es importante recalcar que existe otro programa diferente llamado Loki que se puede instalar a través del sistema de gestión de paquetes de Linux APT.

```
sudo apt update

#Instalar python 3
sudo apt install -y python3 python3-pip libssl-dev
#Instalar las librerias de python
sudo pip install yara-python pylzma psutil netaddr colorama pymisp future
sudo pip install rfc5424-logging-handler

#Intalar Loki
sudo apt install git
git clone https://github.com/Neo23x0/Loki.git
cd ./Loki/
python3 ./loki-upgrader.py
git clone https://github.com/Neo23x0/signature-base.git
```

Para que el programa pueda escanear todos los directorios deberá tener acceso a ellos, de modo que en caso de querer realizar un escaneo completo del sistema será necesario ejecutar el programa con root, para lo que además será necesario instalar las librerías de python con el usuario root para que tenga acceso a ellas.

Apéndice C
MISP

En este apéndice se comentan la instalación y configuración de la herramienta [55, 61, 53]. En el proyecto no se han empleado ninguna de las utilidades adicionales que incluyen los módulos, pero se incluyen como material adicional. Los archivos de configuración y scripts necesarios se encuentran en el repositorio creado para el proyecto [6].

C.1 Instalación

La instalación se realiza a partir del repositorio de Github en el que se encuentra el código fuente. La instalación se lleva a cabo a través del script que aporta el proyecto.

```
sudo apt update
sudo apt upgrade
sudo apt install -y git

git clone https://github.com/MISP/MISP.git
cd MISP/INSTALL/
./INSTALL.sh -c -D
```

Una vez instalada la plataforma, se podrá acceder a ella a través de la interfaz web que ofrece. Las credenciales por defecto que son necesario emplear la primera vez que se accede son las siguientes:

```
usuario: admin@admin.test
contraseña: admin
```

Tras el primer inicio de sesión se solicitará una nueva contraseña.

C.1.1 Módulos

La herramienta cuenta con módulos extra que permiten aumentar las capacidades de la herramienta. Estos módulos se instalan, activan y configuran de manera independiente a la herramienta, de forma que no alteran al núcleo y utilidades principales.

La instalación se realiza mediante los siguientes comandos:

```
  pip install --upgrade pip
```

```
sudo apt-get install python3-dev python3-pip libpq5 libjpeg-dev tesseract-ocr
     libpoppler-cpp-dev imagemagick virtualenv libopencv-dev zbar-tools
    libzbar0 libzbar-dev libfuzzy-dev build-essential -y
sudo -u www-data virtualenv -p python3 /var/www/MISP/venv
cd /usr/local/src/
sudo chown -R www-data: .
sudo -u www-data git clone https://github.com/MISP/misp-modules.git
cd misp-modules
sudo -u www-data /var/www/MISP/venv/bin/pip install -I -r REQUIREMENTS
sudo -u www-data /var/www/MISP/venv/bin/pip install .
```

Además, para configurarlos y arrancarlos como un servicio:

```
sudo cp etc/systemd/system/misp-modules.service /etc/systemd/system/
sudo systemctl daemon-reload
sudo systemctl enable --now misp-modules
sudo service misp-modules start #or
/var/www/MISP/venv/bin/misp-modules -l 127.0.0.1 & #to start the modules
```

C.2 Configuración

Una vez realizada la instalación se puede proceder a configurar la herramienta, todo a través de la interfaz web.

C.2.1 Conexión con herramientas

Para que otras herramientas puedan comunicarse con la plataforma e intercambiar información deberán tener acceso red entre ellas, así como configurar en la herramienta externa el acceso a la instancia MISP con la clave de la API.

La clave API pertenece a un usuario de los que existan en la plataforma. Se obtiene en el menú "Event Actions>Automation", en el que habrá que pulsar donde indica para obtenerla. Concretamente la url donde se podrá obtener será "https://ip/users/view/me".

C.2.2 Módulos

Para poder utilizar los módulos es necesario activarlos previamente, para ello hay que acceder a la url "https://127.0.0.1/servers/serverSettings/Plugin", donde aparecerán los módulos agrupados en secciones: importación, exportación y expansión; para poder utilizarlos es necesario activar la sección completa a la que pertenecen. De esta manera para poder importar hay que ir a la sección "Import" y cambiar "Plugin.Import_-services_enable" a true. La ip y puerto de acceso a la instalación de los módulos misp se podrá modificar, pero lo normal es activar la que viene por defecto, pues se suele instalar en la misma máquina que MISP.

Una vez activados los módulos, y con conexión existente entre la instalación de MISP y de los módulos, aparecerán las opciones de configuración de los módulos pertenecientes a los grupos activos, pudiendo así activarse y configurar aquellos módulos que lo requieran.

Puede confirmarse que la conexión es correcta en "https://127.0.0.1/servers/serverSettings/diagnostics" en la sección "Module System".

Apéndice D
Scripts conversión

Los scripts de conversión que se encuentran en el anexo está en el repositorio [6].

Código D.1 Script de conversión hash de txt a formato loki.

```
  with open("ioc_txt.txt") as f:
    with open("ioc_loki.txt", "w") as g:
      for line in f:
        g.write(line[:-1] + ";\n")
```

Código D.2 Script de conversión hash de loki a formato yara.

```
ioc_txt = "ioc_txt.txt"

f = open(ioc_txt,"r")
for line in f:

  hash = line[:-1]

  # Deletes the ';' from the hash read from the list of IoC using the loki
      format
  hash = hash[:-1]

  g = open(str(hash)+".yara","w")
  g.write("""import "hash"

  rule """ + str(hash) + """
  {
    condition:
    hash.sha256(0,filesize) == """ + '"' + str(hash) + '"' + """

  }""")
  g.close()

f.close()
```

Código D.3 Script de conversión hash de loki a formato misp en stix.

```
ioc_txt = "ioc_txt.txt"
ioc_stix = "ioc_stix.xml"
timestamp = "2022-07-20T19:52:13.854506+00:00"

f = open(ioc_txt,"r")
g = open(ioc_stix,"w")

g.write("""<stix:STIX_Package
xmlns:FileObj="http://cybox.mitre.org/objects#FileObject-2"
xmlns:cybox="http://cybox.mitre.org/cybox-2"
xmlns:cyboxCommon="http://cybox.mitre.org/common-2"
xmlns:cyboxVocabs="http://cybox.mitre.org/default_vocabularies-2"
xmlns:example="http://example.com"
xmlns:indicator="http://stix.mitre.org/Indicator-2"
xmlns:stix="http://stix.mitre.org/stix-1"
xmlns:stixCommon="http://stix.mitre.org/common-1"
xmlns:stixVocabs="http://stix.mitre.org/default_vocabularies-1"
xmlns:ttp="http://stix.mitre.org/TTP-1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" id="ioc:{id}" version
    ="1.2">
<stix:STIX_Header>
<stix:Title>File Hash Reputation Service Results</stix:Title>
  <stix:Package_Intent xsi:type="stixVocabs:PackageIntentVocab-1.0">Indicators
      - Malware Artifacts</stix:Package_Intent>
</stix:STIX_Header>
<stix:Indicators>""".format(id=ioc_stix))

for line in f:

  # Deletes the '\n'
  hash = line[:-1]

  # Deletes the ';' from the hash read from the list of IoC using the loki
      format
  hash = hash[:-1]

  if len(hash)==64 :
    tipo_hash = "SHA256"
  if len(hash)==40 :
    tipo_hash = "SHA1"
  if len(hash)==128 :
    tipo_hash = "SHA512"
  if len(hash)==56 :
    tipo_hash = "SHA224"
  if len(hash)==96 :
    tipo_hash = "SHA384"

  g.write("""
      <stix:Indicator id="ioc:{valor_hash}" timestamp="{time}" xsi:type='
          indicator:IndicatorType'>
        <indicator:Title>File Reputation for {tipo_hash}={valor_hash}</
            indicator:Title>
        <indicator:Type xsi:type="stixVocabs:IndicatorTypeVocab-1.1">File Hash
            Watchlist</indicator:Type>
        <indicator:Observable id="ioc:{valor_hash}">
```

```
          <cybox:Object id="ioc:{valor_hash}">
            <cybox:Properties xsi:type="FileObj:FileObjectType">
              <FileObj:Hashes>
                <cyboxCommon:Hash>
                  <cyboxCommon:Type condition="Equals" xsi:type="cyboxVocabs:
                      HashNameVocab-1.0">{tipo_hash}</cyboxCommon:Type>
                  <cyboxCommon:Simple_Hash_Value condition="Equals">{valor_hash
                      }</cyboxCommon:Simple_Hash_Value>
                </cyboxCommon:Hash>
              </FileObj:Hashes>
            </cybox:Properties>
          </cybox:Object>
        </indicator:Observable>
        <indicator:Indicated_TTP>
          <stixCommon:TTP id="ioc:{valor_hash}" timestamp="{time}" xsi:type='ttp
              :TTPType'>
            <ttp:Title>Malicious file</ttp:Title>
          </stixCommon:TTP>
        </indicator:Indicated_TTP>
        <indicator:Confidence timestamp="{time}">
          <stixCommon:Value vocab_reference="https://en.wikipedia.org/wiki/
              Percentage" vocab_name="Percentage">75</stixCommon:Value>
        </indicator:Confidence>
      </stix:Indicator>""".format(time=timestamp,valor_hash=hash,tipo_hash=
          tipo_hash))

g.write("""
  </stix:Indicators>
</stix:STIX_Package>""")

g.close()
f.close()
```

Bibliografía

[1] Check Point Software. Informe de seguridad del 2022 de check point software, 2022. URL *https://pages.checkpoint.com/cyber-security-report-2022-spanish.html?utm_source=eblast&utm_medium=email&utm_campaign=fm_eb_22q1_latam_es_security_report*. Accedida en 23/08/2022.

[2] Mark Stamp, Mamoun Alazab, and Andrii Shalaginov. *Malware Analysis Using Artificial Intelligence and Deep Learning*. Springer, 1 edition, 2021.

[3] Virustotal stats. URL *https://www.virustotal.com/gui/stats*. Accedida en 14/03/2022.

[4] AV-ATLAS. Repositorio ioc-linux-malware. URL *https://portal.av-atlas.org/malware/statistics*. Accedida en 12/03/2023.

[5] Rami Sihwail, Omar Khairuddin, and Khairul Akram Zainol Ariffin. A survey on malware analysis techniques: Static, dynamic, hybrid and memory analisis. *International Journal on Advanced Science Engineering and Information Technology*, 9 2018.

[6] Fº Javier Ros Raposo. Repositorio ioc-linux-malware. URL *https://github.com/frarosrap/IoC-linux-malware*. Accedida en 20/02/2023.

[7] Kapersky. ¿qué es la ciberseguridad? URL *https://latam.kaspersky.com/resource-center/definitions/what-is-cyber-security*. Accedida en 24/08/2022.

[8] Yuchong Li and Qinghui Liu. A comprehensive review study of cyber-attacks and cyber security; emerging trends and recent developments. *Energy Reports*, 7:8176–8186, 11 2021. ISSN 23524847. doi:10.1016/j.egyr.2021.08.126.

[9] Fortinet. Types of cyber attacks. URL *https://www.fortinet.com/resources/cyberglossary/types-of-cyber-attacks*. Accedida en 23/08/2022.

[10] Glosario malwarebytes. URL *https://blog.malwarebytes.com/glossary/*. Accedida en 13/03/2022.

[11] Monnappa K A. *Learning Malware Analysis*. Packt Publishing Ltd, 1 edition, 2018. ISBN 9781788392501.

[12] Babak Bashari Rad, Suhaimi Ibrahim, and Maslin Masrom. Camouflage in malware: from encryption to metamorphism, 2012. URL *https://www.researchgate.net/publication/235641122*.

[13] Rabia Tahir. A study on malware and malware detection techniques. *International Journal of Education and Management Engineering*, 8:20–30, 3 2018. ISSN 23053623. doi:10.5815/ijeme.2018.02.03. URL *http://www.mecs-press.org/ijeme/ijeme-v8-n2/v8n2-3.html*.

[14] Ilsun You and Kangbin Yim. Malware obfuscation techniques: A brief survey. pages 297–300. 2010. ISBN 9780769542362. doi:10.1109/BWCCA.2010.85.

[15] Jhon Aycock. *Computer Viruses and Malware*. Springer, 2006. ISBN 0387258868.

[16] Vesselin Bontchev. Current status of the caro malware naming scheme. URL *https://bontchev.nlcv.bas.bg/papers/naming.html*.

[17] Victor Marak. *Windows Malware analysis essentials : master the fundamentals of malware analysis for the Windows platform and enhance your anti-malware skill set*. ISBN 9781785281518.

[18] Microsoft malware naming, 2 2022. URL *https://docs.microsoft.com/en-us/windows/security/threat-protection/intelligence/malware-naming*. Accedida en 14/03/2022.

[19] Trend micro malware naming, 12 2019. URL *https://success.trendmicro.com/dcx/s/solution/1119738-new-threat-detection-naming-scheme-in-trend-micro?language=en_US*. Accedida en 14/03/2022.

[20] Penny Chase, Martin Robert, Ivan Kirillov, and Desiree Beck. Malware attribute enumeration and characterization article. *ResearchGate*, 1 2015.

[21] Jimmy Kuo and Desiree Beck. The common malware enumeration (cme) initiative, 9 2005. URL *https://www.virusbulletin.com/virusbulletin/2005/09/common-malware-enumeration-cme-initiative/*. Accedida en 15/03/2022.

[22] Maec github project. URL *https://maecproject.github.io/about-maec/*. Accedida en 15/03/2022.

[23] Inteligencia contra amenazas i. URL *https://www.sothis.tech/inteligencia-de-amenazas-i/#*. Accedida en 22/03/2022.

[24] Eric W. Burger, Michael D. Goodman, Panos Kampanakis, and Kevin A. Zhu. Taxonomy model for cyber threat intelligence information exchange technologies. volume 2014-November, pages 51–60. Association for Computing Machinery, 11 2014. ISBN 9781450331517. ISSN 15437221. doi: 10.1145/2663876.2663883.

[25] Yansi Keim and A. K. Mohapatra. Cyber threat intelligence framework using advanced malware forensics. *Springer*, 2 2019. URL *https://link.springer.com/article/10.1007/s41870-019-00280-3*.

[26] Vasileios Mavroeidis and Siri Bromander. Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence. volume 2017-January, pages 91–98. Institute of Electrical and Electronics Engineers Inc., 12 2017. ISBN 9781538623855. doi: 10.1109/EISIC.2017.20.

[27] Wiem Tounsi and Helmi Rais. A survey on technical threat intelligence in the age of sophisticated cyber attacks, 1 2018. doi:10.1016/j.cose.2017.09.001.

[28] Sol González. Qué es cyber threat intelligence, 11 2021. URL *https://www.welivesecurity.com/la-es/2021/11/08/que-es-cyber-threat-intelligence/*. Accedida en 02/09/2022.

[29] Anomali. Que es una tip. URL *https://www.anomali.com/es/resources/what-is-a-tip#:~:text=Tratamiento%2C%20normalizaci%C3%B3n%2C%20enriquecimiento%20y%20calificaci%C3%B3n,intercambio%20de%20inteligencia%20contra%20amenazas*. Accedida en 21/03/2022.

[30] Inc Cisco Systems. Cisco cyber threat intelligence. URL *CiscoSystems,Inc*. Accedida en 21/03/2022.

[31] Open cti introduction. URL *https://luatix.notion.site/Introduction-f99633ba66ba4ee3af1a4d832208dc99*. Accedida en 24/03/2022.

[32] Yeti documentation. URL *https://yeti-platform.readthedocs.io/en/latest/*. Accedida en 24/03/2022.

[33] Taxii documentation. URL *https://oasis-open.github.io/cti-documentation/taxii/intro*. Accedida en 28/03/2022.

[34] Cybox documentation. URL *http://cyboxproject.github.io/*. Accedida en 28/03/2022.

[35] guía de seguridad ccn-stic-423 indicadores de compromiso (ioc).

[36] Mossé Cyber Security Institute. Yara: A powerful malware analysis tool for detecting ioc's - part 1, 5 2022. URL *https://library.mosse-institute.com/articles/2022/05/yara-a-powerful-malware-analysis-tool-for-detecting-ioc-s-part-1/yara-a-powerful-malware-analysis-tool-for-detecting-ioc-s-part-1.html#yara-a-powerful-malware-analysis-tool-for-detecting-iocs-part-1*. Accedida en 19/10/2022.

[37] Omer Aslan and Refik Samet. A comprehensive review on malware detection approaches, 2020. doi:10.1109/ACCESS.2019.2963724.

[38] R Salazar-Hernández and J Díaz-Verdejo. Generación de tráfico de ataque para la evaluación de sistemas de detección de intrusos, 2009. URL *www.securityfocus.com*.

[39] Jorge Luis Diaz martinez. Estudio comparativo de metodologías de selección de caracteristicas en sistemas de detección de intrusos (ids), basado en anomalias de red, 2016.

[40] Deris Stiawan, Abdul Hanan Abdullah, and Mohd Yazid Idris. The trends of intrusion prevention system network. *ICETC 2010 - 2010 2nd International Conference on Education Technology and Computer*, 4, 2010. doi:10.1109/ICETC.2010.5529697.

[41] Nasreddine Bencherchali. Malware analysis — tools and resources, 9 2019. URL *https://nasbench.medium.com/malware-analysis-tools-and-resources-16eb17666886*. Accedida en 15/04/2022.

[42] INCIBE. El valor de los indicadores de compromiso en la industria, 3 2018. URL *https://www.incibe-cert.es/blog/el-valor-los-indicadores-compromiso-industria*. Accedida en 21/03/2022.

[43] Facundo Muñoz. Qué son los indicadores de compromiso, 2 2021. URL *https://www.welivesecurity.com/la-es/2021/02/22/que-son-indicadores-compromiso-evidencia-puedes-haber-sido-victima-malware/*. Accedida en 17/03/2022.

[44] Antonio Villalón-Huerta, Ismael Ripoll-Ripoll, and Hector Marco-Gisbert. Key requirements for the detection and sharing of behavioral indicators of compromise. *Electronics (Switzerland)*, 11, 2 2022. ISSN 20799292. doi:10.3390/electronics11030416.

[45] David Bianco. A framework for cyber threat hunting part 1: The pyramid of pain, 2015. URL *http://detect-respond.blogspot.com/2013/03/the-*.

[46] Veronica Drake. What is the pyramid of pain?, 7 2022. URL *https://flashpoint.io/blog/the-pyramid-of-pain-and-cyber-threat-intelligence/*. Accedida en 01/09/2022.

[47] Iker Sala. Indicadores de compromiso en la gestión de riesgos, 7 2018. URL *https://www.audea.com/indicadores-compromiso-la-gestion-riesgos/*. Accedida en 21/03/2022.

[48] Openioc github project. URL *https://github.com/fireeye/OpenIOC_1.1*. Accedida en 22/03/2022.

[49] Lauren Rudman and Barry Irwin. Dridex: Analysis of the traffic and automatic generation of iocs. pages 77–84. Institute of Electrical and Electronics Engineers Inc., 12 2016. ISBN 9781509024735. doi:10.1109/ISSA.2016.7802932.

[50] Institute of Electrical and Electronics Engineers. *2018 IEEE International Symposium on Technologies for Homeland Security (HST) : Crowne Plaza Boston - Woburn15 Middlesex Canal Park Road, Woburn. Massachusetts 01801 United States : 23-24 October 2018.* 2018. ISBN 9781538634431.

[51] ¿qué son los indicadores de ataque (ioa)? diferencias con los ioc. URL *https://ciberseguridad.com/guias/prevencion-proteccion/indicadores-ataque-ioa/*. Accedida en 27/12/2022.

[52] crowdstrike. indicators-of-compromise, 10 2022. URL *https://www.crowdstrike.com/cybersecurity-101/indicators-of-compromise/ioa-vs-ioc/*. Accedida en 27/12/2022.

[53] MISP Project. Misp documentation. URL *https://www.misp-project.org/documentation/*. Accedida en 04/10/2022.

[54] Davy Preuveneers and Wouter Joosen. Sharing machine learning models as indicators of compromise for cyber threat intelligence. *Journal of Cybersecurity and Privacy*, 1:140–163, 2 2021. doi:10.3390/jcp1010008.

[55] Misp - open source threat intelligence platform. URL *MISP-OpenSourceThreatIntelligencePlatform*. Accedida en 28/03/2022.

[56] Neo23x0. Loki github project, 2 2022. URL *https://github.com/Neo23x0/Loki*. Accedida en 04/10/2022.

[57] MISP Project. Misp-stix-converter - mapping documentation. URL *https://misp.github.io/misp-stix/documentation/#Attributes-to-STIX-20-mapping*. Accedida en 19/10/2022.

[58] Misp github project. URL *https://github.com/MISP/misp*. Accedida en 22/03/2022.

[59] Neo23x0. signature-base github project, 2 2022. URL *https://github.com/Neo23x0/signature-base*. Accedida en 04/10/2022.

[60] Claudio Guarnieri. Cuckoo sandbox book. URL *https://cuckoo.readthedocs.io/en/latest/*. Accedida en 04/10/2022.

[61] Introducción misp, 5 2018. URL *https://fwhibbit.es/misp-introduccion-e-instalacion*. Accedida en 16/03/2022.