G. Fernandez, and A. Nieto, "Configuración de honeypots adaptativos para análisis de malware", *III Jornadas Nacionales de Investigación en Ciberseguridad (JNIC 2017)*, pp. 91-98, 2017.
NICS Lab. Publications: https://www.nics.uma.es/publications

Configuración de honeypots adaptativos para análisis de malware

Gerardo Fernandez y Ana Nieto

Network, Information and Computer Security (NICS) Lab
Lenguajes y Ciencias de la Computación
Universidad de Málaga, España
Email:{gerardo,nieto}@lcc.uma.es

***Abstract*—Este trabajo propone una arquitectura de despliegue de honeypots adaptativos, configurados dinámicamente a partir de los requisitos del malware que intenta infectar los servicios trampa. A diferencia de otros trabajos sobre honeypots adaptativos, los mecanismos de adaptabilidad aquí diseñados tomarán como base información de inteligencia sobre amenazas actuales, indicadores de compromiso (IOCs) conocidos, así como información de actividades sospechosas actualmente en estudio por los analistas. Este conocimiento será empleado para configurar honeypots de manera dinámica, permitiendo satisfacer los requisitos necesarios para que el malware pueda desplegar toda su operativa.**

***Index Terms*—Honeypot, malware, adaptativo, dinamico, inteligencia, IOCs.**

Tipo de contribución: *Investigación original (límite 8 páginas)*

I. Introducción

De acuerdo al último informe emitido por CCN-CERT, las incidencias por ataques dirigidos contra empresas, organizaciones y usuarios aumentó un 40% [1]. Este amplio abanico de afectados tiene su razón de ser en el empleo de técnicas cada vez más sofisticadas para afectar a la seguridad de los sistemas. En concreto, una de las mayores preocupaciones hoy día es cómo responder de manera efectiva a las campañas de difusión de malware destinadas a obtener información reservada, destruir las comunicaciones y servicios, o bien secuestrar los datos. Mientras que frenar los ataques, una vez conocidos los vectores de infección, es relativamente sencillo, los ataques *zero day* no son nada fáciles de detectar y frenar. Más aún, no podremos detener nuevos ataques sin llegar a comprender, en su totalidad, cómo opera el malware que intenta infectar un sistema.

Los honeypots son sistemas diseñados para capturar ataques mediante simulación de servicios y/o aplicaciones reales. Emplean técnicas de engaño que procuran satisfacer la demanda del atacante, dando respuesta válida a peticiones de servicio y aceptando las modificaciones que pretende realizar en el equipo *objeto del ataque*. Hay dos escenarios de uso dependiendo de los objetivos perseguidos:

- *Reproducir servicios del entorno de producción*: mostrando una huella similar a la de los servicios ofrecidos en el entorno productivo. El objetivo en este caso es el de distraer al atacante, mientras se recogen evidencias de las técnicas empleadas para poder fortalecer los servicios del entorno productivo ante ataques similares.
- *Entornos de investigación*: mostrando una configuración de honeypots cuyo objetivo es el de analizar nuevas técnicas de ataque y prevenir su difusión mediante la creación de reglas para los sistemas de contramedida.

Mientras que el primer escenario requiere medidas estáticas destinadas a proteger a una infraestructura concreta, el segundo escenario precisa de sistemas capaces de obtener del atacante la máxima información posible para poder ser analizada. Este trabajo se centra en el segundo escenario, concretamente en el diseño de sistemas de captura adaptativos que puedan responder a ataques realizados de manera automatizada.

El principal problema al que se enfrentan este tipo de soluciones radica en la carencia de información previa al ataque. Es decir, o bien se consideran escenarios concretos de estudio (servidores web en MS Windows, ataques a protocolos SSH/TELNET, etc.) o, en otro caso, se estudian familias de malware y *se diseñan* honeypots a su medida (p.ej. Mirai) [2]. En ambos casos se perderá la capacidad de disparar nuevas fases del malware que no esté dirigido hacia estos escenarios preconcebidos.

Con este fin se propone una plataforma que permite capturar evidencias y adquirir conocimiento sobre malware nuevo en fases tempranas de su lanzamiento al público, empleando para ello sistemas de distribución e intercambio de información de inteligencia sobre malware. Estos sistemas están diseñados para distribuir direcciones IP comprometidas (para los sistemas de filtrado), o bien servir de plataforma para el intercambio de información sobre las características y el funcionamiento del malware. El objetivo de este artículo es integrar este tipo de servicios para desplegar honeypots diseñados según las características del malware que los atacan.

Este artículo se estructura como sigue. La sección II detalla los trabajos relacionados. La sección III describe los componentes de la arquitectura propuesta, cuya interacción se analiza en base a dos casos de uso en la sección IV. En la sección V se argumenta sobre la viabilidad de implementación y despliegue de la solución propuesta. Finalmente, se presentan las conclusiones y el trabajo futuro.

II. Trabajos relacionados

Se han realizado trabajos con anterioridad con la intención de adaptar los servicios ofrecidos en un honeypot a la demanda de servicios realizadas por el atacante [3], [4], [5]. Honeytrap modela un servicio de interceptación de conexiones

capaz de levantar servicios de acuerdo a una configuración preestablecida. Honeyweb es capaz de emular servidores Apache, Microsoft IIS e incluso Netscape. La decisión de qué versión se ofrece se toma tras analizar la URL que llega en la petición de conexión, de manera que se configuren las cabeceras HTTP acorde a las necesidades del atacante.

Por otro lado, se han realizado implementaciones de servicios trampa para el protocolo TELNET que se adaptan para ofrecer hasta 8 arquitecturas de procesador [6]. La decisión de qué arquitectura emplear dependerá del tipo de comando lanzado por el atacante. Se han hecho trabajos similares con SSH [7] [8] cuyos mecanismos de adaptabilidad se centran en la interacción con el atacante a través de la sesión SSH establecida. SIPHON [9] enfoca la construcción de honeypots empleando dispositivos físicos interconectados a través de wormholes que redirigen los ataques recibidos hacia los dispositivos trampa.

Con un ámbito más amplio de aplicación, el trabajo recogido en [10] muestra un sistema de gestión dinámica de honeypots de alta y baja interacción, desplegados de manera virtualizada en función de la evaluación del motor de decisión de honeybrid. La toma de decisión, en este caso, está centrada en detectar tráfico interesante a partir de reglas preestablecidas y según acciones disparadas por motores de detección de intrusiones como Snort.

No obstante, los trabajos anteriores no tienen en cuenta aspectos específicos del comportamiento del malware. En algunos casos porque el ámbito de aplicación no está dirigido a malware de propagación automática, sino a evaluar ataques realizados de manera manual. En otros casos el ámbito de estudio se centra más en implementaciones concretas de dispositivos, protocolos o servicios, partiendo del conocimiento previo sobre los ataques sucedidos en ellos. Este trabajo tiene como intención resaltar los beneficios derivados de incorporar información viva existente sobre las campañas de malware actuales, los indicadores de compromiso conocen (IOCs) conocidos, o bien información de inteligencia disponible a través de proyectos como *Malware Information Sharing Platform* (MISP) [11] o *Virus Total Intelligence* (VTI) [12], con vistas a ofrecer un entorno lo más cercano posible al esperado por el malware para que desate toda su carga y pueda ser objeto de escrutinio y análisis.

III. Arquitectura

El objetivo de este artículo es la creación de entornos trampa para capturar la actividad realizada por malware, adaptándolos progresivamente conforme se van generando nuevas evidencias que permitan determinar qué acción desencadenará a continuación. Para ello es necesario diseñar una arquitectura que posibilite la observación y la configuración dinámica de honeypots de acuerdo a tres fases: (1) aproximación, (2) infección y (3) ejecución del payload.

Durante la fase de *aproximación* el atacante trata de descubrir los servicios en ejecución, comprobando si la víctima ofrece una versión vulnerable de alguna aplicación para la que disponga de un mecanismo de ataque. La solución propuesta evaluará el tipo de servicio que el atacante desea encontrar, ofreciéndole un honeypot que se acerque a sus necesidades.

En caso de que la aproximación resulte satisfactoria el ataque pasaría a la segunda fase, *infección*, donde se ejecutaría el código de infección atacando el servicio con propósitos muy diversos. En este punto es importante analizar el vector de infección para discernir qué tipo de acción desea desatar el atacante sobre el honeypot: inyectar código, subir fichero, dejar código, manipular ficheros, enviar email, averiguar clave de un usuario vía TELNET, etc. En la medida de lo posible se le debe hacer ver que dicha acción se ha realizado con éxito.

De esta forma llegaríamos a la tercera fase, la *ejecución del payload*, donde, en función del tipo de actividad realizada, se simulará la ejecución del código, se ejecutará el fichero descargado en un entorno controlado, o bien se simulará el envío del correo electrónico. Siempre teniendo presente la necesidad de registrar las modificaciones que se pretenden realizar en el entorno de la víctima.

Estas fases se gestionarían por medio de los componentes de control mostrados en la Fig. 1: interceptación de conexiones (IC), configuración de servicios trampa (CST) y monitorización de las evidencias generadas (ME). Estos componentes permitirán realizar un seguimiento de las tres fases del ataque descritas anteriormente. Una parte fundamental de este trabajo es que estos componentes se retroalimentan con información que permitirá predecir el siguiente paso que el código malicioso pretende realizar. Actualmente existen multitud de servicios que permiten obtener información sobre la actividad desarrollada por el malware, ya sea mediante direcciones IPs pertenecientes a campañas de malware, vectores de infección conocidos, firmas o patrones de búsqueda, o por la ejecución explícita de código malware y la observación de las acciones ejecutadas y cambios realizados en el sistema. La solución propuesta permitirá orquestar esta información para la adaptación de los honeypots.

Además de estos elementos principales, se ha contemplado la utilización de dos entornos trampa diferentes que ofrecer al atacante: (i) honeypots especializados en determinados protocolos y servicios, y (ii) entornos de alta interacción donde ejecutar los ficheros generados por el malware.

La relación existente entre los componentes se detalla a continuación.

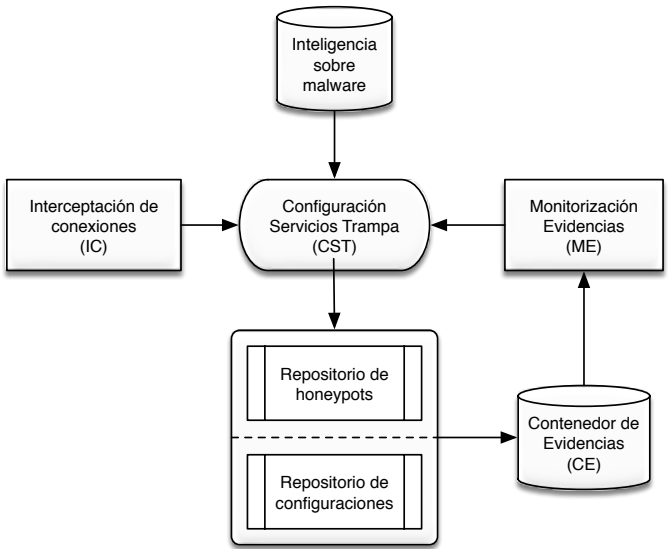

Fig. 1. Diagrama de arquitectura

A. Interceptación de conexiones

El funcionamiento de un honeypot típico consiste en permanecer a la escucha en una serie de puertos preestablecidos, de manera que al realizarse una conexión a uno de estos puertos responde según se ha predeterminado en su configuración. Algunos honeypots de baja interacción únicamente reciben peticiones y las almacenan (p.ej. honeyd) mientras otros de media interacción (p.ej. Inetsim [13]) son capaces de operar a nivel de servicio, respondiendo a las peticiones del usuario según los patrones configurados por su administrador.

El componente para la *interceptación de conexiones* (IC) tendrá como función (i) permanecer a la escucha en todos los puertos objeto de estudio, (ii) recibir y aceptar conexiones y (iii) enviar peticiones de servicio al siguiente componente para la configuración de los honeypots (CST, Fig. 1). En dichas peticiones se incluirá toda la información que se haya podido recopilar en el momento de establecer la conexión (IP y puerto origen, cabeceras del protocolo enviadas, etc.) de manera que el componente de configuración de servicios trampa pueda estimar, con mayor exactitud, el honeypot más adecuado al tipo de conexión recibida.

B. Configuración de servicios trampa

El componente para la *configuración de servicios trampa (CST)* será capaz de discernir, de manera dinámica, qué honeypot es el más conveniente para el tipo de ataque que se va a desarrollar. Para ello es preciso disponer de una batería de honeypots pre-configurados, de manera que sea fácil conmutar de uno a otro en función de las necesidades del atacante, o bien realizar modificaciones en tiempo de ejecución sobre la configuración de los mismos.

Por ejemplo, el protocolo SMTP se suele emplear para el envío de correo electrónico con adjuntos que incluyen algún tipo de código malicioso. Si se recibe una petición al puerto 25 es necesario redirigir esa conexión a un honeypot que sea capaz de gestionar la recepción del correo y la extracción del fichero adjunto. En cambio una petición al puerto HTTP/HTTPS puede tener objetivos muy dispares: bien puede tratarse de un ataque sobre el servidor web de Apache en Windows o en Linux, sobre IIS, etc. o en cambio ser un ataque sobre una versión concreta de WordPress, Joomla, etc. Esta diversidad complica la preparación previa de un honeypot que encaje con el ataque recibido.

En definitiva, este componente procesará las peticiones de conexión para poder elegir qué honeypots se deben iniciar o adaptar para el atacante. Esta elección se deberá tomar en base a la información disponible sobre el ataque en curso, a partir de:

- **IP origen**: se deberá comprobar si esa dirección IP pertenece a una campaña de malware actualmente en curso. Caso de ser así, obtener información sobre el malware y los servicios y aplicaciones a los que afecta.
- **IP destino**: si se ha ejecutado algún fichero, y este ha provocado una conexión al exterior, es probable que tengamos que adaptar los servicios trampa, o al menos habilitar la conexión hacia entorno en el que se ha ejecutado el fichero.
- **Cabeceras del protocolo**: los primeros mensajes de muchos protocolos dan información sobre el servicio que espera encontrar en ese puerto, será necesario comprobar estos elementos para decidir qué honeypots levantar.
- **Información del servicio**: direcciones IP destino, carpetas y ficheros en URLs o en servidores de ficheros, procesos en ejecución, entradas en el DNS, etc. es información que facilitará la configuración de honeypots adecuados.
- **Ficheros asociados**: los ficheros descargados contienen información útil sobre el sistema operativo destino, librerías necesarias, uso de recursos, etc. que puede ser usada para seleccionar el entorno de ejecución con mayores posibilidades de éxito.

A modo de ejemplo, la Fig. 2 muestra un grafo del flujo de ejecución de este componente en dos casos particulares: (i) se recibe una petición de servicio desde el componente de interceptación de conexiones, y (ii) la petición se recibe desde el componente de monitorización al detectarse un fichero nuevo en el repositorio de evidencias.

En el primer caso, se procedería a analizar la información relativa a la conexión recibida (lease IP origen/destino, protocolo, datos del servicio, ficheros o carpetas destino, etc.). Esta información será utilizada para intentar averiguar cuál es el malware que está detrás de la conexión. Para ello, se cotejará con la *información de inteligencia* disponible, consultando con servicios de inteligencia externos para detectar coincidencias. En caso de que exista información previa se usará para adaptar un honeypot ya pre-configurado de manera que se ajuste todo lo posible al escenario que desea encontrar el atacante.

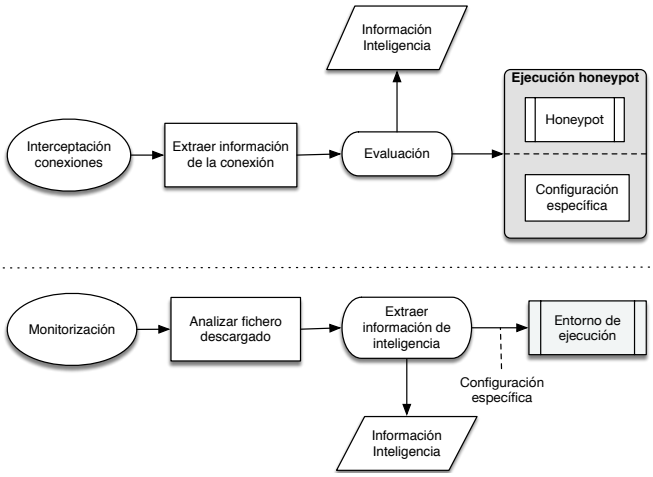

Fig. 2. Configuración de servicios trampa

El segundo caso reflejado en la Fig. 2 corresponde a un escenario en el que el atacante ha provocado la descarga de algún tipo de fichero, bien en un honeypot configurado con anterioridad o en un entorno de ejecución. El proceso de monitorización detectará la existencia de una nueva evidencia e iniciará un proceso de análisis consistente en consultar con los servicios de inteligencia para obtener información previa y crear un entorno de ejecución adecuado para el fichero descargado.

El objetivo final de este componente, por lo tanto, será la de facilitar la continuidad del ataque a la vez que se monitoriza y registra toda la actividad desencadenada.

C. Monitorización de evidencias

La arquitectura prevé la utilización de un contenedor de evidencias donde se depositará cualquier tipo de contenido generado durante el ataque, ya sean ejecutables, código interpretado, código binario, imágenes, documentos, etc. El objetivo de este contenedor de evidencias es doble: recopilar toda la información posible sobre las acciones realizadas por el malware, a la vez que dar continuidad al mismo mediante la ejecución de las distintas fases que implementa.

El componente de *monitorización de evidencias (ME)* controla la aparición de nuevas evidencias creadas en el repositorio. Cuando se detecta un nuevo elemento se analiza su contenido (tipo de fichero, sistema operativo necesario, etc.) para posteriormente enviar una petición de servicio al componente de configuración de servicios trampa. Este último recibirá la información analizada y preparará el entorno de ejecución más adecuado.

D. Despliegue de servicios trampa

Hasta el momento, todos los elementos de la arquitectura descritos corresponden a procesos de control y monitorización. Estos procesos precisan de un conjunto de honeypots preconfigurados para los escenarios más comunes susceptibles de ataque. Afortunadamente existen multitud de honeypots especializados en determinados entornos [3] (Cowrie para SSH, glastopf para HTTP, conpot para PLCs, jackpot para SMTP, elastichoney para elasticsearch, etc.) y otros de ámbito general (inetsim), que pueden usarse en una plataforma como la planteada.

No obstante, los honeypots a utilizar precisan cumplir determinados criterios para que puedan ser empleados por el componente CST:

- Fácilmente configurables mediante la modificación de ficheros de texto.
- Disponer de opciones para la configuración de banners, carpetas de servicios, respuestas ante comandos del protocolo, etc.
- Permitir la configuración del interfaz de red de escucha.
- Incluir funcionalidad para el registro de actividades.
- Disponer de buenas capacidades de ocultación, de manera que el atacante no sea consciente de estar interactuando con un honeypot.

El cumplimiento de estos requisitos permitirá al CST modificar los ficheros de configuración tras la información obtenida sobre el ataque (qué carpetas deben estar disponibles, qué aplicaciones, qué banners de protocolos se esperan encontrar, etc.).

Además de honeypots específicos de baja, media o alta interacción es necesario disponer de entornos de ejecución donde lanzar las evidencias que se puedan obtener durante la fase de ataque. Estos entornos, considerados también como honeypots de alta interacción, estarán formados por máquinas virtualizadas y físicas gestionadas por Cuckoo Sandbox [14]. La elección de este gestor para la configuración de los entornos de ejecución se encuentra fundamentalmente en los siguientes aspectos:

- Ofrece una API que permite enviar ficheros de diversos tipos a entornos preconfigurados de análisis.
- Emite un informe de actividad al concluir la ejecución de los ficheros que es fácil de procesar de manera automática.
- Permite escalar con facilidad el número de entornos de análisis (físicos y virtuales).
- El número y profundidad de los análisis realizados es lo suficientemente completo como para cubrir las necesidades de nuestra plataforma.

IV. Casos de uso

Con vistas a exponer la relación existente entre los distintos componentes de la arquitectura, se ofrece en esta sección una visión del comportamiento que tendría el sistema al ser expuesto a intentos de propagación de dos malwares conocidos: Locky (sección IV-A) y Mirai (sección IV-B).

Se han elegido estos dos ejemplos ya que en conjunto muestran diferentes posibilidades que tendría el relacionar los honeypots adaptativos con información obtenida de inteligencia.

A. Locky

Locky es un ransomware surgido en 2016 que se transmite a través de correo electrónico en varios formatos, pero principalmente como documento MS Word. Emplea macros para descargar un ejecutable en el sistema que cifra los ficheros del usuario reclamando un pago para su restauración.

El punto de partida de una infección con Locky ransomware es la recepción de un correo electrónico (Fig. 3). Esta conexión es recibida por el IC el cual, al estar configurado el puerto 25 con un único posible honeypot, redireccionará la conexión directamente a éste.

De esta manera, el correo electrónico será recibido por un supuesto destinatario y su contenido registrado en el contenedor de evidencias (cabeceras del correo, contenido y adjuntos). El componente ME detectará la creación de una nueva evidencia que, entre otras cosas, contiene un fichero con extensión *.doc*. Para proceder a su análisis enviará al CST una petición para la creación de un honeypot de alta interacción en el que lanzar el documento empleando el visor predeterminado para el tipo de fichero.

El honeypot a iniciar vendrá determinado por la información de inteligencia que haya podido obtener el CST. Normalmente, para malware conocido, basta realizar una hash del fichero para obtener la familia de malware al que pertenece la muestra e información sobre el entorno al que está destinado (sistema operativo, arquitectura 32/64 bits, necesidades de librerías externas, etc.). Aunque los honeypots de alta interacción disponibles dispongan de todas las aplicaciones usuales preinstaladas, hay aspectos como el sistema operativo o la arquitectura del procesador que son determinantes a la hora de seleccionar un entorno de ejecución adecuado. Por otro lado, si se desconoce la familia del malware a partir del hash, podríamos comprobar datos como la IP desde la que se envió el correo electrónico, URLs, direcciones IP que contiene, etc.

En este caso particular, Locky afecta a sistemas Windows y la muestra ha sido testeada en Windows 7 en procesadores Intel de 32bits según los datos recogidos de inteligencia. El configurador de servicios seleccionará, por lo tanto, un

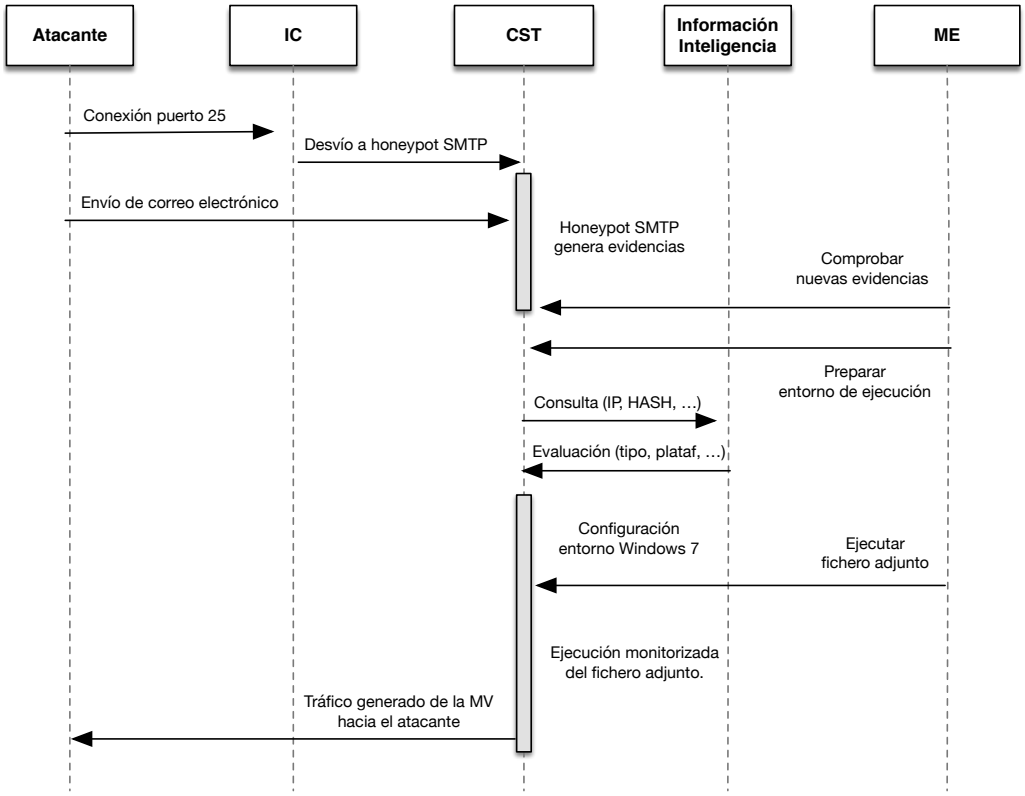

Fig. 3. Caso de uso: Locky Ransomware

honeypot consistente en un equipo Windows 7 32 bits con $wordview.exe$ instalado.

La ejecución de $wordview.exe$ sobre el documento provocará la descarga del payload contenido en él (un fichero ejecutable de 32bits). Acto seguido Locky copiará este fichero al directorio temporal empleando como nombre $svchost.exe$ y borrará el fichero original. En el siguiente paso contactará con el centro de mando y control (C2) para obtener un mensaje en el idioma del usuario, informándole de que los ficheros se están cifrando, e instándole a acceder a una página web personalizada en TOR donde encontrará instrucciones sobre cómo recuperar sus ficheros.

Este comportamiento disparará diferentes eventos en nuestra plataforma:

- Los nuevos ficheros creados son automáticamente almacenados en el contenedor de evidencias.
- Las nuevas conexiones de red son registradas y consultadas posteriormente con inteligencia por si fuese preciso levantar algún servicio adicional. En este caso el tráfico está cifrado, pero la IP destino podría revelar un nuevo nodo C2 no conocido todavía.
- La información volcada en la pantalla para informar al usuario se registra en el contenedor de evidencias, revelando las direcciones de TOR donde el atacante ha colocado las instrucciones a seguir.

B. Mirai

Mirai es una botnet que ataca principalmente a dispositivos empotrados típicos en IoT. En 2016 se hizo famosa por originar un ataque DDoS sobre los servicios DNS del proveedor DYN que ocasionó la desconexión de los servicios GitHub, Twitter, Reddit, Netflix, Airbnb entre muchos otros. Principalmente atacaba servicios TELNET mediante ataques por diccionario, para convertir los equipos infectados en nuevos nodos de la botnet que emplear en los ataques DDoS [15].

En este escenario (Fig. 4), el componente IC recibe una petición de conexión al puerto 23. A continuación consultará la información de inteligencia existente sobre la IP origen de la conexión, con el fin de determinar si es preciso adaptar el honeypot destinado a las conexiones TELNET. La información obtenida revela que el nodo desde el que se realiza la conexión pertenece a una botnet de Mirai, por lo que se inicia la configuración predeterminada del honeypot.

Mirai realiza conexiones TELNET por dos motivos: i) desde un bot para detectar que este puerto está abierto con claves conocidas y, ii) desde un *loader* para provocar la descarga de un fichero malware. En ambos casos la configuración será modificada por el CST para mostrar alguna de las arquitecturas de procesador determinadas por el servicio de inteligencia (ARM en este caso), ya que Mirai cuenta con versiones para distintas arquitecturas.

Suponiendo que estamos en el segundo caso, conexión desde un loader de Mirai, el honeypot registrará los comandos a ejecutar e incluso los ficheros que el *loader* ha especificado que se instalen. Algunas de las implementaciones de honeypots del servicio TELNET son capaces de interpretar correctamente comandos habituales de descarga de ficheros como $curl$ o $wget$ (p.ej. Cowrie). Los ficheros descargados a petición del atacante son almacenados en el contenedor de evidencias.

Una vez que se registra la entrada de una nueva evidencia, el

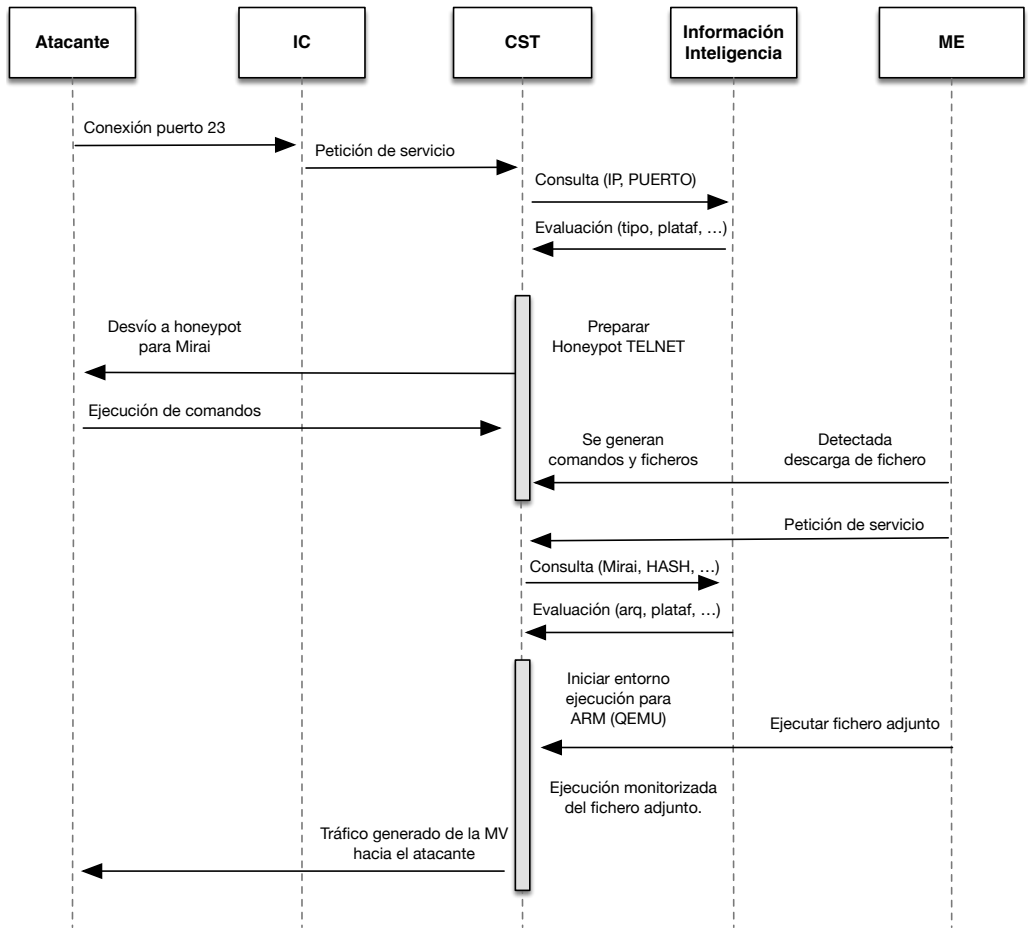

Fig. 4. Caso de uso: Mirai Botnet

componente de monitorización la detecta y envía una petición al CST para que prepare un honeypot para su ejecución. Éste empleará la información de inteligencia obtenida sobre el fichero (p.ej. malware para arquitecturas ARM de 32 bits en entorno Linux), preparando un entorno de emulación que permita ejecutar el fichero (p.ej. QEMU [16]).

Una vez que el fichero ha pasado a ejecución, se podrá observar una conexión hacia una IP externa que, de contrastar con la información de inteligencia disponible, podría revelar un nuevo nodo C2 de Mirai o bien confirmar la actividad de uno ya conocido. El mantener un honeypot actuando como bot de Mirai nos permitirá acceder a información sobre el funcionamiento del C2, como por ejemplo la ejecución de órdenes de ataque sobre direcciones destino.

V. Discusión

La construcción de una plataforma de captura de malware como la diseñada en este trabajo conlleva superar una serie de obstáculos, algunos relativos a criterios propios de implementación y otros de diseño, pero que en gran medida estarán fundamentalmente centrados en cómo integrar la información de inteligencia para obtener el despliegue del mayor número de funcionalidades del malware.

La arquitectura propuesta empleará multitud de honeypots ya disponibles, algunos de uso genérico y otros centrados en determinados protocolos, ofreciendo en conjunto un amplio abanico de soluciones de baja y media interacción para la captura de evidencias. Sin embargo, será preciso desarrollar los tres principales componentes (IC, CST y ME) que posibilitan la construcción dinámica de los escenarios de captura.

A continuación se analizan cuestiones a considerar sobre la implementación de los componentes principales de la arquitectura propuesta.

A. Interceptación de conexiones

Con respecto al componente de *interceptación de conexiones* (IC), existen actualmente aplicaciones como *honeytrap* [4], que realizan una labor parecida a la descrita para IC en esta arquitectura. Honeytrap es un proceso que permanece a la escucha en todos los puertos, de manera que cuando se produce una conexión es capaz de redireccionarla hacia un honeypot predeterminado para que atienda la petición. También tiene la capacidad de aplicar técnicas de *proxy* inverso y *mirroring* para atender las conexiones entrantes. El único inconveniente de esta aplicación es que el honeypot que se sirve está previamente descrito en función del puerto al que acceda el atacante. No obstante, podría servir de base para construir el componente IC que la arquitectura necesita.

B. Monitorización de evidencias

El componente de *monitorización de evidencias* (ME) debe relacionar los distintos eventos que se van produciendo, de manera que las evidencias que se detectan, por ejemplo por la ejecución de ficheros, queden vinculadas a la conexión inicial

que se recibió en el IC. Es decir, es necesario orquestar la ejecución de las distintas fases del ataque bajo un mismo caso, de manera que facilite la correlación y la generación de informes a posteriori.

Hasta el momento se han detectado tres motores diferentes de generación de evidencias:

- El componente IC, registrando las conexiones que se establezcan con algún honeypot.
- Los honeypots de baja y media interacción generarán ficheros y/o comandos, ya sea mediante simulación de servicios o por la ejecución emulada de ficheros para una plataforma distintas a la del sistema.
- Los honeypots de alta interacción crearán ficheros y conexiones como consecuencia de la ejecución de malware en ellos.

En este sentido, la selección de Cuckoo Sandbox [14] para la gestión de los honeypots de alta interacción facilitará, en gran medida, la organización de las evidencias ejecutadas en máquinas virtualizadas y físicas. Esto es debido a que Cuckoo crea una estructura de directorios y ficheros diferente para cada lanzamiento de una máquina de análisis, conteniendo todas las evidencias recolectadas.

C. Configuración de Servicios Trampa

La lógica de las decisiones de despliegue de entornos trampa será uno de los aspectos fundamentales a desarrollar para este componente. Este deberá decidir, en función de la información que pueda reunir, cuál es el entorno más apropiado para servir al atacante. Para ello hará uso de información disponible a través de multitud de servicios de información de inteligencia sobre malware, que englobamos en tres niveles:

L1. Servicios que ofrecen listados de direcciones IPs comprometidas, pertenecientes a botnets o que forman parte de alguna campaña actual de despliegue de malware.

L2. Servicios que permiten consultar información relativa a ficheros o URLs maliciosas, para obtener la familia del malware al que pertenece, arquitectura y sistema operativo destino, y en muchos casos información relativa a la implementación: librerías asociadas, técnicas anti-análisis o anti-vm, procesos a los que inyecta código, etc.

L3. Servicios de intercambio de información sobre malware (malware intelligence). Estos servicios ofrecerían la información más actualizada en relación a la difusión de malware, ya que no sólo permite acceder a IOCs publicados sino también a información sobre incidencias actualmente bajo estudio, por lo que toda la información recopilada no ha sido publicada aún.

Con respecto a los servicios L1, existen un amplio abanico de proyectos que ofrecen indicadores de compromiso sobre malware específico o sobre familias de malware, tipos de dispositivos, etc.

De cara a no perder información valiosa, emplearemos plataformas que permiten agregar feeds combinando varias fuentes. Por ejemplo, *Critical Stack Intel* [17] incluye hasta la fecha 118 feeds y ofrece una aplicación para facilitar su integración con otras soluciones. Si buscamos el dominio $wrcwdxjh.org$ podríamos encontrar una salida similar a la siguiente:

```
wrcwdxjh.org Intel::DOMAIN
from malwaredomains.com,locky
via intel.criticalstack.com F
```

Lo que revelaría que el dominio está relacionado con el ransomware *Locky*.

Para la utilización de servicios L2, hay varias alternativas, aunque dos son las soluciones que, a priori, pueden cubrir todas las necesidades con respecto al análisis de ficheros:

- *Virus Total Intelligence* (VTI) [12]: este servicio permite realizar búsquedas empleando multitud de parámetros relativos al formato y operativa del malware. Además del hash de un fichero es posible consultar IPs o URLs vinculadas a malware, así como buscar por características del propio fichero/URL (sistema operativo, arquitectura, recursos contenidos en el fichero, recursos del sistema usados, etc.). Aunque Virus Total ofrece sin coste acceso a su servicio principal para la evaluación de los antivirus, VTI es un servicio de pago.
- *Hybrid Analysis de PAYLOAD Security* [18]: similar al anterior pero con un conjunto de criterios de búsqueda y una base de muestras de malware menor. En cambio, el análisis que realiza sobre los ficheros añade algunas características interesantes no proporcionadas por VTI como, por ejemplo, un listado con las técnicas anti-análisis implementadas. Este servicio permite la realización de consultas, de manera gratuita, limitando el número máximo de peticiones que el usuario puede realizar.

Ambos servicios ofrecen los resultados de sus consultas en formato JSON para facilitar su procesamiento automático. Una consulta usual será buscar, en base al hash del fichero almacenado en el CE, información relativa a la naturaleza de la evidencia generada. El componente CST buscará especialmente los siguientes datos con vistas a configurar el entorno de ejecución más apropiado:

- Arquitectura y sistema operativo.
- Comunicaciones realizadas con un dominio o IP.
- Ficheros leidos o modificados.
- Librerías y métodos empleados.
- Formato del fichero.
- Técnicas anti-análisis implementadas.

En los casos en los que no hubiese rastro previo del fichero generado en estos servicios, se enviará éste para obtener una primera evaluación de las características y recursos empleados.

Los servicios de nivel L3 serán de utilidad cuando la información recabada por los servicios anteriores no sea concluyente o bien esté ausente. Estos servicios dan acceso a información de inteligencia compartida por analistas de malware de diferentes organizaciones. En ocasiones este tipo de servicios da acceso a información sobre campañas activas de malware no publicadas todavía, ya que son sujeto de estudio por parte de los analistas y únicamente son reflejadas como *actividades sospechosas*.

La solución hará uso de la plataforma MISP [11] para acceder a información de inteligencia. MISP ofrece diferentes vías para el intercambio de información, siendo especialmente valiosa la API para la consulta y obtención de eventos en multitud de formatos (MISP XML, MISP

<Attribute><id>3146</id><org_id>76</org_id><info>[IOC] Lucky Locky the sequel Part II</info><value>31.41.47.37</value>
</Attribute></RelatedAttribute><RelatedAttribute><Attribute><id>3142</id><org_id>2</org_id><info>LOCKY Ransomware via
.doc/.docm/.xls/.zip(.js) files (constantly updated)</info><value>31.41.47.37</value></Attribute></RelatedAttribute>
</Attribute>
<Attribute><id>383019</id><type>md5</type><category>Payload installation</category><to_ids>1</to_ids><uuid>56f58c3b-
4968-4f48-9cbf-faf1950d210f</uuid><event_id>3512</event_id><distribution>5</distribution>
<timestamp>1458932795</timestamp><comment/><sharing_group_id>0</sharing_group_id><deleted>0</deleted>
<disable_correlation>0</disable_correlation><value>3f118d0b888430ab9f58fc2589207988</value><ShadowAttribute/>
<RelatedAttribute><Attribute><id>4585</id><org_id>2</org_id><info>OSINT - FBI Flash MC-000077-MW - Identification of
ransomware variant called Locky</info><value>3f118d0b888430ab9f58fc2589207988</value></Attribute></RelatedAttribute>
</Attribute>

Fig. 5. Consulta en MISP sobre el hash de un fichero

JSON, STIX, STIX JSON, CSV, etc.). Por ejemplo, una consulta sobre el hash (md5) de un fichero con valor *3f118d0b888430ab9f58fc2589207988* devolvería un fichero (MISP XML) en el que se relaciona ese hash con una campaña actual de propagación de Locky ransomware. La Fig. 5 muestra una parte de la respuesta obtenida con la búsqueda, en la que además de la información relativa a ese fichero se relaciona el mismo con varias IPs pertenecientes a nodos de mando y control de Locky.

VI. Conclusiones

La capacidad que los servicios de distribución de inteligencia sobre malware tienen para reconocer ataques recientes es notable. Por ello la información que generan alimentan multitud de soluciones de defensa como los cortafuegos y los sistemas de detección de intrusiones. Incluso servicios como Proxy y DNS emplean esta información para evitar llevar al usuario a sitios maliciosos.

Sin embargo la literatura no relaciona su uso con la construcción de sistemas trampa. Mostrar la versatilidad que permiten para detectar ataques y configurar trampas a medida es uno de los objetivos que pretende este trabajo. Además se ha diseñado una arquitectura funcional que propone un mecanismo para el despliegue automático de honeypots de acuerdo a la información obtenida de inteligencia.

El siguiente paso consistirá en diseñar e implementar los componentes principales de la arquitectura (IC, CST y ME), exponiéndolos a una batería de malware para reflejar sus capacidades reales de adaptabilidad.

Agradecimientos

Este trabajo ha sido financiado por la Junta de Andalucía a través del proyecto FISICCO (TIC-07223), y por el Ministerio de Economía y Competitividad a través de los proyectos PERSIST (TIN2013-41739-R) y SMOG (TIN2016-79095-C2-1-R).

References

[1] "Ccn-cert ia-09/16 ciberamenazas 2015/tendencias 2016, resumen ejecutivo." Centro Criptológico Nacional(CCN), Tech. Rep., 2015.

[2] Cymmetria. (2016) Mirai open source iot honeypot. [Online]. Available: http://blog.cymmetria.com/mirai-open-source-iot-honeypot-new-cymmetria-research-release

[3] M. Nawrocki, M. Wählisch, and T. C. Schmidt, "A Survey on Honeypot Software and Data Analysis," *arXiv.org*, no. 10, pp. 63–75, 2016.

[4] T. Werner. (2009) honeytrap – a dynamic meta-honeypot daemon. [Online]. Available: http://blog.cymmetria.com/mirai-open-source-iot-honeypot-new-cymmetria-research-release

[5] S. Patil, N. Karhade, and Y. Kothekar, "Honeyweb: a web-based high interaction client honeypot," *International Journal of Engineering Research and Applications (IJERA)*, vol. 2, no. 5, pp. 1695–170, 2012.

[6] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, "IoTPOT - A Novel Honeypot for Revealing Current IoT Threats." *JIP*, vol. 24, no. 3, pp. 522–533, 2016.

[7] A. Pauna and V. V. Patriciu, "CASSHH – Case Adaptive SSH Honeypot," in *Recent Trends in Computer Networks and Distributed Systems Security*. Springer, Berlin, Heidelberg, Mar. 2014, pp. 322–333.

[8] G. Wagener, R. State, and T. Engel, "Adaptive and self-configurable honeypots," ... *Management (IM)*, 2011.

[9] J. Guarnizo, A. Tambe, S. S. Bhunia, M. Ochoa, N. O. Tippenhauer, A. Shabtai, and Y. Elovici, "SIPHON - Towards Scalable High-Interaction Physical Honeypots." *CoRR*, vol. cs.CR, 2017.

[10] W. Fan, D. Fernández, and Z. Du, "Adaptive and Flexible Virtual Honeynet," in *Mobile, Secure, and Programmable Networking*. Cham: Springer, Cham, Jun. 2015, pp. 1–17.

[11] C. Wagner, A. Dulaunoy, G. Wagener, and A. Iklody, "Misp: The design and implementation of a collaborative threat intelligence sharing platform," in *Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security*. ACM, 2016, pp. 49–56.

[12] G. Inc. (2017) Virus total intelligence. [Online]. Available: https://www.virustotal.com

[13] T. Hungenberg and M. Eckert. (2014) Inetsim: Internet services simulation suite. [Online]. Available: http://www.inetsim.org

[14] C. Guarnieri, A. Tanasi, J. Bremer, and M. Schloesser. (2012) The cuckoo sandbox.

[15] K. Angrishi, "Turning internet of things(iot) into internet of vulnerabilities (iov) : Iot botnets," Feb. 2017.

[16] F. Bellard, "Qemu, a fast and portable dynamic translator." in *USENIX Annual Technical Conference, FREENIX Track*, 2005, pp. 41–46.

[17] C. S. Inc. (2017) Critical stack intel // feed. [Online]. Available: https://intel.criticalstack.com

[18] P. Security. (2017) Free automated malware analysis service. [Online]. Available: https://www.hybrid-analysis.com