ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

MÁSTER UNIVERSITARIO EN INGENIERÍA INFORMÁTICA

ANÁLISIS DE VULNERABILIDADES EN IOT PARA EL DESPLIEGUE DE HONEYPOTS

ANALYSIS OF IOT VULNERABILITIES FOR HONEYPOT DEPLOYMENT

Realizado por
Antonio Acién Gómez
Tutorizado por
Javier López Muñoz y Ana Nieto Jiménez
Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, Febrero de 2018

Contenido

Tabla de ilustraciones

Índice de tablas

Declaración de originalidad

Yo, Antonio Acién Gómez, declaro que el trabajo que aquí se presenta es totalmente original, y que se han citado correctamente las fuentes de las que se ha obtenido información.

1. Introducción

La *Internet de las Cosas* (Internet of Things, IoT) ha iniciado una revolución tecnológica que afecta al sector público y empresarial, y a los usuarios que día a día delegan en sus objetos personales parte de su rutina diaria. Son de dominio público las estimaciones de dispositivos conectados que se esperan en 2020, que varían entre los diferentes estudios de mercado [Blan2016][Vest2010][Rose2015], pero que en todo caso superan los veinte mil millones (siendo comedidos). Sin embargo, para tratar con exactitud el problema, se ha de definir en qué consiste esta tecnología, sus aspectos de seguridad, y el enfoque dado en este trabajo, los cuales se pasan a describir a continuación.

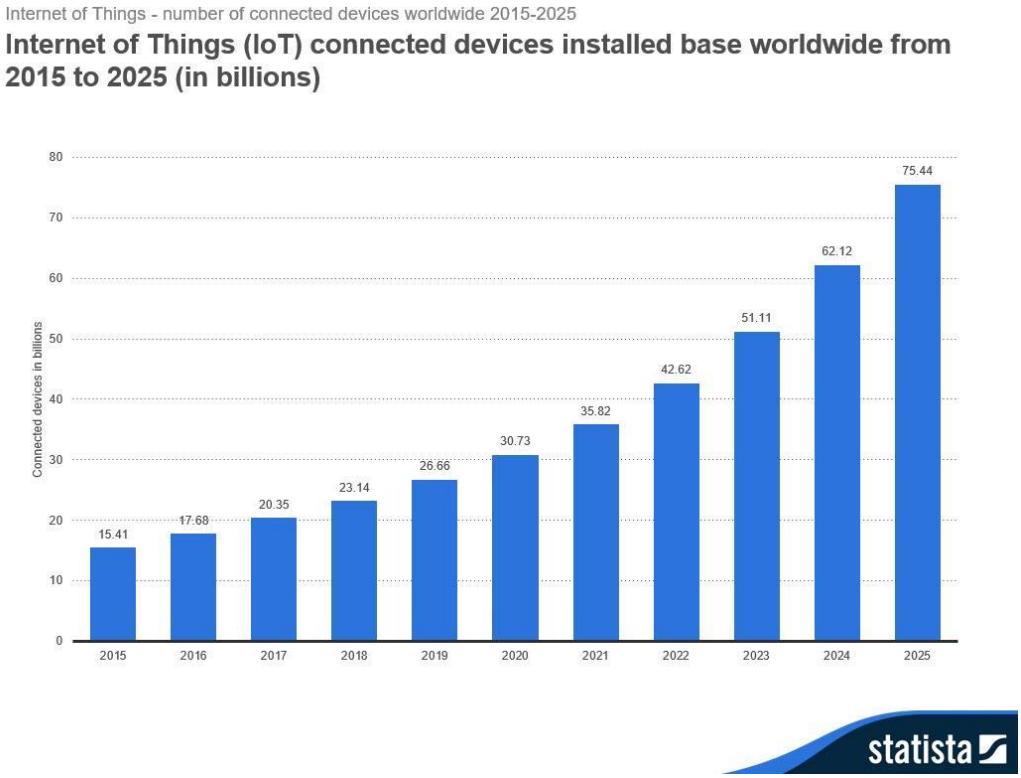

Figura 1: Predicción de número de dispositivos IoT conectados [Stat2018]

El concepto de IoT se originó en 1999 [Asht2009], y ha ido mutando de significado. En la actualidad, IoT se refiere a los dispositivos (ya sean multimedia, electrodomésticos, vehículos, *wearables*…) que contienen software, sensores, actuadores y una conexión de red que les permite intercambiar datos.

Los dispositivos IoT son algo que está muy presente en nuestra vida diaria. Desde un simple router doméstico o Smart TV, a cámaras de seguridad que se conectan con la nube para poder verlas a distancia, o podómetros y relojes que permiten hacer estadísticas de las rutinas de entrenamiento. Cada vez más aparatos permiten la conexión a internet, y están más normalizados en el día a día, llegando a situaciones en las que toman parte en escenarios

delicados, dependiendo de ellos nuestros enseres (*Smart locks* en cerraduras de domicilios), nuestra privacidad (cámaras) o incluso nuestras vidas (automóviles).

El panorama que presentan los dispositivos IoT es muy diverso, pues utilizan tecnologías muy distintas, así como realizar funciones totalmente diferentes entre ellos. Cada uno supone una puerta de entrada distinta a ataques y amenazas, y teniendo en cuenta lo integrados que están en la sociedad actual, su seguridad debe ser un aspecto muy a tener en cuenta.

2. Contextualización

Ante la alta cifra de dispositivos IoT estimados para los años venideros, y el amplio abanico de plataformas y servicios que se han desplegado de forma independiente y en tan breve intervalo de tiempo, es imposible predecir con exactitud el efecto que un ataque dirigido puede ocasionar sobre la red de infraestructuras actual.

Ya en 2011 [Roma2011], la problemática de seguridad comenzó a considerarse seriamente debido al impacto de numerosos ataques efectivos sobre los objetos que se consideran parte de la IoT (p.ej. a través de routers y otros dispositivos empotrados) [Cui2011]. A los ataques no dirigidos necesariamente, perpetrados de forma automática por botnets y otros actores, se suman los ya conocidos *Advanced Persistent Threats* (APT), ataques sofisticados dirigidos sobre usuarios e infraestructuras.

A menudo, se observa en los aparatos IoT disponibles en el mercado, que las implementaciones relativas a la seguridad en el software que incluyen suelen ser pobres, o incluso inexistentes, habiendo casos de dispositivos que no permiten cambiar contraseñas o que cuentan con comunicaciones con servidores externos que no se pueden desactivar [Yu2015]. No hay ningún estándar fijado con pautas de seguridad que se deban incluir en IoT, aunque se está promoviendo legislación al respecto [Kreb2017]. Sin embargo, aunque pasase a haber unos mínimos obligatorios, el problema persistiría, al haber en el mercado una gran cantidad de dispositivos cuya fabricación es anterior, y al no poder aplicarse una ley de manera global, ya que los productos se fabrican en países distintos.

Durante 2016 y 2017, hubo varios casos de malware que, dada su manera de funcionar, se definían como *botnets* (red de dispositivos que operan de forma automática). Estos ataques aprovechaban vulnerabilidades presentes en dispositivos IoT, o bajas medidas de seguridad como credenciales por defecto, para entrar en ellos y controlarlos o borrar sus datos. Estos fueron Tsunami, Amnesia (una modificación del anterior, que borraba los datos de los dispositivos), Gafgyt y BrickerBot. [Como2017] Algunos de ellos, llevaban años presentes en

sistemas Unix, pero no habían supuesto una amenaza real hasta que no han sido combinados con un escenario con baja seguridad y conexión directa a internet, como IoT. [Purs2017]

Sin embargo, el ataque que tuvo una mayor repercusión fue Mirai, a finales de 2016, el mayor ataque de denegación de servicio (DoS) conocido hasta la fecha, con una magnitud de 1.2 Tbps, dejando inoperativos para sus clientes servicios tan grandes como Twitter, Spotify, Amazon, GitHub o Netflix, atacando a su proveedor de DNS [Anto2017]. Este ataque, similar a los descritos anteriormente, ataca dispositivos IoT vulnerables, haciendo que sean parte de una red de bots (botnet) que realiza consultas a un objetivo concreto y a su vez, escaneando la red en busca de nuevos dispositivos a los que propagarse.

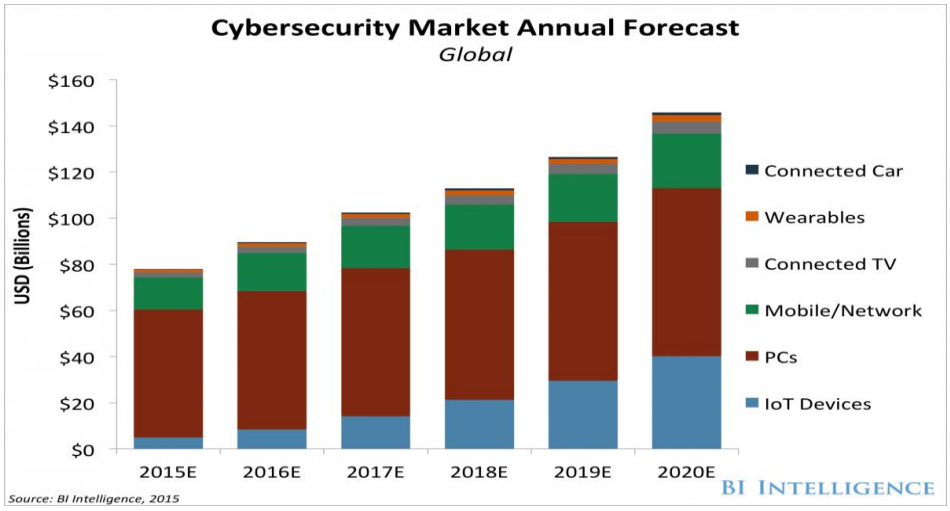

Figura 2: Estimación del mercado anual de ciberseguridad [Busi2015]

Aunque ya había advertencias sobre ello por parte de expertos, Mirai hizo saltar las alarmas relativas a la seguridad en IoT, haciendo la preocupación en este aspecto más patente, y destinándose cada vez más presupuesto al mismo dentro del general en ciberseguridad, como se puede ver en la Figura 2.

El código de Mirai fue liberado, mostrando que se valía de una lista de combinaciones usuario/contraseña que los fabricantes configuraban por defecto para conectarse a los dispositivos de manera remota. Esto hizo saltar las alarmas sobre la flagrante falta de seguridad y la ausencia de la implementación de patrones básicos para evitar estos ataques. La liberación del código hizo que surgiesen nuevas modificaciones y versiones del ataque [Hay2016], explotando más vulnerabilidades que las anteriores [Koli2017], y empeorando en general la situación.

Algunas de estas modificaciones aprovechaban la infección a los dispositivos IoT para *minar bitcoins*, mientras que otras (Hajime) no tenían un comportamiento malicioso aparente, y se

especula que podrían ser malware “bienintencionado” que cierra puertos a los dispositivos IoT para que no sean infectados por Mirai o sus variantes [Koli2017], un proyecto de investigación o hobby, o bien un ataque que aún no se ha hecho efectivo [Edwa2016].

Las variaciones más sofisticadas han aprovechado vulnerabilidades que han ido surgiendo y haciéndose públicas, muchas de las cuales eran sencillas de explotar. Algunas de ellas sacan partido de una vulnerabilidad que se hizo pública en 2017, la cual afecta a más de 1200 modelos de cámaras IP, entre ellas, productos de marcas como D-Link, Logitech, Netcam o Polaroid. Esta variante ha sido nombrada como Persirai [Yeh2017].

Ciertas versiones de Mirai actuaron sobre implementaciones vulnerables de un protocolo llamado TR-064, presente en routers de marcas como Eir o DSL, sumando potencial a la peligrosidad de estos ataques. [Reve2016]

Debido a esta serie de problemáticas y ataques de naturaleza diversa, varias empresas dedicadas a la ciberseguridad empezaron a estudiar el tema más a fondo. Se puede ver que a medida que IoT se convierte en un concepto más popular y usado en el día a día, la cantidad de malware que atrae, específicamente destinado a su plataforma, crece. Compañías como Kaspersky detectan cada vez más muestras de malware, creciendo más del doble cada año.

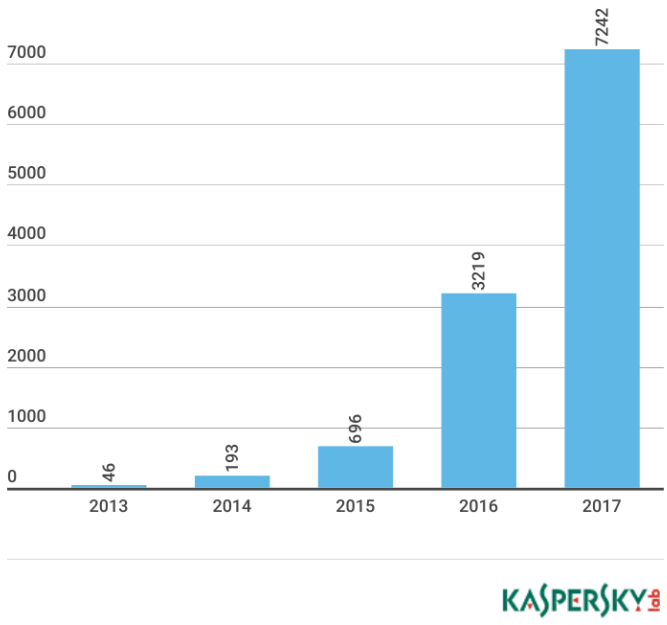

Figura 3: Muestras de malware IoT distintas recogidas por los honeypots de Kaspersky [Mimo2017]

En general, se observa cómo la atención que le presta el mercado a IoT y sus diferentes aspectos no hace más que crecer, y esta tendencia parece mantenerse. Por estas razones es necesario

establecer un sistema que nos permita recabar información de manera segura sobre los potenciales ataques que un dispositivo IoT puede recibir, y los honeypots se prestan a la perfección para esta tarea.

3. Objetivos

En este trabajo se persiguen unos objetivos generales, que se subdividen en tareas más pequeñas y concisas, desarrolladas en las posteriores secciones. A continuación, se detallan, indicando también dónde se resuelven.

O1. **Contextualizar el entorno IoT en base a las vulnerabilidades y amenazas más recientes**

El entorno de los dispositivos IoT es especialmente heterogéneo, con aparatos que se usan tanto en ámbitos industriales (sistemas SCADA, contadores de luz), personales (monitores de bebé, *wearables*) o empresariales (cámaras de seguridad, routers). Para conocer el campo sobre el que se desarrolla el trabajo, es necesario conocer las características de estos dispositivos, y establecer criterios como las vulnerabilidades que presenten o la popularidad de uso de distintas marcas o modelos.

O1.1. **Análisis de la seguridad en IoT:** Para conocer el escenario a tratar, se debe llevar a cabo un estudio sobre la seguridad en este campo. Esto engloba tanto ataques conocidos hasta la fecha, como fallos de seguridad revelados, incluyendo también los trabajos precedentes que trataron la problemática de la seguridad IoT.

O1.2. **Análisis de dispositivos IoT:** Ya que las vulnerabilidades presentadas por los dispositivos y la popularidad de la que gozan son buenas herramientas para saber en qué dispositivos IoT enfocarse, se realizarán búsquedas atendiendo a estos criterios usando herramientas como bases de datos de vulnerabilidades y tiendas online.

O2. **Identificar directrices e indicios que permitan determinar la prioridad en la emulación de dispositivos señuelo**

La densidad de dispositivos IoT, así como la disponibilidad de herramientas que permiten saber si un dispositivo es un honeypot, son factores que dificultan el despliegue de sistemas para recabar información sobre ataques estos entornos. Una clasificación exhaustiva de las características de los dispositivos, así como de las soluciones existentes en materia de honeypots, ayudará a mejorar los sistemas reactivos.

O2.1. **Clasificación de dispositivos IoT:** En base a la información obtenida en el análisis llevado a cabo mediante los métodos presentados en la subsección 4.1, se procede a una clasificación de diversos dispositivos que suscitan interés, ya sea por su amplia

implantación, o por sus características de seguridad. Ambas se detallarán junto a los modelos concretos y las fuentes de dicha información.

O2.2. **Análisis de soluciones honeypot IoT:** Para conocer las soluciones existentes de honeypots IoT es necesario analizar la bibliografía existente al respecto. Se lleva a cabo un estudio de las soluciones propuestas por otros autores, detallando sus características (protocolos usados, carencias, puntos fuertes…) y se establece una clasificación para los mismos, para conocer la viabilidad de su uso atendiendo a estos criterios.

O2.3. **Análisis de otros honeypots relacionados:** Hay honeypots que, aunque no estén específicamente diseñados para ser sistemas trampa IoT, nos pueden ser útiles y son utilizados en algunos trabajos relacionados, por lo que también tendrá cabida su análisis.

O3. **Responder a la necesidad de mejorar la metodología de despliegues de honeypots para entornos con dispositivos restringidos en recursos (IoT)**

Teniendo en cuenta el interés que suscitan los dispositivos IoT para los atacantes, la cantidad de información útil que proporcionan los sistemas trampa y las carencias que presentan las soluciones existentes, deben proponerse mejoras clave que pueden ayudar a hacer estos sistemas más eficaces para entornos IoT. Actualmente los dispositivos IoT son muy vulnerables a los ataques remotos, sin que haya herramientas realmente efectivas para el análisis de los ataques más avanzados contra éstos y las plataformas intermedias que los gestionan.

O3.1. **Proposición de una metodología:** Se ha de detallar la metodología que propone este trabajo para el despliegue de honeypots IoT, aportando su arquitectura y describiendo cada uno de sus componentes.

O3.2. **Validación:** Para validar la metodología es necesario mostrar la prueba de concepto del trabajo, una instanciación de la arquitectura en un entorno que demuestra la aplicación real del trabajo, aportando también los resultados obtenidos tras llevarla a cabo.

Tabla 1: Objetivos del trabajo y secciones donde se resuelven

Objetivo	Subobjetivo	Sección
O1	O1.1	2
	O1.2	4.1
O2	O2.1	4.2
	O2.2	5
	O2.3	5.4
O3	O3.1	6
	O3.2	7

4. Análisis y clasificación de dispositivos IoT

En esta sección se discutirán los métodos utilizados para elaborar tanto el análisis como la clasificación de los dispositivos IoT, especificando las fuentes de información de las que se han recogido los datos y los aspectos en los que se centran. Tras esto, se presenta un resumen de los dispositivos más populares y/o vulnerables, y se establece una clasificación.

4.1 Metodología

En el análisis se han realizado distintos tipos de búsquedas: tanto en buscadores de dispositivos IoT conectados a la red, para conocer su accesibilidad y popularidad, como en webs que anuncian dispositivos con vulnerabilidades.

Además, se llevaron a cabo búsquedas de dispositivos en sitios de compra online (Amazon, Network Webcams…) para conocer cuáles eran más populares y vendidos, y se buscaron vulnerabilidades para ellos.

Por último, a medida que se analizaban dispositivos, se han ido buscando y recogiendo versiones de su firmware, con objeto de una posible emulación posterior.

2.1.1 Buscadores IoT

- **SHODAN** [Shod2018]: Es un motor de búsqueda para dispositivos conectados a Internet, permitiendo filtrarlos de acuerdo a distintos criterios, gracias a sus *banners*[1]. Dispone de servicios como una API para realizar consultas; HoneyScore [Hone2018], un indicador de probabilidad de que un host indexado sea un honeypot; y Scanhub [Scan2018], que permite analizar escaneos personales con Nmap.
- **Censys** [Cens2018]: Este buscador realiza escaneos sobre todo el espacio de direcciones IPv4 con ZMap, un escáner de red. También dispone de una API y permite descargar los resultados en modo *raw* de la base de datos y un archivo JSON con los mismos. Las búsquedas permiten ver datos como puertos abiertos o subdominios a simple vista. También cuenta con filtros según protocolo o palabras en el *banner*. A diferencia de SHODAN, Censys es un buscador semántico.
- **Reposify** [Repo2018]: Se construyó con el objetivo de mejorar la seguridad en dispositivos conectados a internet, contando con una API que permite a las empresas de seguridad monitorizar los dispositivos de sus clientes para comprobar malas configuraciones (credenciales por defecto, firmware obsoleto, etc.).

[1] Mensaje de bienvenida que muestran los dispositivos IoT cuando un cliente intenta conectarse a ellos, posibilitando su reconocimiento, ya que en ocasiones indican la versión del firmware instalado o el modelo concreto.

- **Thingful** [Thin2018]: Consiste en un buscador geográfico de dispositivos IoT, los cuales se pueden clasificar en distintas categorías, atendiendo, de nuevo, a sus *banners*. Cuenta con la funcionalidad de realizar búsquedas por palabras clave que los dispositivos presenten o por localización geográfica.
- **Wigle** [Wigl2018]: Se trata de un buscador de redes mediante su SSID. Nos permite localizar algunos dispositivos IoT que usan redes de punto de acceso con nombres predeterminados.

2.1.2 Páginas que anuncian vulnerabilidades

- **Exploitee.rs** [Expl2018]: Wiki que se especializa en listar dispositivos IoT, entre otros. Detalla sus vulnerabilidades y cómo explotarlas, así como enlaces de compra online.
- **CVE Details** [Cved2018a]: Página web con datos sobre CVE (*Common Vulnerabilities and Exposures*, Vulnerabilidades y exposiciones comunes), detallando cada una de las vulnerabilidades con sus códigos de identificación, grados de peligro y facilidad de explotación, y algunas gráficas útiles (número de vulnerabilidades registradas por año y fabricante, por año y modelo, vulnerabilidades clasificadas por tipo para un modelo concreto…).
- **Exploit Database** [Exdb2018]: Web donde se detallan exploits para vulnerabilidades concretas, especificando su código CVE y en ocasiones incluyendo muestras de código descargable.

4.2 Clasificación

La clasificación se ha establecido en base a los ámbitos en los que los dispositivos IoT listados operan. Podrían surgir categorías nuevas más específicas en caso de que surjan nuevos dispositivos que no se encuentren englobados en las ya propuestas.

Se detalla en la lista el fabricante y el modelo del dispositivo, así como su ránking de ventas en ciertas webs (dentro de su tipo de dispositivos) y su número de apariciones en buscadores especializados en IoT. En caso de conocerse vulnerabilidades en cualquiera de las versiones de su firmware, será especificado en la columna correspondiente.

Tabla 2: Tabla de dispositivos IoT populares y/o vulnerables

Tipo de dispositivo	Modelo	Ranking página de ventas[2]	Ranking Buscadores[8]	¿Vulnerabilidad Conocida?
Routers	TP-Link TL-WR841N	1 (Amazon España, categoría: routers)	3 (Shodan, tags:router) – 47K resultados	Sí [Cved2012a] [Cved2012b] [Cved2012c]
	TP-Link TL-WR740N		2 (Shodan, tags: router) – 78K resultados	Sí [Cved2015a] [Cved2014a]
	TP-Link TL-WR741ND		5 (Shodan, tags: router) – 11K resultados	Sí [Cved2012a] [Cved2012b]
	Linksys E2500		2 (Shodan, tags: linksys) – 3K resultados	Sí [Cons2014] [Expl2016]
	Netgear WNR1000v3		3 (Shodan, tags: netgear) – 3K resultados	Sí [Expl2013a]
	Linksys E1500	12 (Amazon USA, categoría: routers)		Sí [Expl2013b]
	Zyxel gama AMG1302 y P-660HN			Sí [Sans2016] [Good2016] [Reve2016]
	Sagecom Livebox			Sí [Cved2017a]
	T-Mobile/ASUS AC1900	1 (Amazon USA, categoría: routers)		Sí [Cvode2014b]
	Netgear R6700	3 (Amazon USA, categoría: routers)		Sí [Cved2018b]
	TP-Link N540	5 (Amazon USA, categoría: routers)		No
	Linksys WRT54GL	2 (Amazon USA, categoría: routers)		Sí [Cved2009]
	Linksys E2500	4 (Amazon USA, categoría: routers)	2 (Shodan, tags: linksys) –1.8K resultados	No
	Linksys E4200	8 (Amazon USA, categoría: routers)	4 (Shodan, tags: linksys) – 500 resultados	Sí [Cved2014c]
Cámaras IP	DLink DCS-932L	17 (Amazon España, categoría: cámaras de seguridad)		Sí [Dlin2018]
	DLink DCS-930L			Sí [Dlin2018]
	DLink DCS-5300		Censys, tags: “dcs-5300” – 73 resultados	No
	Sony SNC-RZ25		Censys, tags: “snc-rz25” – 798 resultados	Sí [Cved2007a]
	Axis M1054		Censys, tags: “axis m1054” – 401 resultados	No
	Axis 2100		Censys, tags: “axis 2100” – 350 resultados	No
DVRs/NVRs	Sony NSR-500	5 (Network Webcams)	Censys, tags: “nsr-500” – 3 resultados	No
	Axis Companion Recorder	1 (Network Webcams)	Censys, tags: “axis companion” – 10 resultados	No
	Hikvision DS-7604/7608/7616	2 (Network Webcams)		No
	UniFi NVR		2 (Shodan, tags: nvr – 357 resultados)	No
	Zmodo NVR (ZMD-DT-SCN8)		5 (Shodan, tags: dvr – 450 resultados)	No
Antenas	Ubiquiti AirGrid M AG-HP-5G27		4 (Shodan, tags: router – 13K resultados)	No
Smart TV	TiVo Series2 Firmware		2 (Shodan, tags: dvr – 3’6K resultados)	No
Dispositivos industriales	Schneider Electric BMX P34 2020		1 (Shodan, tags: Schneider electric – 608 resultados)	Sí [Cved2011a]
	Schneider Electric BMX NOE 0100		2 (Shodan, tags: Schneider electric – 289 resultados)	Sí[17]

[2] Los datos relativos a las páginas de ventas y buscadores han sido recopilados a fecha 13 de febrero de 2018.

Tipo de dispositivo	Modelo	Ranking página de ventas[2]	Ranking Buscadores[8]	¿Vulnerabilidad Conocida?
	Schneider Electric SAS TSXETY4103		3 (Shodan, tags: Schneider electric – 169 resultados)	No
	Schneider Electric TM221CE40T		4 (Shodan, tags: Schneider electric – 117 resultados)	No
	Schneider Electric TM221CE40R		5 (Shodan, tags: Schneider electric – 60 resultados)	No
	Schneider Electric gama Modicon M340			Sí [Schn2015]
	Schneider Electric Homelynk (LSS100100)			Sí [Cved2017b]
	Schneider Electric TSEXTG3000			Sí [Cved2014d]
	Omron CJ2M		Censys, tags: "omron cj2m" – 4256 resultados Shodan, tags: "omron cj2m" – 521 resultados	Sí [Iscs2015]
Drones	DJI Phantom 3	9 (Amazon, categoría: hobby RC Quadcopters & Multirotors)		Sí [Yuan2015] [Phas2017] [Truj2016] [Stag2017] [Luo2016]
	Parrot AR 2.0			Sí [Szab2016]
	UDI U818A	8 (Amazon, categoría: hobby RC Quadcopters & Multirotors)		Sí [Foxb2017]
	Holy Stone S160	1 (Amazon, categoría: hobby RC Quadcopters & Multirotors)		No
Dispositivos domésticos	Tetera iKettle			Sí [Hugh2015] [Paul2015] [Munr2015a]
	Frigorífico Samsung RF28HMELBSR			Sí [Vend2015]
	Báscula FitBit Aria			Sí [Munr2015b]
	Lavavajillas Miele PG 8528			Sí [Cved2017c]
	Luces Philips Hue			Sí [Rone2016]
Otros	CloudPets			Sí [Sara2017]

4.3 Conclusiones

La mayoría de los dispositivos aquí presentados están a la venta en el mercado y cualquiera puede usarlos. Algunos de ellos son los más vendidos de su categoría, o recomendados por sitios tan grandes como Amazon. Teniendo en cuenta que muchos de ellos presentan vulnerabilidades en las versiones del firmware que traen de fábrica, y que el usuario medio no se preocupa de las actualizaciones o medidas de seguridad básicas, como cambiar la contraseña por defecto, el escenario resultante es muy preocupante.

Hay que indicar también que, aunque se trate de dispositivos operados por algún administrador que se ocupe de instalar las actualizaciones pertinentes, muchos fabricantes las lanzan demasiado tarde, siendo los detalles de las vulnerabilidades conocidos para entonces, y pudiendo haberse originado y sufrido ataques.

El amplio uso de estos dispositivos se puede corroborar realizando búsquedas en los buscadores IoT. Se puede comprobar que algunos modelos llegan fácilmente a los cientos y miles en número, y en algunos casos, hasta decenas de miles de dispositivos conectados, y por tanto, expuestos de manera directa.

Esto hace que el panorama sea especialmente suculento para el atacante: miles de dispositivos vulnerables directamente conectados a internet, con usuarios sin conocimientos especializados siendo inconscientes de lo expuestos que están por la falta de seguridad.

En cuanto al firmware recogido, excepto en ocasiones concretas, ha sido difícil de encontrar las versiones con vulnerabilidades, puesto que los propios fabricantes han procedido a su retirada de las webs oficiales, aunque algunos aún se podían encontrar en repositorios online. Disponiendo de este firmware, se podrían emular los dispositivos, haciendo las veces de honeypot, pero con una interacción total, ya que se puede reproducir su funcionamiento completo.

5. Análisis y clasificación de honeypots IoT

En esta sección se llevará a cabo un análisis sobre las soluciones de honeypots IoT existentes y los trabajos realizados sobre ellas. Asimismo, se categorizan los honeypots atendiendo a diferentes criterios.

5.1 Tipos de Honeypots

Los honeypots se pueden clasificar en base a diferentes criterios, que nos permiten saber detalles sobre su comportamiento o estructura real.

Se pueden clasificar en base a la interacción que tengan con las conexiones entrantes, esto los divide en:

- **Baja interacción**: son honeypots que tienen respuesta básica a ciertos comandos pero no van más allá, reproducen una parte limitada del comportamiento real del sistema. Un ejemplo sería un honeypot que simplemente presente un prompt de login para registrar qué usuarios y contraseñas prueban los atacantes como entrada. Su configuración es sencilla y su consumo de recursos bajo, pero es sencillo notar que no se trata de un dispositivo real.
- **Media interacción**: en estos honeypots crece la relación entre el atacante y el sistema, llegando a colocar recursos falsos, como sistemas de ficheros, servidores de FTP o SSH. Pueden ser útiles para capturar comandos que puedan ejecutar los atacantes, ver qué vulnerabilidades intentan explotar o qué malware intentan ejecutar.

- **Alta interacción:** ofrecen un sistema con las características que el dispositivo real, con acceso total al atacante, ya que las vulnerabilidades son completamente explotables. Se pueden recopilar más datos, pero también suponen un riesgo mayor, ya que el sistema se puede ver comprometido.

Atendiendo a cómo está realizada la implementación de los honeypots se pueden considerar las siguientes categorías:

- **Simulados:** son aquellos honeypots cuyo comportamiento imita al de un dispositivo real mediante recursos falsos, salidas preprogramadas o patrones definidos.
- **Emulados:** cuentan con una plataforma que les proporciona una infraestructura virtual para que el honeypot corra sobre ella, la cual es igual que la del dispositivo real. En ocasiones se usan como sandbox, pudiendo ser fácilmente reseteados a un estado anterior si un atacante logra modificar partes importantes del sistema.
- **Dispositivos reales:** se usa el hardware real como honeypot, estando disponible al atacante el entorno al completo.

5.2 Motivación

Los honeypots son una herramienta especialmente útil para detectar, analizar y mitigar amenazas y ataques en este entorno. En primer lugar, al simular ser sistemas reales, atraen la atención de los atacantes, especialmente si se trata de software automatizado que no realiza comprobaciones o distinciones. Además, las consecuencias de los ataques, si son exitosos, y logran comprometer el sistema trampa, son nimias, puesto que no hay ninguna infraestructura real tras ellos, y se pueden ejecutar de manera aislada, sin que afecten al resto del entorno.

Los honeypots se presentan como víctimas reales para el atacante, y dependiendo de su tipo, pueden ser más o menos realistas. La interacción que tenga lugar entre el atacante y el honeypot nos permitirá recopilar datos de su comportamiento (comandos que intente ejecutar, frecuencia de conexión, localización de la misma…) para poder comprender mejor su funcionamiento y poder evitar y mitigar futuros ataques.

También hay que tener en cuenta que simular un dispositivo IoT puede ser sencillo si se trata de un sistema empotrado con un firmware simple, que tenga respuesta a ciertos comandos básicos o simule un sistema de ficheros, o muy complicado, si el dispositivo en cuestión tiene una fuerte dependencia del hardware (simular el funcionamiento de cámaras, por ejemplo, es muy costoso).

Por estos motivos, el uso de honeypots como mecanismo de análisis de ataques a dispositivos IoT es una herramienta potente, flexible, y favorable, como se puede deducir por su uso en el campo, y por tanto, serán el fundamento de este trabajo presentado.

5.3 Estado de la técnica

El ámbito de honeypots específicamente dedicados a dispositivos IoT es muy reciente, por lo que la cantidad de trabajo hecho al respecto no es muy extensa. Dados los ataques de *DDoS* (*Distributed Denial of Service*, que últimamente proceden de botnets compuestas de dispositivos IoT infectados [Angrishi2017], y la dificultad que supone remediar su vulnerabilidad en muchos casos, los honeypots IoT han despertado el interés en diversos sectores. Desde honeypots de uso cotidiano (p.ej. HoneyDroid), hasta el sector industrial (p.ej. Conpot) se hacen eco de la relevancia de estos mecanismos para capturar y analizar amenazas que afectan a los dispositivos IoT, cada vez más arraigados en nuestro día a día.

Cabe destacar que, si bien hay una amplia diversidad de artículos que anuncian propuestas honeypots IoT, no todas las soluciones resultan ser adecuadas para entornos IoT.

5.3.1 Arquitecturas honeypot

Algunos de los trabajos relacionados son más que honeypots en sí. Se trata de arquitecturas de despliegue de honeypots con varios componentes, que en ocasiones hacen uso de otras soluciones o plataformas. Se listarán estos en primer lugar, ya que en concepto se asemejan más a la metodología propuesta en este trabajo.

En el caso de IoTPOT [Guar2017], los honeypots usan emulaciones de dispositivos reales (mediante QEMU). IoTPOT dispone de un conjunto de honeypots de baja interacción, que tienen la respuesta a ciertos comandos que pueden ejecutar las conexiones entrantes. Si algún comando se desconoce, se redirige a una emulación del dispositivo, donde se ejecuta realmente y se devuelve la salida. Esta salida se guarda para futuras respuestas, enriqueciendo al honeypot de baja interacción, mientras que la emulación se puede resetear para que no se propague el efecto que puedan tener los comandos ejecutados y tener el sistema a modo de sandbox. Así se tiene un sistema con la misma respuesta de un honeypot de alta interacción sin los riesgos que supone uno, aunque con el coste de su emulación.

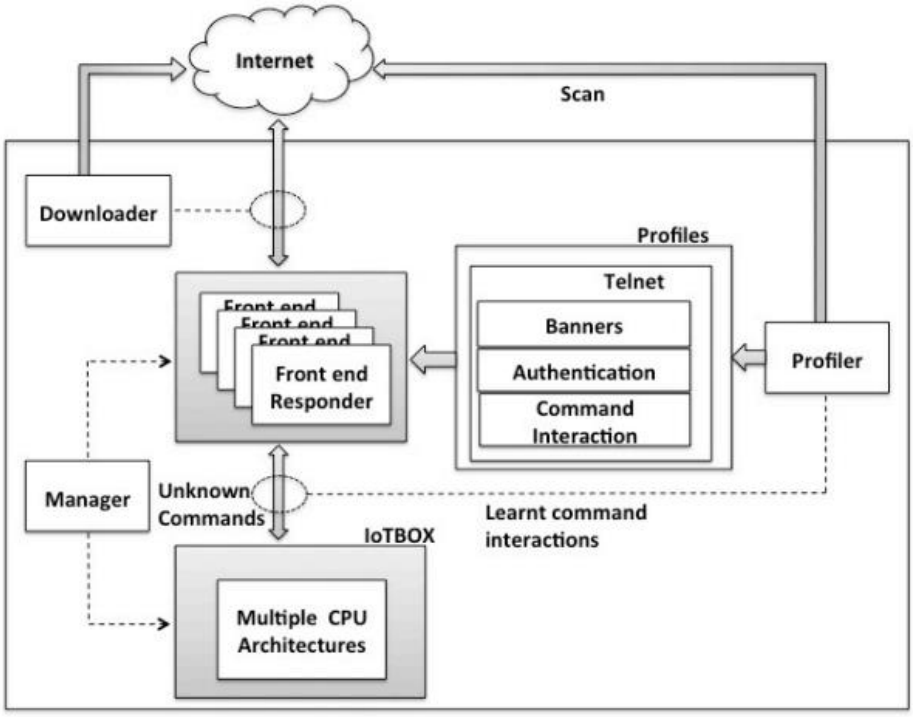

Figura 4: Arquitectura de IoTPot [Guar2017]

Otra característica interesante de IoTPOT es que cuando recibe una conexión, se intenta conectar al dispositivo del que la ha recibido, ya que considera que podría ser otro dispositivo IoT infectado intentando propagarse. De esta manera guarda su banner para tener más respuestas posibles que ofrecer a las próximas conexiones.

Sin embargo, IoTPOT tiene varias características que se podrían mejorar. En primer lugar, se ciñe exclusivamente a conexiones por Telnet. Si bien la mayoría de dispositivos IoT que han sido víctimas de botnets han sido compometidos a través de conexiones por este puerto, hay muchos sistemas que intentan conectarse por SSH o HTTP, además de vulnerabilidades presentes en estos protocolos. Por último, aunque la arquitectura que se ha seguido se detalla claramente, contando incluso con el firmware usado en las máquinas emuladas (OpenWrt y Debian para distintas arquitecturas de procesadores), el código que se ha utilizado en los honeypots de baja interacción para capturar banners y redirigir comandos a los sistemas emulados no está disponible, por lo que no es trivial reproducir su comportamiento.

En el caso de SIPHON [Pa2015], el sistema que se presenta sí que cuenta con dispositivos reales, aunque la arquitectura de despliegue es muy distinta.

SIPHON tiene dos objetivos principales: demostrar que la localización geográfica de los dispositivos es relevante a la hora de los ataques que pueda sufrir, y tener un sistema que

permita hacer parecer que hay un alto número de dispositivos disponibles con una cantidad limitada de aparatos reales.

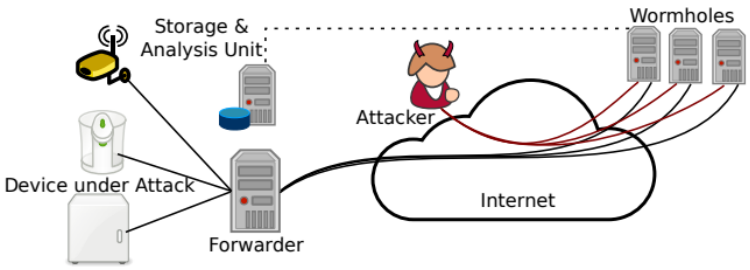

Figura 5: Arquitectura de SIPHON [Pa2017]

La arquitectura de SIPHON se basa en varios servidores Cloud con un servicio de fowarding que redirigen los paquetes o peticiones hacia un conjunto de dispositivos reales en un entorno controlado (routers, cámaras IP, DVRs…). Esto permite obtener respuestas sobre dispositivos reales, teniendo así un servicio no sólo de alta interacción, sino distribuido a ojos de los posibles atacantes, en el que además hay aparentemente más máquinas de las que realmente se encuentran en un funcionamiento.

Una de las ventajas que presenta SIPHON es que la redirección no tiene por qué hacerse a un dispositivo real, sino que puede ser también a una simulación o emulación, por lo que no es estrictamente necesario contar con aparatos conectados. Es decir, una vez el atacante ha descubierto la IP a la que intenta establecer una conexión, esperará una respuesta, que puede ser de un dispositivo real o de una máquina que imite su funcionamiento, según se haya dirigido la IP del servidor Cloud al aparato en sí o a dicha máquina.

Un riesgo de la arquitectura de SIPHON es que, al contar con dispositivos reales, además de su coste, éstos pueden ser comprometidos si un atacante consigue escalar los suficientes privilegios, lo cual podría tener repercusiones y extender aún más algunos ataques. Para prevenir esto, los aparatos tendrían que ser reseteados cada cierto tiempo o contar con un Intrusion Prevention System (IPS), lo cual añade complejidad. Otra desventaja es el coste de despliegue de la arquitectura, ya que el alquiler de varios servidores Cloud en distintas localizaciones geográficas supone una inversión que hay que asumir, al igual que adquirir los dispositivos reales. Además, el tipo de honeypots depende de la existencia de su equivalente dispositivo real.

Un inconveniente que presentan ambas soluciones descritas es que no consideran IPv6, que como se ha visto recientemente, también está siendo objetivo de ataques, y recientemente hay

botnets de IoT que intentan explotarlo. El espacio de direcciones de IPv6 es mucho mayor que el de IPv4, y las herramientas para escanearlo que utilizan las botnets (Nmap y ZMap) no serían capaces de explorarlo en un tiempo razonable, por lo que el despliegue de honeypots ha de hacerse de manera distinta, ocupando varias direcciones y viendo los patrones que siguen los escaneos. Ya hay algunos trabajos realizados en este ámbito [Schindler2015], aunque no específicos para IoT.

Otras soluciones, como ThingPot [Wang2017] han explorado nuevos escenarios, como los dispositivos IoT configurables por nuevos medios, tales como los mensajes XMPP a través de un servidor que implemente este protocolo de mensajería instantánea. En concreto, mediante la emulación de unas luces Philips Hue, permite dos entradas: la configuración a través de una API REST y a través de los mensajes mencionados. Obtienen mediante esta prueba de concepto, diversos perfiles de ataque, pero todos ellos a través de la API. Este trabajo se enfoca mayormente en mostrar cómo el acceso mediante XMPP podría mejorar la seguridad, ya que no hay interés por parte de los atacantes, ni una tendencia clara en explotar las posibles vulnerabilidades de este protocolo. Esta conclusión no tiene por qué ser cierta, ya que el uso de un protocolo menos extendido (y que por tanto, suscite menos interés a la hora de ser víctima de ataques) no significa que se tenga más seguridad, ya que es una forma de la desaconsejada *seguridad por oscuridad*.

Por otra parte, IoTCandyJar [Luo2017] se postula como un honeypot de *interacción inteligente*, un término que usan para describir la manera de interactuar del sistema, basada en recabar respuestas de dispositivos existentes.

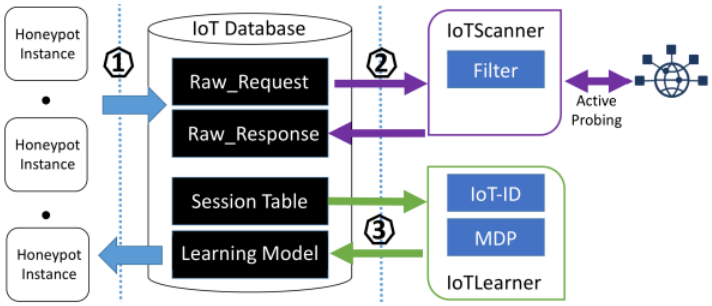

Figura 6: Arquitectura de IoTCandyJar [Luo2017]

Esto se lleva a cabo situando un honeypot para que atraiga la atención de los atacantes. Cuando IoTCandyJar recibe un mensaje, lo redirige a un sistema que lo envía a una gran cantidad de dispositivos IoT expuestos en la red, recogiendo sus respuestas. Guarda las respuestas en una base de datos y de todas ellas, escoge la más indicada y la devuelve, para que el honeypot

interactúe así con el atacante. Si éste vuelve a enviar un mensaje, se repite el proceso. La elección de la respuesta se hace guardando en cada caso la que tiene un número de interacciones más alto, esto es, si a la respuesta del sistema, el atacante vuelve a realizar una petición, se almacena esta continuación de la comunicación para posteriores usos.

IoTCandyJar cuenta también con un sistema de filtro que limpia los mensajes maliciosos que envía el atacante, y no redirige estos a dispositivos IoT reales, para no correr el riesgo de infectarlos y propagar un ataque.

También se encuentra el trabajo presentado por Krishnaprasad P [Kris2017], que toma como base varios de los trabajos vistos anteriormente, ofreciendo varias mejoras respecto a sus diseños y funcionamiento. Este honeypot coger la arquitectura de IoTPOT y extiende su soporte a más protocolos, como SSH, HTTP y TR-064. Otra de las ventajas que ofrece frente a éste es que su código es libre, además de estar disponible como *Docker* (un contenedor de aplicaciones con una tecnología similar a la virtualización) y ser ligeras, tanto en peso como en consumo de recursos.

Este honeypot sigue el mismo funcionamiento descrito en IoTPOT: cuenta con varios honeypot de baja interacción, que si desconocen la respuesta a un comando, redirigen la petición a uno de alta interacción, el cual se encuentra en un sandbox para evitar tanto verse afectado, como propagar el ataque, además de facilitar su recuperación a un estado anterior si surge algún problema. Este honeypot de alta interacción al que se redirigen las peticiones se basa en QEMU para simular firmware OpenWrt (firmware para routers también usado en IoTPOT) a la hora de atender peticiones Telnet, SSH y HTTP. En cuanto al protocolo TR-064, hace uso de una solución que se comenta en secciones posteriores (5.2.2) en este trabajo: HonyeThing. Además, mediante el stack ELK, propone una organización centralizada de los logs de cada honeypot, haciéndola escalable y eficiente.

5.3.2 Honeypots específicos

Recientemente ha habido nuevos trabajos en esta área. Algunos han tomado como base los presentados anteriormente para desarrollarse, mientras que otros se han fijado en las tendencias de los ataques IoT y se han centrado en problemáticas específicas.

Entre estos últimos se encuentra HoneyThing [Chad2017], que se basa en la vulnerabilidad del protocolo TR-064, que afecta a ciertos modelos de routers. Algunas versiones de la botnet Mirai actuaron sobre implementaciones vulnerables de dicho protocolo, que fue pensado para configurar, gestionar o monitorizar dispositivos de red (routers, switches, dispositivos VoIP) de manera remota, normalmente en una LAN, evitando tener que enviar técnicos y haciendo las

gestiones mediante un “servidor de auto-configuración”. TR-064 fue deprecado, aunque un protocolo relacionado, TR-069, se usa para configuración mediante redes WAN.

Estos protocolos escuchan en el puerto 7547 (y algunos en el 5555), y algunos routers de marcas tales como Eir o DSL implementaban ambos servicios. Esto dejaba la puerta abierta a que se configuren estos routers de manera remota y poco segura, ya que TR-064 no está pensado para funcionar en WAN, y por tanto no cuenta con las medidas de seguridad suficientes. Explotando esta vulnerabilidad se han llegado a ejecutar comandos arbitrarios que podían hacer obtener las contraseñas de la red WiFi, de la interfaz de administración, o modificar la configuración del firewall [Reve2016].

Al observarse una subida del tráfico en los puertos 7547 y 5555 debido a las consultas generadas por ataques de distintas versiones de Mirai, HoneyThing fue diseñado como un honeypot de baja interacción que simula un router presentando esta vulnerabilidad.

Algunos de los trabajos se centran en escenarios concretos, tales como las PAN (*Personal Area Network*, red de área personal) y los dispositivos IoT que funcionan en este entorno mediante un protocolo concreto [Dowl2017]. El protocolo estudiado ha sido ZigBee, usado en WSN (*Wireless Sensor Networks*, redes inalámbricas de sensores), especialmente en domótica. Dispositivos como las luces Philips Hue vistas anteriormente, se pueden comunicar mediante este protocolo.

Otros trabajos relacionados proponen sistemas para modelar teóricamente un escenario de ataque a un dispositivo IoT con distintas estrategias usando teoría de juegos [DuyLa2016], o bien realizan un análisis de las soluciones disponibles [Nawrocki2016]. Además de trabajos de investigación y artículos, hay proyectos que están usando estas tecnologías, como es iHoney, que desarrolla herramientas para la ciberseguridad en entornos industriales (análisis del tráfico, prevención y detención de intrusiones).

También cabe destacar la existencia de proyectos más simples, que pretenden simular un dispositivo IoT completo, tales como cámaras, implementando movimientos, rotaciones y zoom. Algunos de los que implementan esta funcionalidad son Wificam [Brit2017] y Honeypot Camera [Bred2015]. Otros se basan en ataques o vulnerabilidades específicos, como MTPot, diseñado especialmente para Mirai [Gold2016].

5.4 Comparativa de honeypots IoT

Tabla 3: Soluciones de honeypots específicos para IoT

Honeypot	Ámbito	Interacción	Protocolo ataques	Arquitectura	Otras	Dependiente HW	Descargable	Mantenimiento[3]
IoTPOT	Industrial, doméstico, personal	Alta	Telnet	**Componentes:** - Downloader - Profiler - Manager (envía comandos desde el frontend responder "dummy" a una arquitectura real) **Fases:** - Intrusión - Infección - Monetización (se ejecutan binarios procedentes de servidores de C&C para sacar provecho de los dispositivos)	• Diferentes tácticas de login (aceptar cualquiera, no aceptar ninguno, aceptar tras varios intentos, aceptar credenciales concretos) • Emula arquitecturas en QEMU y cuenta con OpenWrt (firmware para routers basado en Linux) • Escaneo automático de las IPs atacantes para obtener información sobre ellas ("banner messages") • Emula arquitecturas MIPS, ARM y PPC.	No	No	Artículo: 2015 [Pa2015]
Conpot	Industrial	Media	Modbus (TCP), SNMP	Corre en un máquina emulando un firmware, pero hay trabajos en los que se han desplegado en varios servidores cloud [Jich2016].	• Simula sistemas industriales (ICS: Industrial Control System), incluyendo PLCs, sistemas de control de gasolineras… • Cuenta con simulación de protocolos como Modbus o SNMP. • Se puede usar para redes SCADA, BACNets (control de edificios, alarmas, sensores, calefacción…), sistemas HVAC (Heating, ventilating and cooling), etc.	No	Sí [Conp2014]	Artículo: Junio 2014 [Buza2014], Septiembre 2016 [Jich2016]
SIPHON	Profesional, personal	Alta	SSH, HTTP	**Componentes:** - Wormholes - Forwarders	• Trendnet Camera Emulator para simular una cámara IP, el resto son dispostivos reales.	Sí	No	Artículo: Enero 2017 [Guar2017]

[3] Fechas de últimos *commits* (aportaciones o actualizaciones de código) recogidas el 16/02/2018.

Honeypot	Ámbito	Interacción	Protocolo ataques	Arquitectura	Otras	Dependiente HW	Descargable	Mantenimiento[3]
				- Unidad de almacenamiento y análisis **Fases:** - Implementación de honeypots - Puesta a punto de forwarders - Localizaciones para los wormholes - Esconder la naturaleza de honeypot	• User/pass definidos con distintos grados de dificultad para estudiar interacción de atacantes. • Servidores cloud geolocalizados en distintas partes del mundo que están enrutados a varios dispositivos IoT reales en un laboratorio (a ojos de los atacantes hay muchos más dispositivos de los que realmente hay). • También se probó con dispositivos simulados (cámaras con imagen fija) y de baja interacción.			
Honeypot cámara	Profesional, personal (específico cámara)	Baja	HTTP		• Simula el comportamiento de una cámara real mediante el movimiento de una imagen panorámica.	No	Sí [Bred2015]	Último commit: Junio 2015
IoTCandyJar	Profesional, personal	*Inteligente*	Muchos (HTTP, SSH, Telnet, TR-064, XMPP, MQTT, UPnP, CoAP, MS-RDP…)	**Componentes:** - Honeypot - IoT Database - IoTScanner - IoTLearner **Fase continua:** - Honeypot recibe petición. - IoTScanner consulta dispositivos existentes. - IoTLearner decide la major respuesta. - IoT Database devuelve al honeypot la major respuesta.	• Decide qué respuesta es la óptima mediante una función de recompensa que tiene en cuenta el número de respuestas del atacante. • Las posibles respuestas las recoge de dispositivos IoT reales a los que envía las peticiones. • Consta de un filtro para no propagar infecciones a dispositivos IoT enviando comandos que pudieran ser maliciosos.	No	No	Artículo: Julio 2017 [Luo2017]
ThingPot	Profesional, personal	Media	XMPP, HTTP	**Componentes:** - Honeypot (punto de entrada) - Servidor XMPP - API REST - Dispositivo IoT emulado	• Se centra en el protocol XMPP. • Simula unas luces Philips Hue.	No	Sí [Wang2017b]	Artículo: Noviembre 2017 [Wang2017a] Último commit: Agosto 2017

Honeypot	Ámbito	Interacción	Protocolo ataques	Arquitectura	Otras	Dependiente HW	Descargable	Mantenimiento[3]
HoneyThing	Específico routers	Baja	TR-064		• Emulación de routers con vulnerabilidad en el Puerto 7547 por soporte WAN del protocol TR-064.	No	Sí [Erde2016]	Último commit: Marzo 2016
ZigBee Honeypot	Personal	Media	ZigBee		• Versión modificada de Kippo soportando el protocol ZigBee, alojada en una máquina AWS. • Se centra en Personal Area Networks.	No	No	Artículo: Junio 2017 [Dowl2017]
Multi-purpose IoT Honeypot	Profesional, personal	Alta	HTTP, SSH, TR-064, Telnet	**Componentes:** - Honeypots "dummy" de baja interacción - Dispositivos OpenWrt emulados con QEMU - Dispositivos emulados con HoneyThing - Manager **Fases:** - Recepción de commandos - Consulta en lista de respuestas - Consulta a dispositivo emulado (opcional)	• Versión open-source y mejorada de IoTPOT, soportando más protocolos. • Usa algunas soluciones de esta table, como HoneyThing.	No	Sí [Kris2017b]	Artículo: 2017 [Kris2017a] Último commit: Mayo 2017
MTPot	Específico Mirai	Baja	SSH (subidas SCP y SFTP)		• Específico para Mirai, que afectó a dispositivos IoT	No	Sí [Gold2016]	Último commit: Noviembre 2016
Wificam	Profesional, personal (específico cámaras)	Baja	HTTP		• Emula una cámara IP con una serie de vulnerabilidades en su servidor HTTP. • No cuenta con simulación de imagen.	No	Sí [Brit2017]	Último commit: Abril 2017

Tabla 4: Resumen otros honeypots

Honeypot	Ámbito	Interacción	Características relevantes		Viabilidad de emulación		
			Protocolo ataques	Otras	Dependiente HW	Descargable	Mantenimiento[3]
Dionaea	Profesional	Baja	SMB (también de manera básica HTTP, HTTPS, FTP, TFTP, SIP, MSSQL). Soporta IPv6.	• Fork de Nepenthes. • Soporta IPv6. • Usa libemu para encontrar el código de shell en el payload de los paquetes de los atacantes y obtener una copia.	No	Sí [Schl2014]	Último commit: Junio 2014
Scriptgen / SGNET	Profesional	Media	Independiente	• Máquinas de estado finito para aprender el comportamiento de cualquier protocol • Si una máquina de estado finito no sabe cómo responder a un protocolo, se le pasa a una máquina real usada como un sandbox, haciendo el honeypot de baja interacción de proxy. Los resultados se comparten en SGNET para aprendizaje mutuo.	Sí	No	Artículo: Febrero 2007 [Leit2007]
Honeyd	Profesional	Baja / Media (si se usa como proxy)	Telnet	• Permite que un solo host tenga varias direcciones y simulación de topologías de red completas. • Usado como base para otros honeypots.	No	Sí [Prov2007]	Última versión: Mayo 2007
Cowrie	Profesional	Media	SSH (subidas SFTP y SCP)	• Login muy flexible (aceptar cualquier usuario, aceptar un usuario y cualquier contraseña, aceptar cualquier usuario y una contraseña, aceptar credenciales concretos, aceptar tras X intentos, aceptar en un rango de X a Y intentos…) • Simula sistemas de ficheros completos y directorios como /etc y /proc. • Hace log de todas las interacciones de los atacantes y guarda copias de lo que haya sido descargado con wget.	No	Sí [Oost2018]	Último commit: Febrero 2018

Honeypot	Ámbito	Interacción	Características relevantes		Viabilidad de emulación		
			Protocolo ataques	Otras	Dependiente HW	Descargable	Mantenimiento[3]
Hyhoneydv6	Profesional, académico	Alta	IPv6 (ICMP)	• Honeypots de baja interacción que redirigen las respuestas desconocidas a honeypots de alta interacción en máquinas virtuales, las cuales se crean bajo demanda. Los honeypots de baja interacción se crean en base a una distribución de probabilidad de las direcciones que usarán los atacantes.	No	No	Artículo: Junio 2015 [Schi2015]
Shockpot	Académico, profesional	Baja	HTTP (servicio apache)	• Diseñado para encontrar atacantes basándose en una vulnerabilidad grave concreta (Shellshock / CVE-2014-6271).	No	Sí [Tros2015]	Último commit: Diciembre 2015
Honeydroid	Personal, profesional	Alta	Varios (todos los soportados por Android)	• Cuenta con un microkernel aislado del sistema que permite correr las aplicaciones de logs de manera que no se puedan ver afectadas por posibles ataques. • Se ejecuta sobre hardware real para darle funciones de red móvil, pero podría ejecutarse sobre un emulador (QEMU).	No	No	Aún en desarrollo.

5.5 Conclusiones

Las soluciones específicas de honeypots para IoT propuestas hasta la fecha presentan aspectos a mejorar, comentados en la subsección anterior. No obstante, cada vez van surgiendo nuevos artículos más exhaustivos, debido al auge de estas tecnologías. Siendo IoT un escenario cada vez más relevante, es de esperar que aparezcan propuestas novedosas que suplan las carencias de las ya presentes.

También se puede ver cómo las soluciones discutidas tienen en cuenta el coste de la simulación y el mantenimiento de los sistemas, buscando soluciones intermedias que permitan tener un honeypot con una interacción lo suficientemente alta para engañar a posibles atacantes, pero sin contar con el coste que supondría tener el sistema real corriendo y con las respuestas necesarias a cada comando.

También se aprecia cómo tienen en cuenta factores tales como la localización geográfica o el dispositivo desde el que se origina el ataque. Esto permite establecer perfiles de atacantes, sabiendo su procedencia y los dispositivos que prefieren infectar, que tienen mayor facilidad para ello, o simplemente están más extendidos.

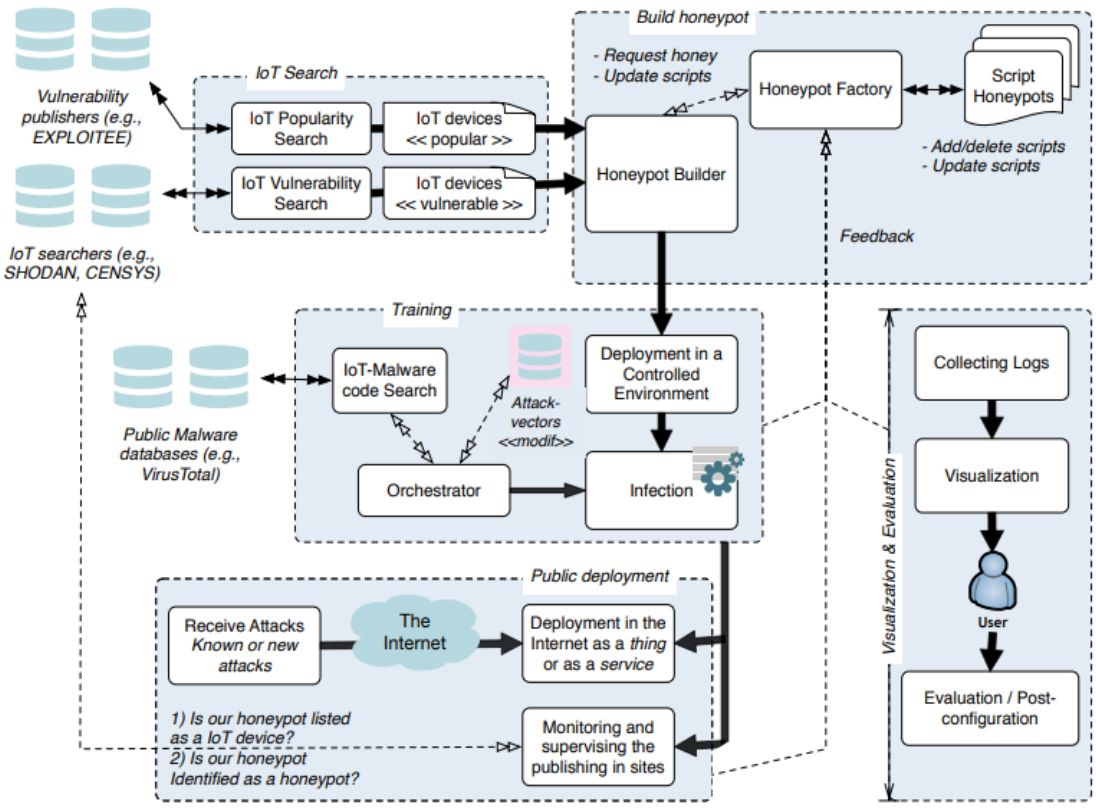

Figura 7: Metodología propuesta

Como se puede apreciar, un honeypot que tuviera en cuenta todas las posibilidades (interacción alta, sistema IPS, compatibilidad con IPv6, cobertura de varios protocolos…) sería costoso y requeriría de una gran coordinación entre las partes que lo compondrían.

6. Metodología propuesta

Para lograr los objetivos se propone una metodología de trabajo con varias partes diferenciadas, las cuales se pasan a describir a continuación. Se puede observar una visión general de la misma en la Figura 7, con los distintos bloques y pasos que se siguen.

6.1 Búsqueda IoT

Los primeros desafíos abordados son hacer el honeypot atractivo para los atacantes, y hacer que el conjunto de honeypots desplegado sea representativo del contexto IoT. Ambos puntos son importantes y tienen ciertas dificultades. En el primero de ellos se ha de encontrar un compromiso: si se trata un honeypot de baja interacción, o que no represente un dispositivo IoT *atractivo*, los atacantes pueden darse cuenta de que es un honeypot fácilmente. Sin embargo, si se cuenta con honeypots de alta interacción, emulaciones de sistemas completos, o dispositivos reales, el coste del sistema sería muy alto. En el segundo está la problemática de la heterogeneidad del escenario IoT, ya que los dispositivos son muy distintos entre sí (cámaras, routers, *wearables*…) y tienen respuestas diversas a los posibles comandos de los atacantes. Un trabajo que no satisfaga estas condiciones tiene el riesgo de ser demasiado específico, o quedar rápidamente obsoleto, como se ha visto en el análisis de las soluciones honeypot IoT existentes en la Sección 5.

Los métodos usados en el análisis, tales como usar motores de búsqueda especializados, páginas de venta populares y webs especializadas en malware y vulnerabilidades, se han descrito en la Subsección 5.1, y los resultados de llevarlo a cabo, en las posteriores.

6.2 Construcción de honeypots

Este paso en la metodología se centra en la elaboración de honeypots usando la información recabada en la fase anterior: dispositivos IoT populares, y dispositivos IoT vulnerables, coincidiendo en ocasiones ambos aspectos, como se puede ver en la Tabla 1.

Aunque la implementación de honeypots puede depender de muchos factores (su tipo, su dependencia del hardware…), es importante mantener un repositorio de soluciones ya implementadas, para aumentar la eficiencia. Esto podría ahorrar mucho tiempo si, por ejemplo,

se tuviera que implementar un honeypot de características similares a uno ya existente, pero con un firmware distinto que corrija vulnerabilidades.

Para clasificar y catalogar estas soluciones, se recomienda usar una *Honeypot Factory* (Fábrica de honeypots) durante esta fase. Este componente se usará para controlar el acceso y las modificaciones a los archivos que describan la configuración de los honeypots. A su vez, el componente recibirá feedback para clasificar mejor los honeypots y mejorar el Honeypot Builder.

En este trabajo, la infección de los sistemas se lleva a cabo de forma manual, por lo que tanto esta fase como la anterior no se ponen en funcionamiento. Sin embargo, son necesarias para que al desarrollar honeypots en el futuro siguiendo esta metodología, éstos cubran todo el entorno IoT y sean útiles.

6.3 Entrenamiento

Durante esta fase, se lleva a cabo el despliegue de los honeypots en un entorno controlado para probarlo. Para ello, el honeypot se infecta usando malware conocido. En algunos casos, infectar un dispositivo no es en absoluto trivial. Uno de los propósitos de este paso en la metodología es evaluar la viabilidad de un honeypot para ser infectado. Hay que destacar que, en algunos casos, es deseable que el honeypot sea lo más vulnerable posible (por ejemplo, para recabar información sobre ataques automatizados u oportunistas), pero en otros debería ser un desafío para el atacante, ya que el honeypot si no presenta unos niveles de seguridad mínimos, las sospechas sobre estar atacando a un sistema trampa crecen.

6.4 Despliegue público

El *feedback* más útil al sistema implementado mediante esta metodología se obtiene de esta fase. Durante estos pasos, el honeypot se despliega expuesto directamente a internet, disponible para los atacantes.

Uno de los problemas a los que se puede enfrentar el honeypot desplegado es pasar desapercibido frente a los atacantes, debido a la alta densidad de aparatos IoT disponibles. Y lo que podría ser peor, si el honeypot es identificado como tal por el atacante, éste podría difundir la información a otros atacantes.

Por tanto, hay que tener en cuenta que se necesita *promocionar* el sistema trampa para hacer que los motores de búsqueda IoT indexen los honeypots desplegados. Estos sistemas deben ser comprobados a fin de detectar si se ha llevado a cabo correctamente esta indexación, por lo que se requiere una monitorización constante durante esta fase.

Por último, los sistemas deben ser realistas. La complejidad de este punto es crucial, puesto que si, por ejemplo, se despliega un honeypot de un dispositivo industrial en un sistema SCADA, para imitar correctamente su funcionamiento, habría que implementar intercambios de comandos con controladores que, en muchas ocasiones, siguen protocolos propietarios, lo cual sería muy costoso de llevar a cabo a la perfección.

6.5 Visualización y evaluación

Esta fase es paralela a las de entrenamiento y despliegue público. Sin embargo, los resultados de ambas fases tienen que estar claramente separados y clasificados. Esto nos facilitará saber si los resultados esperados antes de realizar el despliegue se corresponden con los reales obtenidos más tarde.

En particular, esta fase está pensada para hacer que la solución sea usable por un administrador o investigador que use el sistema y pueda observar los resultados de manera clara y sencilla.

Se ha de enfatizar que los logs pueden ser recogidos en diferentes formatos y esto puede representar un problema a la hora de la visualización, por lo que la traducción entre diferentes formatos se tiene en cuenta en esta fase.

La metodología también considera el *feedback* que provee el usuario para mejorar y configurar las plataformas de los honeypots, asegurando así su mantenimiento.

7. Validación

Las pruebas realizadas se ejecutarán en la plataforma VMWare vCloud. Esta sección describe el progreso de la preparación del entorno, así como las pruebas realizadas, los objetivos, dificultades y lecciones aprendidas.

7.1 Descripción del entorno

La plataforma vCloud se ajusta bien a los objetivos del trabajo porque proporciona un entorno que se puede aislar, sin posibilidad de que se propaguen ataques al exterior y afecten a máquinas en producción en la fase descrita en la Subsección 6.3. Además, la herramienta de *snapshots*, que permite tomar instantáneas de las máquinas virtuales en cualquier momento y poder restablecer las configuraciones y estados anteriores, resulta muy útil en caso de que alguna de las máquinas quede afectada por una infección.

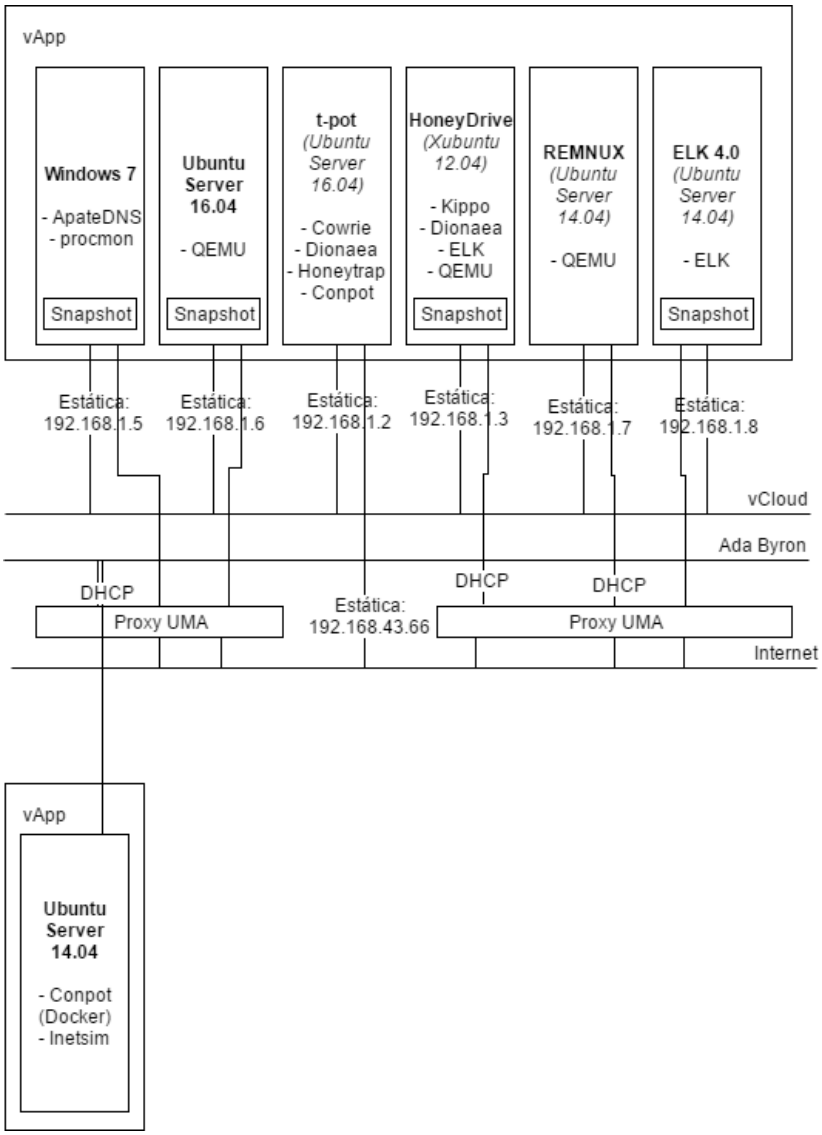

Figura 8: vApps del entorno vCloud donde se realiza el despliegue (máquinas y redes)

Asimismo, la naturaleza versátil de las máquinas virtuales es de gran ayuda, puesto que ofrece la posibilidad de cambiar características, como las conexiones de red, la capacidad de procesamiento, o la memoria, sin un cambio costoso de llevar a cabo en equipos físicos.

Algunas máquinas, como se puede ver en la Tabla 4, están destinadas a la centralización de *logs* (registros) para poder procesarlos y obtener una visión conjunta de lo que sucede en el entorno (cumpliendo la función de la fase de visualización y evaluación de la Subsección 6.5), mientras que otras están destinadas a recibir malware y ver cómo afecta la infección en su sistema y se propaga al resto, o tienen honeypots para recoger activamente datos de los ataques. Las máquinas están conectadas a dos redes: *vCloud*, que es una red interna y cerrada, y *AdaByron*, que se trata de una conexión externa a internet mediante un proxy.

Tabla 5: Resumen de las máquinas virtuales usadas en el despliegue

Máquina virtual	Propósito	Instalado	IP
Windows 7 (WIN7)	• Potencial víctima de malware	• *System Internal Suite* (procmon - monitor de procesos) • ApateDNS	192.168.1.5
Ubuntu Server 16.04 (UBU1)	• Potencial víctima de malware • Despliegue firmware de routers	• QEMU • Honeytrap • INetSim	192.168.1.6
T-pot (Ubuntu 16.04)	• Suite de Honeypots configurados en dockers – ELK[4] (Cowrie, Dionaea, Honeytrap, Conpot)	• Configurado: Cowrie, Dionaea, Honeytrap, Conpot	192.168.1.2 192.168.43.66 (Puede acceder a internet sin proxy.)
HoneyDrive (Ubuntu 12.04)	• Suite de Honeypots configurados aunque no en dockers – ELK (Kippo, Dionaea)	• QEMU • Filebeat • Configurado: Kippo, Dionaea	192.168.1.3
Remnux 6 (Ubuntu 14.04)	• Análisis de malware	• QEMU	192.168.1.7
Ubuntu Server 14.04 (UBU2)	• Pruebas de honeypots en dockers	• Docker conpot • Docker intesim	DHCP
ELK (Ubuntu 14.04)	• Visualización de los logs de los honeypots con ElasticSearch, logstash y Kibana	• ElasticSearch • Logstash • Kibana • Grok	192.168.1.8

7.2 Prueba de concepto

En esta sección se pasan a describir las pruebas realizadas en el marco del trabajo. Primeramente, se dará una visión general mediante una tabla en la que se indican sus aspectos básicos, y más tarde serán detalladas, comentando los problemas encontrados, las soluciones, las posibles extensiones, y los resultados y conclusiones obtenidos de ellas.

Tabla 6: Pruebas llevadas a cabo en el despliegue y sus características

Prueba	Objetivos	Dificultades	Lecciones aprendidas
ElasticSearch & logstash	• Gestionar logs de los honeypots	• Enviar los logs de honeypots desde otra máquina no es trivial, ya que los mensajes por TCP a Logstash tienen un formato o siguen unos protocolos concretos.	• Filebeat y las demás herramientas *beats* que se integran con el stack ELK son de gran ayuda al enviar archivos como entrada. • El *Logstash Forwarder* está obsoleto y la solución de

[4] ELK: ElasticSearch, Logstash, Kibana

Prueba	Objetivos	Dificultades	Lecciones aprendidas
		• Los parámetros por defecto en los archivos de configuración daban lugar a problemas de sobreescritura y escalabilidad.	Filebeat es mucho más simple y eficiente. • Hay que controlar todos los parámetros de los archivos de configuración, ya que pueden no ajustarse al problema en cuestión o no funcionar correctamente.
Kibana	• Muestra visual de los logs de los honeypots	• La creación de patrones para los índices que buscar no coincide con los nombres de los índices creados por Logstash. • La creación de las visualizaciones depende de los campos que detecte Kibana o que el usuario filtre en el archivo de configuración de Logstash. • Los tipos de los campos en Kibana son esenciales, ya que no permite operar con campos de tipo desconocido.	• La configuración de Kibana es altamente dependiente de ElasticSearch y Logstash. • La parte de *filter* del archivo de configuración de Logstash es esencial, ya que determina los campos que se podrán usar en las visualizaciones de Kibana. • Cuando se actualizan los campos que reciben los logs, hay que refrescar los índices en el menú de configuración de Kibana para que les asigne automáticamente tipos nuevos.
Honeypot Kippo	• Conectar por SSH al honeypot Kippo en la máquina HoneyDrive, observar el registro de logs y redirigirlo al stack ELK mediante Filebeat.	• La configuración por defecto del firewall de la máquina ELK puede impedir enviar los logs correctamente.	• En un entorno cerrado no son necesarias políticas de firewall estrictas. • Las imágenes de máquinas virtuales con configuraciones predeterminadas pueden necesitar algún ajuste.
Parsing con Grok	• Convertir los mensajes del log de Kippo recibidos en la máquina ELK en campos útiles para Kibana.	• El parsing de los mensajes mediante expresiones regulares no es muy versátil, crea una entrada por cada comando con argumentos distintos.	• Los plugins de Logstash, como Grok y Beats nos facilitan enormemente las tareas.
Infección con Mirai	• Ejecutar la versión de Windows de Mirai y observar su comportamiento.	• Obtener los archivos de ejecución de Mirai es complicado si no se tiene acceso a bases de datos de malware. • Ya que se está en un entorno controlado, hay que suplantar al servidor del que se descargan los archivos, dado que la máquina infectada no tendrá conexión externa.	• Hay herramientas que permiten redireccionar las conexiones externas hacia donde se desee (como ApateDNS) y otras que permiten simular un entorno con muchos servicios (INetSim), las cuales son muy útiles para hacer este despliegue. • Las bases de datos como VirusTotal y Malwr son los mejores lugares donde

Prueba	Objetivos	Dificultades	Lecciones aprendidas
			buscar archivos referentes a la infección con malware.
Indexación en Logstash	• Indexar los logs provenientes de otras máquinas para su visualización en Kibana.	• Los identificadores de cada document se asignan por defecto mediante una expresión regular que no funciona, haciendo que fueran coincidentes. Por ello se sobreescribían unos a otros, provocando una pérdida de información.	• El plugin Rubydebug es útil a la hora de ver el envío y recepción de los mensajes. • Utilizar identificadores basados en timestamp no es una solución escalable en un log, ya que se producen varios mensajes en la misma franja de tiempo. • Kibana cuenta con un límite de 1000 índices distintos, por lo que hay que elegirlos de forma inteligente (por ejemplo, un índice por día sería suficiente para almacenar casi tres años de logs).

7.2.1 Centralización de logs en ELK

Dado que los honeypots se encuentran desplegados en distintas máquinas, cada una va elaborando los registros de actividad en su propio sistema de ficheros, con sintaxis que puede cambiar y guardándose en directorios que dependen de la configuración. Por ello, la solución adoptada es mantener los honyepots operativos en sus máquinas y disponer de una máquina centralizada dedicada a la gestión de estos logs. Así es posible tener una monitorización de todos los honeypots de manera clara y conjunta, observando su comportamiento.

Con este fin se ha usado el stack ELK (ElasticSearch, Logstash y Kibana), por varios motivos, como la sencillez de las visualizaciones finales (que son intuitivas y comprensibles para cualquier usuario) o la amplia disponibilidad de plugins que facilitan muchas de las tareas proyectadas para el sistema.

7.2.1.1 Funcionamiento de ELK

ELK es una solución que combina tres tecnologías/servicios: ElasticSearch, Logstash y Kibana. El flujo general de estos servicios puede verse en la Figura 9, que recoge su funcionamiento junto con el honeypot Kippo, pero para mayor claridad, a continuación se describe en detalle.

Logstash es básicamente un enrutador de datos: recoge los datos, ya sea de máquinas externas mediante una conexión TCP por un puerto que se especifique en la configuración, o de archivos disponibles en la máquina en la que se encuentre. Una vez recibidos estos datos, los procesa (*parsing*) mediante el plugin Grok, para crear campos útiles y comprensibles destinados a ElasticSearch, al cual le envía la información. ElasticSearch es una herramienta de búsqueda en

logs muy potente (escalable y con capacidad de hacer análisis de los datos en tiempo real), que formatea los datos que recibe como documentos JSON y los almacena en índices para hacer su consulta sencilla a Kibana. Este último es un front-end para que el usuario pueda visualizar todos estos datos de manera gráfica. Tiene una gran versatilidad, ya que cuenta con una amplia gama de gráficos distintos en los que se pueden mostrar nuestros datos y agruparlos, filtrarlos, etc.

7.2.1.2 Despliegue de honeypot Kippo

Para la primera prueba consistente en evaluar el potencial de ELK como herramienta de visualización y gestión de logs de los honeypots, se ha desplegado el honeypot Kippo en la máquina HoneyDrive, basada en Xubuntu 12.04. La figura a continuación muestra el entorno específico de esta prueba y la relación entre las máquinas.

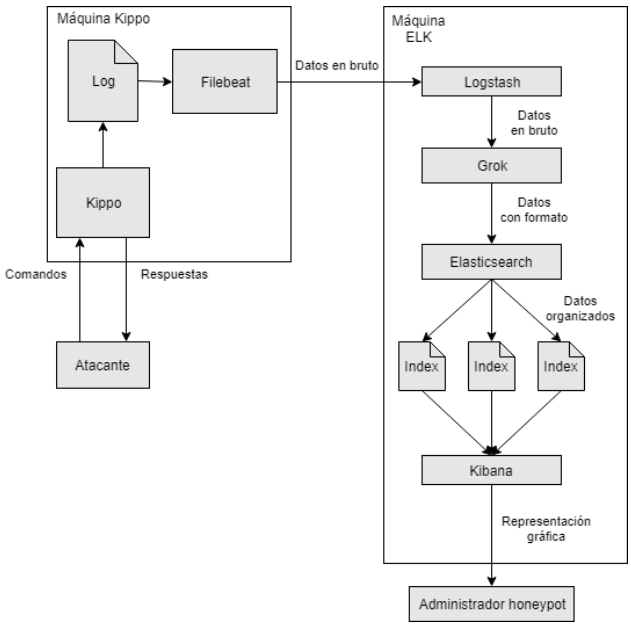

Figura 9: Despliegue de honeypot Kippo y del stack ELK

El log de la máquina Kippo se guarda en */honeydrive/Kippo/logs/kippo.log* por defecto, con un formato predeterminado. La máquina en la que corre el stack ELK está basada en una MV que provee Bitnami, la cual tiene instalada la última versión de todas las herramientas (5.3), basada en Ubuntu Server 14.04. Para enviarle los logs desde HoneyDrive se ha instalado el plugin Filebeat. Este plugin se configura para enviar el archivo de log a la máquina ELK en el puerto 5044.

En este punto de la configuración se deberían ver los logs siendo enviados correctamente mediante entradas en el log de Filebeat (*/var/log/mybeat/mybeat.log*), pero dada la configuración del firewall en la máquina de ELK, esto no sucedía correctamente correctamente. Para solventar esto, como se trata de una configuración local y una primera prueba provisional,

se ha desactivado por completo el firewall de esta máquina para cualquier puerto. Además, para evitar problemas con direccionamiento proxy, se ha añadido la IP estática de la máquina ELK al archivo */etc/hosts* con el nombre *elk*.

Una vez Logstash recibe los ficheros y los reenvía a ElasticSearch, se han de elaborar visualizaciones con Kibana. La máquina de Bitnami presenta la limitación de no contar con una interfaz gráfica en la que se pueda ejecutar un navegador para observar la funcionalidad de Kibana, por lo que se ha de acceder mediante otro método. Una solución sería instalar un entorno gráfico al que tenga acceso un perfil de administrador. Como de momento no se trata de un entorno de producción, se ha decidido primar la simplicidad, conservar la configuración de red, y las máquinas livianas sin el peso que supone un entorno gráfico. La alternativa escogida es acceder desde la máquina de HoneyDrive. Para ello se necesita cambiar la configuración de ElasticSearch en el archivo */opt/bitnami/elasticsearch/config/kibana.yml*, modificando los parámetros *network.host* y *network.publish_host*, haciendo que estén disponibles para el resto de la red y no sólo para la máquina local. En el caso de la máquina virtual de Bitnami, la configuración ya viene preparada para poder acceder a Kibana desde fuera. En caso de que no fuera así, se tendría que preparar un proxy inverso o modificar el parámetro de *server.host* y los hosts del sistema.

Al iniciar Kibana se presenta una ventana de configuración para elegir en qué se desean llevar a cabo las búsquedas. Dado que todos nuestros logs provienen de Logstash, se guardan con el nombre *logstash-YYYY.MM.dd* (indicando las últimas letras año, mes y día), por lo que se realizó una búsqueda de todos los índices cuyo comienzo fuese *logstash-* mediante la expresión regular *logstash-**. El nombre de los índices se puede cambiar en la parte *output* del fichero de configuración de Logstash. Este índice, por supuesto, se puede cambiar, poniendo por ejemplo, prefijos para particularizar los índices, indicando el honeypot desde el que provienen los registros.

7.2.1.3 Almacenamiento de logs y ElasticSearch

En las primeras fases de la prueba, se pudo comprobar que, aunque se enviaban los mensajes desde los logs de Kippo a la máquina ELK, no se mostraban todos en Kibana. Para comprobar que el problema no se encontraba en la recepción de los mensajes, se usó el plugin Rubydebug de Logstash. Tras comprobar esto, la única parte en la que se podía producir el fallo era en la indexación.

En la parte de *output* de la configuración de Logstash se envía cada mensaje de Kippo a un índice con el formato por defecto indicado anteriormente, especificando año, mes y día, por lo que al

recibir varios mensajes el mismo día, se sobreescribían en el mismo índice y sólo era posible visualizar el último. La primera solución a esto fue hacer índices de grano más fino, incluyendo hasta los segundos, por lo que los mensajes se indexaron como *logstash-YYYY.MM.dd-HH:mm:ss*.

Sin embargo, al generar mucho tráfico, se observaba que el número de índices crecía de una manera alarmante. Teniendo en cuenta que Kibana no trabaja a partir de 1000 índices, y lo óptimo es tener menos índices de mayor tamaño, fue necesario buscar otro método.

Para conseguir organizar los índices de tal manera, se observó la forma de trabajo de ElasticSearch, que trata de incluir todos los documentos posibles en un mismo índice. En este caso, esto no ocurría porque el archivo de configuración especificaba que el ID de cada documento fuese *%{logstash_checksum}*, que debería elaborar un valor MD5 con todos los campos y se lo asignase como identificador. No obstante, esto no estaba funcionando, por lo que se le asignaba como ID a todos los documentos la cadena literal "*%{logstash_checksum}*". Esto provocaba que cada vez que llegase un mensaje nuevo, al haber otro documento guardado con el mismo ID, se tomase como una actualización y se sobreescribiese. Simplemente eliminar esta fila en el archivo de configuración permitió generar menos índices y que fueran más extensos, haciendo una solución más escalable y rápida, ya que la muestra de los datos en Kibana es prácticamente igual.

7.2.1.4 Visualización de logs provenientes de Kippo

Cuando ya se pueden observar en Kibana los mensajes de Kippo, la siguiente tarea es crear visualizaciones relevantes que representen los datos obtenidos mediante los honeypots y poder estudiarlos. Para ello, se han de usar los campos que procesa Kibana de los documentos JSON que le provee ElasticSearch. El problema es que estos campos no contienen la información formateada de una manera que permita ser usada de manera cómoda.

Por ello, se filtran las entradas en Logstash para que se ajusten a unos campos que se especifican en la configuración. Esta tarea se lleva a cabo con Grok, que mediante expresiones regulares permite dividir el mensaje y guardar en una variable cada campo que el usuario considere relevante. Es importante apuntar que hay que refrescar los patrones de los índices de Kibana para que tenga en cuenta los campos nuevos obtenidos mediante Grok, porque de lo contrario se mostrarán como tipo *unknown* (desconocido) y no se podrán usar para elaborar visualizaciones.

Una vez configurados los campos para las visualizaciones, Kibana permite hacer distintos tipos de agregaciones (cuenta, suma, media…) y elegir qué datos se van a representar según ellos en

los ejes. Se pueden escoger términos, rangos de IP o números, entre otros. Para el caso que nos ocupa se han escogido dos visualizaciones sencillas: el número de entradas en el honeypot por IP y el número de ejecuciones de cada comando. El primero nos interesa para saber desde qué IP se han ejecutado más comandos, y el segundo para saber qué es lo que más intentan los atacantes al honeypot. Estas visualizaciones se guardan y añaden al *dashboard* para poder observarlas de un vistazo al entrar.

7.2.2 Infección con Mirai (versión Windows)

En octubre de 2016, un malware específicamente enfocado a dispositivos IoT llamado Mirai causó un ataque distribuido de denegación de servicio contra el proveedor de DNS estadounidense Dyn, haciendo que miles de dispositivos (como routers, DVRs, cámaras IP, etc.) que conservaban las credenciales de acceso por defecto del fabricante formasen parte de una botnet a merced de servidores de *command and control*. La magnitud de este ataque no tenía precedentes y afectó a servicios muy populares, tales como Spotify, Twitter, Netflix o Amazon estuvieran afectados.

Recientemente, el malware ha dado el salto a plataformas Windows, aunque comportándose de forma distinta. Anteriormente, sólo los dispositivos IoT que fueran visibles de forma pública (expuestos con una IP, como muchas cámaras de seguridad o routers de empresas, por ejemplo) eran susceptibles a Mirai. Ahora, si el malware entra en un equipo Windows y se extiende a los dispositivos que se encuentren en su red, los aparatos que se encuentren tras un *firewall* se pueden ver comprometidos.

A continuación se detalla la infección de Mirai de un equipo Windows en el entorno de pruebas controlado. Los pasos del proceso de infección se muestran en la Figura 10.

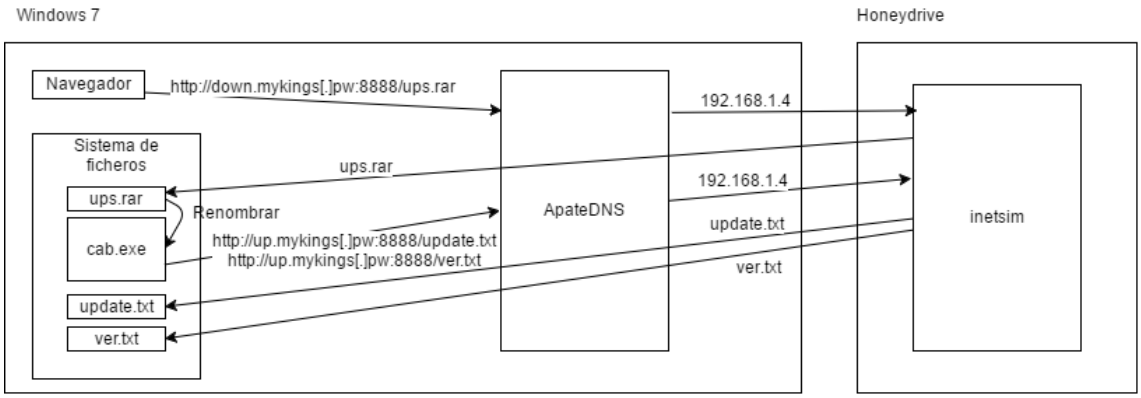

Figura 10: Despliegue de Mirai en Windows 7 y HoneyDrive

Los *samples* del malware se han descargado de VirusTotal [Viru2018] y Malwr [Malw2018], y el procedimiento de infección se ha realizado conforme un artículo de Securelist [Grea2017]. La máquina de Windows 7 que se va a infectar es una máquina virtual en una vApp sin conexión a internet, sólo a la red interna virtualizada, a la cual están conectadas a su vez otras máquinas virtuales.

7.2.2.1 Descarga del software malicioso

Como el primer paso de la infección es descargar un ejecutable malicioso y la máquina no dispone de conexión a internet, se ha montado en otra máquina virtual (HoneyDrive) de la vApp un servidor web con los archivos que se obtendrían. El servidor se ha puesto a punto mediante INetSim, en el puerto 8888 (ya que es al que intenta conectarse el ejecutable de Mirai), y colocando los archivos a descargar con los nombres pertinentes en el directorio raíz que se ofrece a las conexiones. Una vez se tiene lista esta parte del entorno, en Windows 7 hay que redirigir las conexiones para que, si la URL del ejecutable tiene como destino una web externa, la reciba la máquina HoneyDrive. Esto se hace mediante ApateDNS, que actúa como un servidor de DNS, resolviendo todas las conexiones externas hacia la IP que nos interesa.

7.2.2.2 Ejecución del malware

Una vez obtenido el primer archivo, *ups.rar*, se siguen las acciones detalladas en el informe de SecureList y se renombra a *cab.exe* (aunque se podría haber elegido cualquier otro de los nombres a los que se suele cambiar), procediendo a su ejecución. Si se realiza la ejecución con permisos de administrador, cambiará la configuración de DNS de la red a las direcciones 114.114.114.114 y 8.8.8.8, la primaria y secundaria respectivamente. En caso de ejecutarlo como usuario normal, intentará realizar esta acción, pero se le denegará el permiso y se cerrará. Tras ello, intenta de nuevo conectarse a dos direcciones del mismo dominio del que se descargó el ejecutable. ApateDNS las vuelve a redirigir al servidor INetSim de HoneyDrive. Uno de los archivos que intenta descargar es *ver.txt*, el cual consigue.

La salida por consola del ejecutable es

DNS set ok.

ver different web:1.0.0.7 local:, needs update.

La última parte del mensaje se refiere a que también intenta descargar el archivo *update.txt*. Sin embargo, el MD5 del archivo que aparece en el artículo de Securelist no está disponible para descarga desde ninguna de las plataformas de *samples* de malware que han sido usadas. Con el

comando *file* de Unix, se puede ver que de todos los *samples* de malware de Mirai, sólo uno de los disponibles es de tipo texto, por lo que se renombra a *update.txt* y se ejecuta de nuevo, pero el resultado es el mismo. Al llegar a este punto, el ejecutable se cuelga.

Si se arranca el ejecutable sin ApateDNS, mostrará por consola el mensaje *get file list failed, exit*, y se borrará a sí mismo.

Puede que algunos de los pasos detallados no sean esenciales para llevar a cabo con éxito la infección, pero a fin de recrearla con toda la fidelidad posible, se han seguido todos ellos. Ante la imposibilidad de encontrar el archivo *update.txt* en ninguna de las bases de datos de malware consultadas, el proceso de infección queda terminado antes de propagarse con éxito.

7.2.2.3 Visualización en ELK

Para que quedase constancia de las ejecuciones de la infección, se tuvo que cambiar el comportamiento del envío de archivos al stack ELK. Desde Filebeat, además del log de Kippo, como se detalla en el apartado anterior, el log *service* de INetSim, que detalla las peticiones que atiende y sus detalles (origen, archivos que sirve, timestamp…). Una vez llegaron los mensajes de manera adecuada y se hizo el *parsing* de su formato mediante el plugin Grok, hubo que separarlos, ya que tenerlos en los mismos índices de Kippo bajo el nombre *logstash-** suponía una mezcla de datos sin relación. Para ello, se filtraron los mensajes según su archivo de procedencia: los mensajes de Kippo se almacenan ahora en índices con el nombre *kippo-** y los referentes al tráfico de Mirai (y otras peticiones a INetSim) bajo *inetsim-**. Esto facilita poder identificar según el comportamiento de las peticiones, si está ocurriendo alguna infección y discernir si se trata de Mirai o de alguna otra botnet.

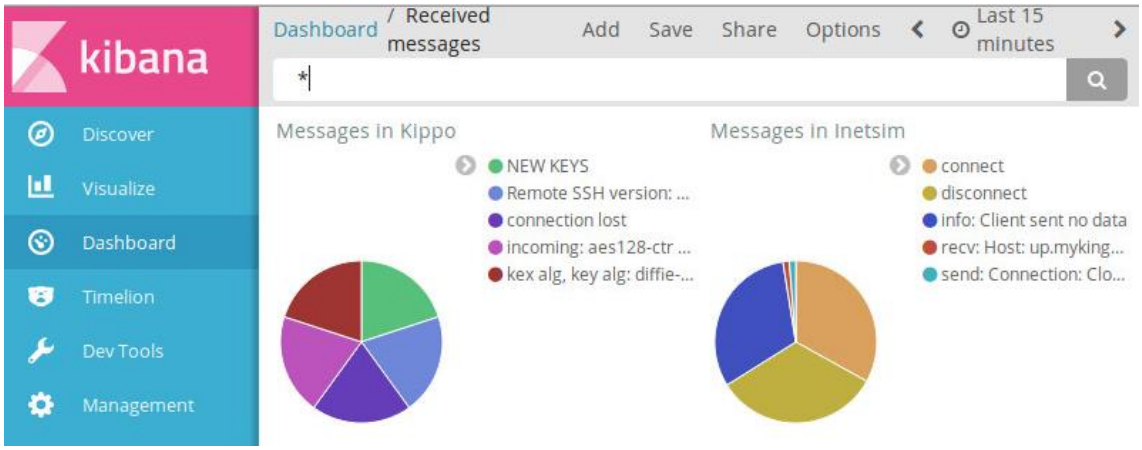

Figura 11: Visualizaciones en Kibana de los mensajes del log de INetSim y Kippo.

En la Figura 11 se puede ver cómo en los mensajes de los logs de INetSim, además de conexiones (en la leyenda, en naranja), hay peticiones al host *up.mykings.pw* (en la leyenda en rojo), tratándose del servidor al que intenta contactar el ejecutable de Mirai. Redirigido por ApateDNS en la máquina de Windows, llega al servidor de INetSim, el cual le devuelve los archivos que pide (en la leyenda, en azul celeste). Todos estos mensajes quedan registrados y podrían usarse como alerta para saber que hay una versión de Mirai intentando obtener los archivos maliciosos.

7.2.2.4 Trazas de red

De forma complementaria, además de guardar los registros del log de INetSim para su posterior consulta, se lleva a cabo paralelamente una captura del tráfico de red entre la máquina infectada y la que suplanta al servidor. En unas capturas es posible ver cómo los archivos se transfieren entre las dos máquinas.

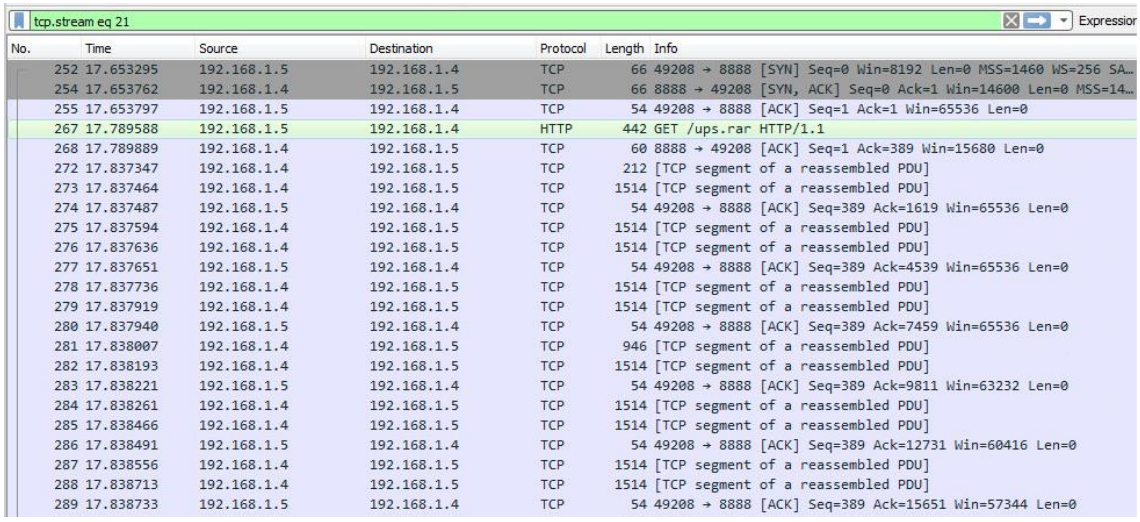

Figura 12: Petición y envío del archivo ups.rar (o cab.exe como ejecutable)

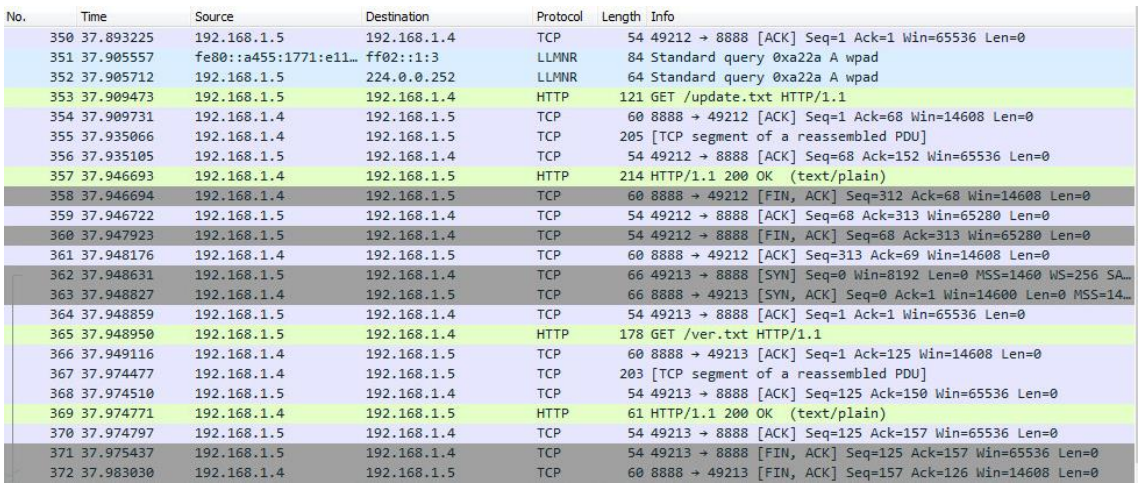

Figura 13: Petición y envío de los archivos update.txt y ver.txt

La primera captura de Wireshark muestra la petición del archivo *ups.rar* y su recepción en el equipo Windows. Este podría ser el primer indicador de que algo fuera de lo normal está ocurriendo en el equipo Windows. La siguiente muestra las últimas peticiones de archivos que se hace al servidor desde el equipo Windows que será infectado con Mirai.

8. Extensiones y consideraciones

A continuación se detallan algunos puntos abiertos que podrían elaborarse como extensión a este trabajo para ampliaciones futuras:

- **Infección desde una botnet Mirai**. En lugar de descargar el malware y realizar la infección manualmente, sería interesante provocar que la máquina se infecte desde una botnet de Mirai ya existente. Este enfoque permitiría ver cómo la máquina es indexada por los mecanismos de monitorización de dispositivos con Mirai. No sería sencillo, puesto que significaría dejar la máquina expuesta al exterior y contribuiría a posibles propagaciones.
- **Infección a red de dispositivos IoT desde Windows**. Crear una honeynet para este caso específico, en el que el equipo Windows esté conectado con dispositivos IoT y ver cómo se infectan.
- **Emulación de dispositivos IoT con QEMU**. Mediante una plataforma de emulación, y disponiendo de firmware de los dispositivos a tratar, podrían crearse máquinas que simularan ser dispositivos IoT vulnerables a modo de honeypot, pero con una interacción completa. Este procedimiento es complejo, porque en muchas ocasiones el firmware presenta una fuerte dependencia de la plataforma hardware a la que se destina (arquitecturas de los procesadores, sistemas de archivos…).
- **Conexión con ELK para que muestre resultados del ataque y sistema de inteligencia**. Consiste en hacer que desde HoneyDrive se envíen las acciones a la máquina ELK, y que el sistema de inteligencia se dé cuenta basándose en el comportamiento (peticiones, mensajes, *timing*) de que se trata de Mirai. Ya que este malware en concreto se destruye automáticamente si el equipo se queda sin conexión tras la infección, decidir aislar automáticamente el equipo Windows 7 (infectado) hasta que se copie el código de Mirai, las evidencias y aquellos ficheros relevantes para el análisis del malware.
- **Desplegar "infección" de Hajime o dejarse infectar por una botnet**. Como se ha descrito anteriormente, Hajime se trata de una botnet que basa su funcionamiento en el mismo patrón que Mirai: entra a los dispositivos que conserven las credenciales por defecto del

fabricante mediante telnet, usando prácticamente la misma lista de usuarios y contraseñas. Sin embargo, una vez entra, bloquea el acceso desde varios puertos, para evitar ataques vía telnet o TR-069/TR-064. Al contrario que Mirai, no tiene servidores de *command & control*, sino que se basa en un esquema *peer-to-peer*, lo que hace más difícil detener su avance, y no tiene (por el momento) capacidad de efectuar ataques de DDoS.

9. Conclusiones

A lo largo del trabajo, se ha detallado cómo el escenario IoT no sólo es novedoso y desafiante, sino una realidad palpable. La mayoría de las personas que utilizan internet en su vida diaria contarán con uno o varios dispositivos IoT, e interactuarán con una infinidad a lo largo del día, muchas veces sin ser siquiera conscientes de ello.

Los dispositivos IoT no han sido diseñados con las suficientes medidas de seguridad, y se les han dado roles muy importantes, tales como preservar nuestra privacidad o seguridad. La situación resultante es un aparato muy vulnerable llevando a cabo tareas muy sensibles.

Mediante la observación de los ataques recientes comentados en el trabajo, se puede concluir que el perfil de un atacante a un dispositivo IoT, no es el de un ataque dirigido a un dispositivo concreto con un propósito definido, sino el de un ataque masivo y automatizado, normalmente llevando a cabo un escaneo de la red, y sin ser especialmente sofisticado y discreto.

Este tipo de ataques llevan a cabo comprobaciones rutinarias a cada dispositivo al que se conectan, comprobando su respuesta a las mismas. Debido a las características de estos ataques, en las que no hay una intervención humana que pueda sacar conclusiones, es más complicado que el ataque llegue a la conclusión de que está interactuando con un honeypot.

Contando con esta ventaja, el sistema de honeypot desplegado es una de las mejores alternativas, y perfectamente viable por su bajo coste de implementación y operación. Además, el entorno virtualizado permite asegurarse de que las consecuencias del ataque no se extenderán, y hace posible observar cómo se desarrolla una infección de malware IoT en un espacio controlado, sacando conclusiones de su comportamiento que nos permitan elaborar medidas de prevención en el futuro, además de no tener que contar con el gasto que supone disponer de dispositivos IoT físicos, pudiendo emularlos.

Asimismo, dado el análisis llevado a cabo, tanto de dispositivos, como de soluciones honeypot, se puede ver que la necesidad de sistemas que nos permitan conocer más sobre los ataques IoT es imperiante. No se trata de casos aislados, sino de diversos problemas presentes incluso en

primeras marcas (Sony, Axis, productos recomendados por Amazon), con cientos de miles de aparatos susceptibles a ataques, y de una tendencia creciente en el malware dirigido específicamente a esta plataforma.

Referencias

[Asht2009] Ashton, K. (2009). That 'internet of things' thing. *RFID journal*, *22*(7), 97-114.

[Blan2016] Blanter, A., & Holman, M. (2016). Internet of things 2020: a glimpse into the future. *Available at Kearney https://www. atkearney. com/documents/4634214/6398631/AT+ Kearney_Internet+ of+ Things*, *2020*.

[Vest2010] Vestburg, H. (2010). CEO to shareholders: 50 billion connections 2020. *Ericsson, April*, *13*.

[Rose2015] Rose, K., Eldridge, S., & Chapin, L. (2015). The internet of things: An overview. *The Internet Society (ISOC)*, 1-50.

[Roma2011] Roman, R., Najera, P., & Lopez, J. (2011). Securing the internet of things. *Computer*, *44*(9), 51-58.

[Cui2011] Cui, A., & Stolfo, S. J. (2011, April). Reflections on the engineering and operation of a large-scale embedded device vulnerability scanner. In *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security* (pp. 8-18). ACM.

[Anto2017] Antonakakis, M., April, T., Bailey, M., Bernhard, M., Bursztein, E., Cochran, J., ... & Kumar, D. (2017). Understanding the mirai botnet. In *USENIX Security Symposium*.

[Koli2017] Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J. (2017). DDoS in the IoT: Mirai and other botnets. *Computer*, *50*(7), 80-84.

[Luo2016] Luo, A. (2016). Drones Hijacking.

[Rone2016] Ronen, E., & Shamir, A. (2016, March). Extended functionality attacks on iot devices: The case of smart lights. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on* (pp. 3-12). IEEE.

[Kris2017a] Krishnaprasad, P. (2017). *Capturing attacks on IoT devices with a multi-purpose IoT honeypot* (Doctoral dissertation, PhD thesis, INDIAN INSTITUTE OF TECHNOLOGY KANPUR).

[Wang2017] Wang, M., Santillan, J., & Kuipers, F. (2017). ThingPot: an interactive Internet-of-Things honeypot.

[Luo2017] Luo, T., Xu, Z., Jin, X., Jia, Y., & Ouyang, X. (2017). IoTCandyJar: Towards an Intelligent-Interaction Honeypot for IoT Devices. Black Hat.

[Dowl2017] Dowling, S., Schukat, M., & Melvin, H. (2017, June). A ZigBee honeypot to assess IoT cyberattack behaviour. In Signals and Systems Conference (ISSC), 2017 28th Irish (pp. 1-6). IEEE.

[Edwa2016] Edwards, S., & Profetis, I. (2016). Hajime: Analysis of a decentralized internet worm for IoT devices. Rapidity Networks, 16.

[Pa2015] Pa, Y. M. P., Suzuki, S., Yoshioka, K., Matsumoto, T., Kasama, T., & Rossow, C. (2015). IoTPOT: analysing the rise of IoT compromises. EMU, 9, 1.

[Guar2017] Guarnizo, J. D., Tambe, A., Bhunia, S. S., Ochoa, M., Tippenhauer, N. O., Shabtai, A., & Elovici, Y. (2017, April). Siphon: Towards scalable high-interaction physical

honeypots. In Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security (pp. 57-68). ACM.

[Yeh2017] T. Yeh, D. Chiu, and K. Lu, “Persirai: New Internet of Things (IoT) Botnet Targets IP Cameras,” blog, TrendLabs, 9 May 2017 https://blog.trendmicro.com/trendlabs-security-intelligence/persirai-new-internet-things-iot-botnet-targets-ip-cameras/, accedida a 14/02/2018

[Kreb2017] Krebs, B. “New Bill Seeks Basic IoT Security Standards,” blog, KrebsOnSecurity, 1 Aug 2017 https://krebsonsecurity.com/2017/08/new-bill-seeks-basic-iot-security-standards/, accedida a 14/02/2018

[Hay2016] Hay Newman, L., "The Botnet that Broke the Internet Isn’t Going Away,” blog, Wired, 12 Sep 2016 https://www.wired.com/2016/12/botnet-broke-internet-isnt-going-away/, accedida a 14/02/2018

[GoAh2018] GoAhead, “EmbedThis GoAhead: Simple, Secure Embedded Web Server”, página web, https://www.embedthis.com/goahead/ accedida a 14/02/2018

[Reve2016] Reverse Engineering Blog “Eir’s D1000 Modem Is Wide Open To Being Hacked,” blog, Reverse Engineering Blog, 7 Nov 2016 https://devicereversing.wordpress.com/2016/11/07/eirs-d1000-modem-is-wide-open-to-being-hacked/

[Hone2018] Honeyscore “Shodan Honeyscore: Honeypot Or Not?”, página web, accedida el 14/02/2018 https://honeyscore.shodan.io/

[Scan2018] Scanhub “Shodan Scanhub: Make Nmap Results Searchable”, página web, accedida el 14/02/2018 https://scanhub.shodan.io/

[Expl2018] Exploitee.rs Wiki, página web, accedida el 14/02/2018 https://www.exploitee.rs/

[Cved2018a] CVE Details, página web, accedida el 14/02/2018 https://www.cvedetails.com/

[Exdb2018] Exploit Database, página web, accedida el 14/02/2018 https://www.exploit-db.com/

[Cved2012a] CVE Details, “CVE-2012-5687”, página web, accedida el 14/02/2018 https://www.cvedetails.com/cve/CVE-2012-5687/

[Cved2012b] CVE Details, “CVE-2012-6276”, página web, accedida el 14/02/2018 https://www.cvedetails.com/cve/CVE-2012-6276/

[Cved2012c] CVE Details, “CVE-2012-6316”, página web, accedida el 14/02/2018 http://www.cvedetails.com/cve/CVE-2012-6316/

[Cved2015a] CVE Details, “CVE-2015-3035”, página web, accedida el 14/02/2018 http://www.cvedetails.com/cve/CVE-2015-3035/

[Cved2014a] CVE Details, “CVE-2014-9350”, página web, accedida el 14/02/2018 http://www.cvedetails.com/cve/CVE-2014-9350/

[Cons2014] Constantin, L., “Exploit released for vulnerability targeted by Linksys router worm,” blog, PCWorld, 17 Feb 2014, https://www.pcworld.com/article/2098520/exploit-released-for-vulnerability-targeted-by-linksys-router-worm.html

[Expl2016] Exploit Database, “Linksys E-series - Unauthenticated Remote Code Execution”, página web, 16 Feb 2016, accedida el 14/02/2018 https://www.exploitdb.com/exploits/31683/

[Expl2013a] Exploit Database, “NETGEAR WNR1000 - Authentication Bypass”, página web, 2 Abr 2013, accedida el 14/02/2018 https://www.exploit-db.com/exploits/24916/

[Expl2013b] Exploit Database, “Linksys E1500/E2500 - Multiple Vulnerabilities”, página web, 11 Feb 2013, accedida el 14/2/2018 https://www.exploit-db.com/exploits/24475/

[Sans2016] SANS ISC InfoSec Forums, “TR-069 NewNTPServer Exploits: What we know so far”, lista de correo, 27 Nov 2016, accedida el 14/02/2018

[Good2016] Goodin, D., “Newly Discovered router flaw being hammered by in-the-wild attacks,” Ars Technica, blog, 28 Nov 2016 https://arstechnica.com/information-technology/2016/11/notorious-iot-botnets-weaponize-new-flaw-found-in-millions-of-home-routers/

[Cved2017a] CVE Details “CVE-2017-6552”, página web, accedida el 15/02/2018 http://www.cvedetails.com/cve/CVE-2017-6552/

[Cved2014b] CVE Details “CVE-2014-2718”, página web, accedida el 15/02/2018 https://www.cvedetails.com/cve/CVE-2014-2718/

[Cved2018b] CVE Details “Netgear R6700 Firmware Vulnerability Statistics”, página web, accedida el 15/02/2018 https://www.cvedetails.com/product/35406/Netgear-R6700-Firmware.html?vendor_id=834

[Cved2009] CVE Details “CVE-2009-3341”, página web, accedida el 15/02/2018 https://www.cvedetails.com/cve/CVE-2009-3341/

[Cved2014c] CVE Details “CVE-2014-8244”, página web, accedida el 15/02/2018 https://www.cvedetails.com/cve/CVE-2014-8244/

[Dlin2018] Exploit Database, “D-Link DCS Cameras - Multiple Vulnerabilities”, página web, accedida el 15/02/2018 https://www.exploit-db.com/exploits/24442/

[Cved2007a] CVE Details, “CVE-2007-3488”, página web, accedida el 15/02/2018 https://www.cvedetails.com/cve/CVE-2007-3488/

[Cved2011a] CVE Details, “CVE-2011-4859”, página web, accedida el 15/02/2018 https://www.cvedetails.com/cve/CVE-2011-4859/

[Schn2015] Schneider Electric, "Important Security Notification - SEVD-2015-344-01 - GoAhead Web Server vulnerability“, notificación oficial, 10 Dic 2015 http://download.schneider-electric.com/files?p_Doc_Ref=SEVD-2015-344-01

[Cved2017b] CVE Details, “CVE-2017-5157”, página web, accedida el 15/02/2018 https://www.cvedetails.com/cve/CVE-2017-5157/

[Cved2014d] CVE Details, “CVE-2014-9197”, página web, accedida el 15/02/2018 http://www.cvedetails.com/cve/CVE-2014-9197/

[Icsc2015] ICS-CERT “Advisory (ICSA-15-274-01) - Omron Multiple Product Vulnerabilities”, documento oficial, 1 Oct 2015 https://ics-cert.us-cert.gov/advisories/ICSA-15-274-01

[Yuan2015] Yuan, J. (2015). “GPS Spoofing of UAV”, Qihoo 360 Technology Co. Ltd.

[Phas2017] Phasenoise, “Review & Teardown of a cheap GPS Jammer”, Phasenoise, blog, 30 Nov 2017 http://phasenoise.livejournal.com/3185.html

[Truj2016] Trujano, G. B. R. R. F., Chan, B., Beams, G., & Rivera, R. (2016). Security Analysis of DJI Phantom 3 Standard. Massachusetts Institute of Technology, May.

[Stag2017] Stagno, P., “Hacking the DJI Phantom 3”, Voidsec, blog, 13 Ene 2017 https://voidsec.com/hacking-dji-phantom-3/

[Szab2016] Szabó, M., “Drone Hacking”, Github, repositorio de código, accedido el 15/02/2018 https://github.com/markszabo/drone-hacking

[Foxb2017] Fox-Brewster, T., “Watch A Very Vulnerable $140 Quadcopter Drone Get Hacked Out Of The Sky”, Forbes, blog, 25 Abr 2017 https://www.forbes.com/sites/thomasbrewster/2017/04/25/vulnerable-quadcopter-drone-hacked-by-ut-dallas-cyber-researchers/#732afeea1037

[Hugh2015] Hughes, M., “Why the iKettle Hack Should Worry You (Even If You Don’t Own One)”, MakeUseOf, blog, 23 Oct 2015 https://www.makeuseof.com/tag/ikettle-hack-worry-even-dont-one/

[Paul2015] Pauli, D., “Connected kettles boil over, spill Wi-Fi passwords over London”, The Register, blog, 19 Oct 2015 https://www.theregister.co.uk/2015/10/19/bods_brew_ikettle_20_hack_plot_vulnerable_london_pots/

[Munr2015a] Munro, K., “Yet another attack against the iKettle wireless kettle. Rumpy pumpy and fire alarms?”, PenTestPartners, blog, 23 Jun 2015 https://www.pentestpartners.com/security-blog/yet-another-attack-against-the-ikettle-wireless-kettle-rumpy-pumpy-and-fire-alarms/

[Vend2015] Venda, P., “Hacking DefCon 23’s IoT Village Samsung fridge”, PenTestPartners, blog, 18 Aug 2015 https://www.pentestpartners.com/security-blog/hacking-defcon-23s-iot-village-samsung-fridge/

[Munr2015b] Munro, K., “Extracting your WPA PSK from bathroom scales”, PenTestPartners, blog, 8 Jun 2015 https://www.pentestpartners.com/security-blog/extracting-your-wpa-psk-from-bathroom-scales/

[Cved2017c] CVE Details, “CVE-2017-7240”, página web, accedida el 15/02/2018 https://www.cvedetails.com/cve/CVE-2017-7240/?q=CVE-2017-7240

[Paul2017] Pauli, D., “IoT worm can hack Philips Hue lightbulbs, spread across cities”, The Register, blog, 10 Nov 2016 https://www.theregister.co.uk/2016/11/10/iot_worm_can_hack_philips_hue_lightbulbs_spread_across_cities/

[Sara2017] Sarabia, D., “Un oso de peluche expone en Internet dos millones de conversaciones entre padres e hijos”, ElDiario, periódico online, 28 Feb 2017

[Chad2017] Chadha, S., “Honeything: The IoT Trap System”, Medium, blog, 26 Mar 2017 https://medium.com/@shreya16aug/honeything-the-iot-trap-system-12e7bf9b6a8a

[Jich2016] Jicha, A., Patton, M., & Chen, H. (2016, September). SCADA honeypots: An in-depth analysis of Conpot. In Intelligence and Security Informatics (ISI), 2016 IEEE Conference on (pp. 196-198). IEEE.

[Viru2018] VirusTotal, página web, accedida el 15/02/2018 https://www.virustotal.com/es/

[Malw2018] Malwr, página web, accedida el 15/02/2018 https://malwr.com/

[Grea2017] Great, "New(ish) Mirai Spreader Poses New Risks", SecureList, blog, 21 Feb 2017 https://securelist.com/newish-mirai-spreader-poses-new-risks/77621/

[Buza2014] Buza, D. I., Juhász, F., Miru, G., Félegyházi, M., & Holczer, T. (2014, February). CryPLH: Protecting smart energy systems from targeted attacks with a PLC honeypot. In International Workshop on Smart Grid Security (pp. 181-192). Springer, Cham.

[Leit2007] Leita, C., Dacier, M., & Wicherski, G. (2007). SGNET: a distributed infrastructure to handle zero-day exploits. Institut Eurecom, France, Tech. Rep. EURECOM, 2164.

[Schi2015] Schindler, S., Schnor, B., & Scheffler, T. (2015). Hyhoneydv6: A hybrid honeypot architecture for ipv6 networks. International Journal of Intelligent Computing Research, 6.

[Conp2014] Conpot, página web, accedida el 16/02/2018 http://conpot.org

[Bred2015] Bredo, A., Honeypot-Camera, repositorio de código, accedido el 16/02/2018 https://github.com/alexbredo/honeypot-camera

[Wang2017a] Wang, M., Santillan, J., & Kuipers, F. (2017, November) ThingPot: an interactive Internet-of-Things honeypot. TU Delft, Netherlands, BrightSight.

[Wang2017b] Wang, M., ThingPot, repositorio de código, accedido el 16/02/2018 https://github.com/Mengmengada

[Erde2016] Erdem, O., HoneyThing, repositorio de código, accedido el 16/02/2018 https://github.com/omererdem/honeything

[Kris2017b] Krishnaprasad, P., repositorio de código, accedido el 16/02/2018 https://github.com/krishnaprasad-p

[Gold2016] Goldberg, I., MTPot, repositorio de código, accedido el 16/02/2018 https://github.com/Cymmetria/MTPot

[Brit2017] Britton, T., Wificam, repositorio de código, accedido el 16/02/2018 https://github.com/h-m-s/wificam

[Schl2014] Schloesser, M., Dionaea, repositorio de código, accedido el 16/02/2018 https://github.com/rep/dionaea

[Prov2007] Provos, N., Honeyd, página web, accedida el 16/02/2018 http://www.honeyd.org/

[Oost2018] Oosterhof, M., Cowrie, repositorio de código, accedido el 16/02/2018 https://github.com/micheloosterhof/cowrie

[Tros2015] Trost, J., Shockpot, repositorio de código, accedido el 16/02/2018 https://github.com/threatstream/shockpot/

[Shod2018] Shodan, página web, accedida el 16/02/2018 https://www.shodan.io/

[Cens2018] Censys, página web, accedida el 16/02/2018 https://censys.io/

[Repo2018] Reposify, página web, accedida el 16/02/2018 https://www.reposify.com/

[Thin2018] Thingful, página web, accedida el 16/02/2018 https://www.thingful.net/

[Wigl2018] Wigle, página web, accedida el 16/02/2018 https://wigle.net/

[Stat2018] Statista, Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions), página web, accedida el 20/02/2018 https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

[Busi2015] Business Intelligence, "This one chart explains why cybersecurity is so important", 22 Abr 2016, blog, accedido el 20/04/2016 http://www.businessinsider.com/the-cybersecurity-report-threats-and-opportunities-2016-4-22

[Mimo2017] Mimoso, M., "IoT malware activity already more than doubled 2016 numbers", ThreatPost, blog, 19 Jun 2017, accedido el 20/02/2018 https://threatpost.com/iot-malware-activity-already-more-than-doubled-2016-numbers/126350/

[Como2017] Comodo, "Your IoT Devices May Be Potential Targets to Malware That Wipe Out Data", 23 May 2017, blog, accedido el 20/02/2018 https://blog.comodo.com/endpoint-security/your-iot-devices-may-be-potential-targets-to-malware-that-wipe-out-data/

[Purs2017] Pursche, O., "IoT malware? Nothing that's new!", LinkedIn, 1 Ago 2017, blog, accedido el 21/02/2018 https://www.linkedin.com/pulse/iot-malware-nothing-new-olaf-pursche