

**Universidad  
Rey Juan Carlos**

Escuela Técnica Superior  
de Ingeniería Informática

**Grado en Ingeniería del Software**

**Curso 2024-2025**

**Trabajo Fin de Grado**

**EVALUACIÓN DE LA SEGURIDAD DE  
DISPOSITIVOS IOT MEDIANTE LA  
CONFIGURACIÓN DE UN HONEYPOT**

**Autor: Fernando Alonso Calderón  
Tutor: Tomás Isasia Infante**



# Agradecimientos

Gracias.

A los profesores que me han enseñado a lo largo de estos cuatro años.

A mi tutor Tomás por estar ahí, cuando le consultaba.

A todos mis amigos por haberme acompañado en el camino.

A mi familia, especialmente a mi abuela, por escucharme y apoyarme durante todo el proceso, guiarme en toda mi carrera y haberme ayudado a ser una mejor persona.



# Resumen

El uso de dispositivos Internet of Things (IoT) está creciendo exponencialmente, esperándose que para el año 2030 haya un total de 30 billones de dispositivos IoT conectados. Uno de los entornos con mayor número de dispositivos IoT es el doméstico, donde hay diversidad de dispositivos con conectividad a Internet, como son luces, termostatos, cámaras, cerraduras, accesorios, sensores (movimiento, temperatura, humedad, calidad del aire), sistemas integrados y microcontroladores. Los dispositivos IoT presentan vulnerabilidades recurrentes (credenciales por defecto, dispositivos inseguros, firmware sin actualizar) que han posibilitado la aparición de botnets masivas que mantienen el ecosistema IoT como objetivo preferente. El uso de honeypots especializados en IoT permite recopilar telemetría real de estos ataques.

Este Trabajo de Fin de Grado presenta el diseño e implementación de una **honeynet orientada a dispositivos IoT en un entorno doméstico**. La honeynet está formada por: (1) honeypots o seúelos que emulan dispositivos propios de entornos IoT domésticos, como son SSH, Telnet, MQTT, MQTTS y IPP, usando honeypots ya existentes de código abierto; y (2) dispositivos IoT reales, como son, una pasarela TRÅDFRI (IP/CoAP a Zigbee), una bombilla LED TRÅDFRI (Zigbee); y un reloj Samsung Galaxy Watch Active con conectividad Wi-Fi y Bluetooth.

El sistema incluye funcionalidades de almacenamiento de logs y trazas de tráfico, y visualización; para ello, se usa una pila de gestión y analítica de logs de código abierto (ELK - Elasticsearch, Kibana, Beats, y Logstash) y una herramienta de análisis de tráfico de red (Wireshark).

El propósito de la honeynet es identificar patrones, entender las tácticas de los atacantes, vulnerabilidades explotadas, así como evaluar la efectividad de las defensas existentes e identificar riesgos para la red doméstica.

**Palabras clave:** Ciberseguridad, Redes IoT, Raspberry Pi, HoneyPot, honeynet, Vulnerabilidad, Amenaza, Vector de Ataque, Cowrie, T-Pot, ELK, filebeat, Wireshark.



# Índice de contenidos

<b>Índice de tablas</b>	<b>X</b>
<b>Índice de figuras</b>	<b>XIII</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Contexto y alcance . . . . .	2
1.2. Objetivos específicos del trabajo . . . . .	2
1.3. Estructura del documento . . . . .	3
<b>2. Estado del Arte</b>	<b>4</b>
2.1. Seguridad en el Ecosistema IoT . . . . .	4
2.1.1. Definición y características del IoT . . . . .	4
2.1.2. Superficie de ataque y vulnerabilidades comunes . . . . .	5
2.1.3. Casos reales de ataques a dispositivos IoT . . . . .	5
2.1.4. Panorama actual de las amenazas . . . . .	6
2.2. Honeypots en entornos IoT . . . . .	6
2.2.1. Concepto y objetivos de los honeypots . . . . .	6
2.2.2. Clasificación por nivel de interacción . . . . .	6
2.2.3. Aplicaciones y estudio de casos . . . . .	7
2.2.4. Limitaciones y consideraciones prácticas . . . . .	8
2.3. Plataformas honeypot para IoT . . . . .	9
2.3.1. Cowrie para SSH/Telnet . . . . .	9
2.3.2. T-Pot y plataformas multi-honeypot . . . . .	9
2.3.3. Otras soluciones relevantes . . . . .	9
<b>3. Metodología y plan de trabajo</b>	<b>11</b>
3.1. Metodología . . . . .	11
3.1.1. Herramientas y materiales . . . . .	11
3.1.2. Diseño experimental . . . . .	13
3.2. Plan de trabajo . . . . .	14
<b>4. Diseño e implementación</b>	<b>16</b>
4.1. Red doméstica . . . . .	16

4.1.1. Diseño de red doméstica . . . . .	16
4.1.2. Configuración de dispositivos de red de la honeynet . . . . .	19
4.2. Honeynet . . . . .	21
4.2.1. Protocolos IoT . . . . .	21
4.2.2. Diseño de la honeynet . . . . .	22
4.2.3. Despliegue y configuración de los honeypots . . . . .	26
4.3. Despliegue de los dispositivos IoT reales . . . . .	27
4.4. Comprobación del funcionamiento de la red . . . . .	30
4.4.1. Aislamiento de subredes domésticas . . . . .	30
4.4.2. Acceso a los honeypots desde la red doméstica . . . . .	32
4.4.3. Acceso externo a los honeypots desde Internet . . . . .	34
<b>5. Recogida de datos y análisis</b>	<b>36</b>
5.1. Shodan . . . . .	37
5.2. Recogida de datos de los honeypots . . . . .	38
5.3. Análisis de datos de los honeypots . . . . .	40
5.3.1. Evolución temporal de los intentos de acceso . . . . .	40
5.3.2. Credenciales de fuerza bruta en Cowrie . . . . .	41
5.3.3. Origen de los ataques . . . . .	42
5.3.4. Comandos y archivos más comunes . . . . .	43
5.3.5. Caso práctico - Análisis malware . . . . .	44
5.4. Recogida de datos de los dispositivos IoT reales . . . . .	46
5.5. Análisis de datos de los dispositivos IoT reales . . . . .	47
5.5.1. Dispositivos TRÅDFRI . . . . .	47
5.5.2. Reloj Samsung Galaxy . . . . .	49
<b>6. Conclusiones y trabajos futuros</b>	<b>52</b>
<b>Bibliografía</b>	<b>53</b>
<b>Apéndices</b>	<b>57</b>
<b>A. Configuración Servidor HP</b>	<b>59</b>
<b>B. Preparación de las Raspberry PIs</b>	<b>60</b>
<b>C. Instalación y configuración de Cowrie</b>	<b>61</b>
<b>D. Instalación y configuración de T-Pot</b>	<b>63</b>



# Índice de tablas

4.1. Requisitos del sistema para instalar T-Pot . . . . .	26
5.1. Volumen diario de eventos en los índices <code>cowrie-logs</code> . . . . .	39
5.2. Conexiones por servicio . . . . .	40
5.3. Archivos descargados y frecuencia de aparición . . . . .	43



# Índice de figuras

4.1.	Red doméstica inicial . . . . .	17
4.2.	Red doméstica final . . . . .	18
4.3.	Red Sercomm. . . . .	19
4.4.	Port mirroring en el TP-Link TL-SG608E . . . . .	20
4.5.	Red inicial del honeynet . . . . .	23
4.6.	Red final del honeynet. . . . .	24
4.7.	Plan de direccionamiento en la honeynet . . . . .	25
4.8.	Clientes DHCP Slate AX. . . . .	25
4.9.	Dispositivos Trådfri . . . . .	27
4.10.	Captura Trådfri con puerto CoAPs . . . . .	28
4.11.	Escaneo de puertos Trådfri . . . . .	28
4.12.	Dispositivos Samsung Galaxy Watch Active . . . . .	29
4.13.	Conectividad Samsung Galaxy Watch Active . . . . .	29
4.14.	Escaneo de puertos reloj . . . . .	30
4.15.	Port forwarding Slate AX . . . . .	32
4.16.	Esquema pruebas de redirección desde red doméstica . . . . .	33
4.17.	Prueba píng y nmap desde red doméstica . . . . .	33
4.18.	Port forwarding Sercomm . . . . .	34
4.19.	Esquema pruebas de redirección desde Internet . . . . .	34
4.20.	Prueba nmap desde Internet . . . . .	35
5.1.	Resultado Shodan (47.62.221.118) para la honeynet . . . . .	37
5.2.	Ejemplo honeypot identificado en Madrid (5.188.181.190) . . . . .	38
5.3.	Comparativa del número de ataques a Cowrie y T-Pot. . . . .	40
5.4.	Credenciales más utilizadas . . . . .	41
5.5.	Comparativa del origen de ataques a Cowrie y T-Pot. . . . .	42
5.6.	Comandos más utilizados . . . . .	43
5.7.	Ejecución en ANY-RUN: procesos, conexiones y alertas. . . . .	44
5.8.	Línea temporal procesos ejecutados por el malware. . . . .	46
5.9.	Ataques CoAPs TRÅDFRI. . . . .	48
5.10.	Ataques mDNS TRÅDFRI. . . . .	48
5.11.	Puertos abiertos televisión Samsung (Fuente: [28]). . . . .	50
5.12.	Ataques a los puertos 8080 y 8081. . . . .	50



# 1

## Introducción

El concepto de Internet de las Cosas (IoT por sus siglas en inglés) se refiere a una red de objetos físicos que están interconectados y que incluyen sensores, actuadores, microcontroladores y otras tecnologías. La idea principal es que estos objetos puedan conectarse e intercambiar datos con otros dispositivos y sistemas. Esto lo hacen a través de una red, que puede ser una red privada (como una red doméstica) o la propia Internet pública, lo que permite que los componentes de la red puedan 'verse' e interactuar entre sí. Los dispositivos que se comunican de esta forma son muy variados, desde los electrodomésticos de casa hasta equipos industriales.

El incremento exponencial en la adopción de dispositivos IoT [1], tanto en el ámbito doméstico como en el empresarial, ha traído consigo un aumento significativo en la incidencia de ciberataques dirigidos hacia estos, convirtiéndolos en un objetivo clave para los atacantes hoy en día [2].

El estudio detallado de las metodologías de ataque empleadas contra los dispositivos IoT es fundamental para prevenir incidentes. El hecho de que haya más ataques a estos dispositivos está relacionado con una idea simple: cuantos más dispositivos y menos seguridad, mayor es la amenaza.

El crecimiento del concepto de “casa inteligente” ha sido un factor muy importante en este aumento. Al integrar aparatos domóticos en las casas, las tareas cotidianas cada vez están más gestionadas por dispositivos automáticos.

Una parte muy importante de los ataques recibidos por estos dispositivos busca crear “zombies”, conocidas como *botnets* [3]. Los atacantes utilizan todos

estos dispositivos infectados para actividades ilegales, como enviar correos no deseados (spam), distribuir malware (virus) o lanzar ataques para entorpecer o bloquear servicios (ataques de denegación de servicio distribuidos o DDoS).

El conocimiento de las técnicas, tácticas y procedimientos (*TTPs*) que usan los atacantes no solo ayuda a predecir y evitar ataques antes de que ocurran, sino que, en el caso de que ya haya ocurrido el ataque, ayuda a reducir el impacto. Por ello, para analizar cómo funcionan los ataques sin poner en riesgo los sistemas finales, el usar honeypots o honeynets es una de las estrategias más efectivas.

Un *honeypot* es un sistema (hardware o software) que deliberadamente está mal configurado con vulnerabilidades y servicios los cuales son aparentemente atractivos para un atacante. Su objetivo es atraer y actuar como señuelo para los atacantes, de manera que, todas sus acciones queden registradas y puedan ser analizadas para mejorar la seguridad en el sistema real.

Cuando muchos honeypots se interconectan forman una red aislada conocida como *honeynet*. Esto ayuda a reproducir escenarios más complejos en los que un atacante podría tratar de hacer movimientos laterales entre los distintos servicios buscando posibles brechas de seguridad.

## 1.1. Contexto y alcance

El objetivo de este TFG es el despliegue de una honeynet IoT en una red doméstica. El honeynet desplegado servirá para hacer un análisis de los comportamientos de los atacantes y así ayudar a definir medidas de seguridad adecuadas. Además, permite identificar los vectores de ataque más utilizados por los atacantes.

## 1.2. Objetivos específicos del trabajo

- **Implementar una honeynet aislada:** La creación de una subred dentro de la red doméstica totalmente independiente y aislada del resto.
- **Capturar tráfico y actividades maliciosas sobre protocolos IoT:** Configurar, monitorizar y registrar los intentos de conexión y los comandos en protocolos como SSH, Telnet, MQTT y CoAP, entre otros.
- **Análisis y conclusiones:** Analizar la información recolectada para identificar patrones de ataque, evaluar la eficiencia de la honeynet como sistema de defensa y extraer conclusiones de como actúan los atacantes y como se podría mejorar la defensa.

### 1.3. Estructura del documento

La estructura del TFG es la siguiente:

- **Capítulo 2: Estado del arte**

Se presenta el estado del arte y la fundamentación académica del trabajo, incluyendo:

- Definición y características del IoT.
- Superficie de ataque y vulnerabilidades comunes.
- Concepto, objetivos y clasificación de honeypots.

- **Capítulo 3: Metodología y Plan de Trabajo**

Se describen las herramientas y materiales utilizados, el proceso que llevó al diseño experimental del proyecto, la información de la recolección de los datos, así como el procedimiento paso a paso que se ha llevado a cabo para el objetivo.

- **Capítulo 4: Diseño e Implementación**

Se detalla el despliegue de la red doméstica aislada y la honeynet:

- Arquitectura de red y configuración de dispositivos.
- Despliegue de honeypots.
- Verificación del funcionamiento mediante pruebas de conectividad y escaneo.

- **Capítulo 5: Recogida de Datos y Análisis**

Se explica cómo se capturaron, almacenaron y analizaron tanto los logs como las trazas del tráfico de red:

- Visualización en escaner público Shodan.
- Análisis de los honeypots señuelo: Cowrie y T-Pot.
- Análisis de los dispositivos IoT reales.

- **Capítulo 6: Conclusiones y trabajos futuros**

Se sintetizan los resultados obtenidos, se evalúa si se han conseguido los objetivos propuestos, y se plantean posibles mejoras y líneas de investigación para el futuro.

# 2

## Estado del Arte

Los dispositivos del *Internet of Things* (IoT) en entornos domésticos, como cámaras de seguridad y electrodomésticos inteligentes, requieren medidas específicas de seguridad para evitar que introduzcan amenazas en la red. Debido a su conectividad, estos dispositivos son susceptibles a ataques remotos y la explotación de vulnerabilidades, lo que puede derivar en robo de datos, ataques DDoS o compromisos a otros sistemas conectados.

La seguridad en IoT es especialmente compleja, ya que muchos dispositivos se diseñan priorizando la funcionalidad y la facilidad de uso por encima de la protección, lo que incrementa los riesgos. Dada su creciente presencia en la vida cotidiana y en entornos empresariales, tanto usuarios como organizaciones deben afrontar los desafíos de seguridad asociados al IoT.

En este capítulo se presenta un estado del arte de diferentes aspectos de la seguridad en este tipo de entornos.

### 2.1. Seguridad en el Ecosistema IoT

#### 2.1.1. Definición y características del IoT

De forma general, el IoT se define como la red de objetos físicos conectados que recopilan e intercambian datos sin intervención humana directa. Esta conexión entre los dispositivos aporta buenas funcionalidades, pero puede suponer una

amenaza al ser entornos con recursos limitados.

- **Limitaciones de hardware:** CPU, memoria y almacenamiento reducidos impiden desplegar medidas de defensa más robustas y seguras.
- **Dispositivos *headless*:** como carecen de interfaz gráfica necesitan ser configurados de manera remota. En muchos casos se omite la configuración y simplemente se instala, lo que supone que se quedan con las credenciales por defecto o incluso puertas traseras.
- **Longevidad y falta de parches:** muchos de estos dispositivos no reciben actualizaciones por parte de las empresas o acaban con firmware desactualizado por obsolescencia. Las actualizaciones son muy importantes ya que las empresas arreglan vulnerabilidades a lo largo del tiempo.

### 2.1.2. Superficie de ataque y vulnerabilidades comunes

Los ataques más frecuentes son los siguientes:

- **Fuerza bruta sobre SSH/Telnet:** uso de contraseñas triviales o credenciales por defecto que por prueba y error el atacante acaba descubriendo. Un ejemplo es que durante el desarrollo de este proyecto se detectó que uno de los routers seguía teniendo la contraseña admin:admin.
- **Exploits de firmware desactualizado:** vulnerabilidades conocidas (CVE) que permanecen sin parchear en dispositivos domésticos. Un ejemplo claro es el que se cita en [4] en el que se detecta el CVE-2017-9765 que sufría una vulnerabilidad por desbordamiento de buffer que es un fallo muy típico en la programación.
- **Servicios de mensajería inseguros (MQTT):** brokers mal configurados, sin TLS (cifrado que funciona sobre TCP) ni autenticación, susceptibles a secuestro de sesiones y filtrado de datos.

### 2.1.3. Casos reales de ataques a dispositivos IoT

Aunque en la literatura se describen muchos casos reales de ataques, aquí se han elegido solo dos de los más relevantes:

- **Botnet Mirai (2016):** es un malware que afecta a dispositivos inteligentes y los convierte en una red de bots o zombies infectados. Mirai escanea Internet en busca de dispositivos IoT que ejecutan un procesador ARC el cual ejecuta una versión simple del sistema operativo de Linux. En el caso

de que utilice una combinación de login y password por defecto, Mirai puede entrar e infectarlo [5].

- **VPNFilter (2018):** es un malware diseñado para infectar routers. Este malware no solo roba datos personales sino que además implementa un “kill switch”, un sistema para destruir remotamente un router. Está específicamente diseñado para atacar dispositivos de redes que utilizan Modbus, protocolo muy utilizado en dispositivos IoT industriales [6].

#### 2.1.4. Panorama actual de las amenazas

Las variantes de Mirai continúan evolucionando y extienden su objetivo a dispositivos Bluetooth Low Energy, redes Zigbee y cámaras IP, formando botnets capaces de atacar servicios críticos a gran escala [7].

## 2.2. Honeypots en entornos IoT

### 2.2.1. Concepto y objetivos de los honeypots

Un honeypot es un sistema informático que se ”sacrifica” para atraer ciberataques, como un señuelo. Este simula ser un objetivo vulnerable para los hackers y los intentos de intrusión se utilizan para obtener información sobre los cibercriminales. Además, también sirve para distraer a los atacantes de otros objetivos y mantener a los atacantes lejos de los sistemas más importantes [8].

En resumen, la seguridad de red honeypot está diseñada para atraer a atacantes a entornos de red falsos para [9]:

1. Ver lo que quieren.
2. Analizar cómo intentan cumplir sus objetivos.
3. Aprender a detenerlos.

### 2.2.2. Clasificación por nivel de interacción

La selección de un honeypot adecuado a los requerimientos específicos resulta crucial para la optimización de los resultados obtenidos. A tal efecto, es preciso considerar las diferentes tipologías existentes:

- **Honeypots de alta interacción:**

Consisten en sistemas que operan con servicios reales instalados. Se requiere su aislamiento completo de la red productiva para prevenir que una intrusión exitosa comprometa la red organizacional. La inteligencia generada por estos sistemas es de gran detalle. Además, es muy importante que se intente ocultar en la mayor medida posible que es un señuelo, ya que toda la información es crucial para poder analizar posibles vectores de ataque de los atacantes.

- **Honeypots de interacción media:**

Implementan un conjunto limitado de servicios fundamentales, tales como servidores web o FTP, programados para ofrecer respuestas predefinidas a las interacciones del atacante. Aparte de estas interacciones elementales, los servicios son emulados y no son funcionales en su totalidad, impidiendo así el acceso real al sistema subyacente por parte del atacante. La información obtenida puede ser de menor valor que la que te pueda dar uno de alta interacción, pero también requiere de menos mantenimiento.

- **Honeypots de interacción baja:**

Emulan únicamente un subconjunto de servicios de red elementales, como la conectividad TCP/IP o ICMP. La información recolectada está muy limitada a cosas como escaneos de la red donde se detecta el volumen de tráfico que podrían sufrir los dispositivos en caso de ser vulnerables.

### 2.2.3. Aplicaciones y estudio de casos

En este apartado se describen algunas de las maneras en las que los honeypots pueden ser utilizados:

- **Honeypot de malware:** utilizan recursos atractivos ya que son conocidos por atraer malware. Pueden, por ejemplo, emular un USB para que cuando un ordenador sufra un ataque, el malware ataque al USB.
- **Honeypot de correo no deseado:** simula una dirección de correo trampa que finge estar mal protegido para llamar la atención de los atacantes. Así cuando un spammer lo usa para mandar mensajes de prueba, el honeypot registra la IP y el contenido de lo que es enviado y así generar listas negras que bloquean y protegen a los usuarios reales.
- **Honeypot de bases de datos:** utiliza una base de datos señuelo para identificar ataques de inyección de SQL. Y luego, evitarse implementando un firewall de base de datos.
- **Honeypot de cliente:** como cliente en Internet se puede acceder a servidores maliciosos. Los honeypot de cliente intentan atraer a servidores

maliciosos que los atacantes utilizan para piratear a los usuarios. Se ejecuta en un entorno virtualizado y tienen protecciones de contención para reducir el riesgo de los que están investigando.

- **Honeynet:** conjunto de honeypots interconectados que permite estudiar ataques distribuidos (DDoS, ransomware). Si bien se utiliza para estudiar diferentes tipos de ataques, este tráfico está totalmente separado de los sistemas de la organización para proteger los sistemas importantes.

#### 2.2.4. Limitaciones y consideraciones prácticas

Aunque los honeypots aportan información muy valiosa para entender el comportamiento de los ataques, también es importante conocer sus limitaciones y requerimientos de mantenimiento.

- **Cobertura parcial de ataques:** solo detecta la actividad dirigida específicamente contra él, es decir, no alerta sobre ataques que podría estar sufriendo el sistema real.
- **Riesgo de detección y evasión:** los atacantes son muy conscientes que podrían estar en un honeypot por lo que si detectan que no es un sistema real, podrían ignorarlo o dedicarse a distraer al equipo de seguridad con falsos positivos.
- **Possible plataforma de pivotaje:** un honeypot mal configurado puede ser una ayuda para un atacante para entrar en un sistema real, especialmente si las reglas de firewall no son estrictas.
- **Consumo de recursos y mantenimiento:** aunque en algunos casos son ligeros en hardware como una Raspberry Pi, requieren mantenerse actualizados para evitar vulnerabilidades ya corregidas en los parches.
- **Dependencia de configuración:** que un sistema funcione bien o no, depende mucho del realismo que tenga. Ser capaz de replicar los servicios al máximo ayuda a engañar a los atacantes.
- **Complemento, no sustituto:** a pesar de la valiosa información que proporciona, no sustituye a una defensa real en profundidad. En ningún caso reemplaza a los firewalls ni a ninguna solución de protección de seguridad.

## 2.3. Plataformas honeypot para IoT

### 2.3.1. Cowrie para SSH/Telnet

Cowrie es un honeypot muy popular que emula un servidor SSH y Telnet real. Su funcionamiento es muy sencillo de entender ya que ”finge” que es un servidor Linux [10]. Al igual que muchos dispositivos que se controlan remotamente, tiene una terminal donde puedes “tirar” comandos y interactuar con carpetas y archivos con información supuestamente real. Su funcionamiento es fácil de entender ya que cuando un atacante intenta conectarse por SSH (puerto 22) o Telnet (puerto 23), Cowrie registra la información de acceso (usuario, contraseña, comandos, dirección IP, etc.). Toda esta información se almacena en un registro en el cual se pueden leer con detalle las credenciales que utilizan, las cosas que intentan descargar, todo esto sin necesidad de exponer un sistema de verdad.

### 2.3.2. T-Pot y plataformas multi-honeypot

T-Pot es un conjunto de honeypots agrupados que funcionan en un solo dispositivo [11]. Gracias a usar contenedores Docker, cada uno de ellos funciona de forma aislada y se inicia por separado del resto. El gran beneficio de T-Pot es su fácil despliegue y que con una sola instalación se pueden capturar muchos vectores de ataque. El mayor problema es el nivel computacional que requiere, ya que emula una lista enorme de protocolos, y cada uno con sus configuraciones y niveles de interacción.

### 2.3.3. Otras soluciones relevantes

Además de Cowrie y T-Pot, existen otras plataformas útiles:

- **RioTPot**: plataforma diseñada específicamente para entornos IoT basados en sistemas operativos de tiempo real (RTOS) [12]. La ventaja es que emula nodos de red de bajo consumo como los dispositivos IoT, además CoAP. Recomendable para investigar amenazas en sensores y actuadores con recursos limitados. El problema es la falta de parches de actualización lo que dificulta su instalación.
- **Conpot**: emula dispositivos SCADA lo que supone que está más orientado para procesos industriales a distancia, como fábricas y plantas eléctricas, y no para entornos domésticos [13].
- **Dionaea**: honeypot de baja interacción centrado en la captura de malware de red. A pesar de su baja interacción es ideal para la captura de malware de red

a ser analizado en un sandbox [14]. Emula servicios SMB, HTTP y FTP, y atrae y almacena pruebas de exploit, y binarios maliciosos.

Cada una de estas soluciones aporta un valor distinto al panorama de las amenazas. Con su combinación se puede sacar información muy interesante, y la elección de la combinación más adecuada depende de los protocolos o comportamientos que se quieran analizar.

# 3

## Metodología y plan de trabajo

El presente capítulo detalla la metodología empleada en el trabajo, describiendo el enfoque, los procedimientos y las herramientas utilizadas para abordar los objetivos planteados. Asimismo, se expone el plan de trabajo diseñado para organizar las diferentes etapas del estudio, para así asegurar la obtención de los resultados previstos.

### 3.1. Metodología

#### 3.1.1. Herramientas y materiales

Para el despliegue y monitorización de la honeynet, y el análisis de resultados, se han empleado los recursos expuestos a continuación.

##### ■ **Hardware usado:**

- Router GL.iNet Slate AX con sistema operativo OpenWrt.
- HP-Server con Kali Linux.
- Lenovo Legion 5 con Windows (realización de pruebas).
- Dell XPS 13 9370 con T-Pot.
- Raspberry Pi 3b+ con Cowrie (pi-1).
- Raspberry Pi 3b+ usada para pruebas (pi-2).

- Raspberry Pi 3b+ con Wireshark (pi-3).
- Switch gestionable TP-Link TL-SG608E.
- Dispositivos IoT reales: IKEA TRÅDFRI, bombilla Zigbee y pasarela, y reloj Samsung Galaxy Watch Active.
- Teléfono móvil OPPO usado para acceder a los dispositivos IoT reales usando la correspondiente aplicación.

■ Herramientas usadas:

• **Análisis de datos**

Para el análisis de datos se usa el stack ELK (Elasticsearch, Logstash y Kibana). Este stack incluye tres proyectos de código abierto muy populares: Elasticsearch, Logstash y Kibana, que de forma conjunta permiten recopilar, almacenar y visualizar en tiempo real gran volumen de datos de una manera muy visual. En concreto:

- *Elastisearch* se encarga del almacenamiento y utilización de los datos obtenidos.
- *Logstash* es la tubería que permite el envío de datos desde muchas fuentes.
- *Kibana* ofrece la visualización y las tablas para facilitar el análisis.

• **Broker MQTT – Mosquitto**

Servidor ligero para publicar/suscribir mensajes usando los protocolos MQTT y MQTTS. Usado para realizar pruebas.

• **Agente de logs – Filebeat**

Agente que recoge los ficheros de log y los envía hacia Elasticsearch. Muy ligero y fácil de configurar, útil para el envío de datos desde una Raspberry Pi.

• **Análisis de tráfico – Wireshark**

Captura y análisis de los paquetes de la red, permite filtrar por protocolos y visualizar fácilmente los datos por capas.

• **Docker / Docker Compose**

Contenerización y manejo de honeypots y servicios. Muy eficiente al poder reiniciar todo con un único comando.

• **Python 3**

Entorno para gestión de paquetes de Cowrie.

• **Utilidades de diagnóstico**

- **ping**: comprobación básica para ver la conectividad entre nodos de la red.
- **nmap**: escaneo de puertos y detección de servicios [15].
- **tcpdump**: captura de tráfico en bruto para la realización de pruebas y análisis de red.

- **ss / netstat:** identificación de sockets abiertos y procesos asociados.
- **traceroute:** seguimiento de la ruta de los paquetes hasta un destino.

### 3.1.2. Diseño experimental

Para la experimentación se usa un entorno controlado (honeynet) en el que se han instalado y configurado varios honeypots (Cowrie y T-Pot). La configuración de dichos honeypots se ha personalizado para adaptarla a los servicios IoT que se desean simular (telnet, ssh, MQTT y MQTTS). Adicionalmente, se han conectado a la honeynet dos dispositivos IoT reales, un hub IKEA TRÅDFRI CoAP/IP que controla una bombilla Zigbee, y un reloj Samsung con interfaces bluetooth y WiFi, lo que posibilita que los atacantes puedan intercambiar tráfico con dispositivos IoT reales.

La fase de captura de datos ha durado 10 días en los que se ha dado acceso desde Internet a la honeynet, los datos de los ataques se han ido almacenando en logs y ficheros JSON para su posterior visualización y análisis. Por otro lado, se ha usado la capacidad de puerto espejo del conmutador de la honeynet para copiar todo el tráfico del puerto WAN (puerto que da acceso a Internet) a otro puerto, permitiendo capturar el tráfico de red intercambiado con Internet para su posterior análisis.

Para el análisis de datos se han empleado un conjunto de herramientas especializadas. Los logs generados se han procesado mediante la pila **ELK** (Elasticsearch, Logstash y Kibana), que han permitido visualizar en tiempo real los intentos de conexión, credenciales utilizadas, geolocalización de los atacantes y archivos descargados. En paralelo, el tráfico de los dispositivos reales ha sido inspeccionado con **Wireshark**, aplicando los filtros por puertos, protocolos y respuestas para así poder analizar los patrones de comunicación y posibles anomalías. Adicionalmente, se han analizado los binarios descargados y se han observado las muestras mediante herramientas de sandbox como **ANY.RUN**.

## 3.2. Plan de trabajo

El plan de trabajo se estructura en seis fases clave, cada una orientada a ir avanzando progresivamente y de manera ordenada para conseguir los objetivos definidos en el proyecto:

### 1. Búsqueda y validación de la idea del honeypot.

- Investigación en la literatura académica y en proyectos existentes para adquirir la base teórica del trabajo.
- Definición del objetivo: construir una honeynet doméstica orientada a capturar ataques contra los protocolos IoT.

### 2. Diseño conceptual y selección de las herramientas.

- Elaboración de la arquitectura de la red aislada (doble NAT) y asignación de los rangos IP a cada subred.
- Selección de los protocolos más relevantes en IoT (CoAP, MQTT, etc), elección de honeypots y búsqueda de dispositivos sueño (bombilla Zigbee, smartwatch).

### 3. Despliegue e implementación.

- Configuración del servidor central (HP-Server con Kali Linux).
- Montaje de la subred honeynet en el router Slate AX y routers secundarios.
- Captura con Wireshark de protocolos utilizados por TRÅDFRI y smartwatch.
- Instalación de las Raspberry Pi con Cowrie y T-Pot.

### 4. Verificación y ajustes.

- Pruebas de aislamiento entre las subredes con ping y traceroute.
- Prueba del redireccionamiento de puertos para acceso remoto al honeynet dentro de la subred.
- Prueba desde Internet.

### 5. Captura de datos.

- Inicio de recolección de logs.
- Monitorización de almacenamiento, comprobación de errores y reajustes de honeypots.

### 6. Análisis y revisión.

### Capítulo 3. Metodología y plan de trabajo

---

- Búsqueda de los eventos más relevantes creando visualizaciones con Kibana.
- Conclusiones de los resultados obtenidos, redacción y discusión basado en datos reales.

# 4

## Diseño e implementación

Este capítulo presenta en primer lugar, el diseño y configuración de la red doméstica en la que se ha desplegado la honeynet. El rediseño de la red doméstica ha tenido como objetivo proteger los equipos ya existentes, frente a los posibles atacantes de la honeynet. A continuación, se explica el diseño y configuración de la propia honeynet, justificando las elecciones de diseño adoptadas a lo largo del proceso.

### 4.1. Red doméstica

En este apartado se presenta cómo era la red doméstica inicial (red en el domicilio habitual del estudiante) y la red doméstica final, que incluye una subred aislada del resto a la que se conectarán los distintos señuelos (honeynet).

#### 4.1.1. Diseño de red doméstica

##### Red doméstica inicial

La red doméstica existente (ver figura 4.1) tiene la configuración típica en una casa con acceso a Internet. Se dispone de un contrato de fibra óptica de tipo FTTH (Fiber To The Home) a 600 Mbps con una compañía proveedora de servicios de Internet (ISP - Internet Service Provider). La fibra óptica termina

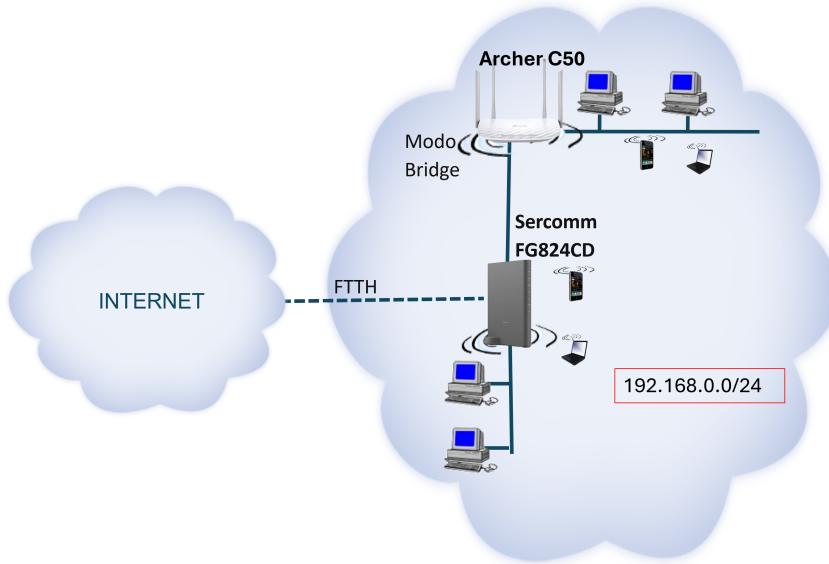


Figura 4.1: Red doméstica inicial

en un router Sercomm FG824CD que da acceso a la red hogar y está ubicado en la planta de abajo. Este router Sercomm usa un cliente DHCP (Dynamic Host Configuration Protocol) para configurarse una dirección IP pública desde un servidor DCHP en el proveedor. Con el fin de ampliar la cobertura Wi-Fi, hay instalado un segundo router (Archer C50) configurado en modo bridge. Así, dentro de la red doméstica se usa direccionamiento privado y todos los equipos pertenecen a la misma subred IP. Un servidor DHCP, en el router Sercomm, se encarga de asignar las direcciones IP dentro de la red doméstica usando el prefijo 192.168.0.0/24.

### Diseño final red doméstica

Posteriormente, se realiza la segmentación de la red para aislar el tráfico, configurando varias subredes IP. El objetivo del nuevo diseño es separar el tráfico doméstico del de la honeynet para evitar ataques hacia los familiares que teletrabajan.

La figura 4.2 muestra que la red final está formada por cuatro subredes IP: la subred del Sercomm (prefijo 192.168.0.X/24), la subred de la planta de arriba (prefijo 192.168.30.X/24) a la que se accede por el router TL-WR841N, la subred de la planta de abajo (prefijo 192.168.32.X/24) a la que se accede por el router Archer C50, y la honeynet (prefijo 192.168.40.X/24) a la que se accede por un nuevo router (Slate AX).

La segmentación entre redes se logra usando una configuración de **doble NAT**

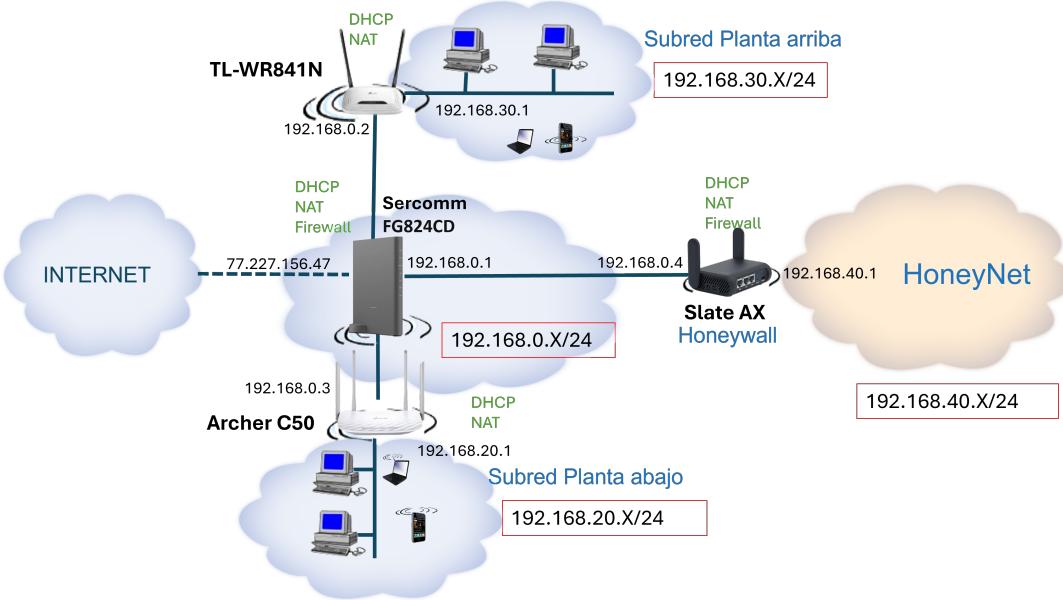


Figura 4.2: Red doméstica final

(Network Address Translation) con dos dispositivos realizando la función de traducción de direcciones: Sercomm y TL-WR841N para acceder a la subred de la planta de arriba; Sercomm y Archer C50 para acceder a la subred de la planta de abajo; Sercomm y Slate AX a la honeynet. Esta configuración de doble NAT no permite establecer conexiones entre dispositivos de distintas subredes de la red doméstica; de este modo, desde la honeynet no se puede acceder a las subredes de la planta de arriba y de la planta de abajo, quedando de este modo aisladas.

Para conseguir esta configuración, los routers de las plantas de arriba y abajo disponen de un servicio NAT, todas las comunicaciones de los dispositivos de la planta de arriba comparten la dirección 192.168.0.2, y todas las comunicaciones de los dispositivos de la planta de abajo comparten la dirección 192.168.0.3. Adicionalmente, el Sercomm, el TL-WR841N, y el Archer C50 disponen de un servidor DHCP para asignar a los equipos de la subred correspondiente, direcciones de los prefijos asignados. Para facilitar la apertura de puertos, las reglas del firewall y el análisis de las capturas de tráfico a los dispositivos siempre se les asigna la misma dirección IP en función de su MAC (ver la figura 4.3 para el caso del router Sercomm).

Un último aspecto a destacar es que para permitir el acceso de los atacantes desde Internet a la honeynet, en el router Sercomm se abren los puertos correspondientes a los servicios IoT actuando de señuelo, y dicho tráfico se redirige al router de acceso a la honeynet, Slate AX con dirección 192.168.0.4.



Figura 4.3: Red Sercomm.

#### 4.1.2. Configuración de dispositivos de red de la honeynet

Para la implementación de la honeynet se dispone de los siguientes dispositivos de red:

- **Router GL.iNet Slate AX 1800**

Este router ha sido elegido por su alto grado de personalización y compatibilidad nativa con OpenWrt [16]. Los modelos TL-WR841N y Archer C50, que eran de los que se disponía inicialmente, llevan el firmware de fábrica de TP-Link que tiene una capacidad de modificación muy limitada. Se llegó a valorar instalar OpenWrt en dichos equipos, pero finalmente se descartó por el riesgo de *brick* y la complejidad del proceso de *flasheo*.

*Aspectos más relevantes de la configuración del OpenWrt:*

- **dnsmasq**: incluye servidores DNS y DHCP ligeros y entre otras funcionalidades permite reservar una IP estática a cada dispositivo según su MAC [17]. Así, cada equipo mantiene siempre la misma dirección sin tener que configurarla manualmente.
- **firewall nftables**: permite la definición de las zonas LAN y WAN, además de mejorar la protección de otros dispositivos gracias a su política DROP, minimizando así la superficie expuesta.

- LuCI: es la ventana gráfica de OpenWrt que tiene, entre otras, pestañas para la visualización de datos del firewall, y del routing.
- Redirección de puertos: se utiliza para abrir los puertos y mandar el tráfico hacia los señuelos y de este modo poder exponer los dispositivos IoT.

#### ■ Switch TP-Link TL-SG608E

Este switch gestionable con 8 puertos Gigabit ha sido utilizado para poder monitorizar el tráfico interno de la honeynet y simplificar la captura de los distintos paquetes de la red.

- *Port Mirroring*: esta funcionalidad ha permitido duplicar el tráfico de los puertos conectados a los honeypots [18] y dispositivos IoT, hacia una Raspberry Pi (pi-3) dedicada al almacenamiento y análisis de tráfico. De este modo se ha concentrado la captura del tráfico en un único punto (ver figura 4.4).

Port Mirror	Mirroring Port
Enable	Port 4

Apply

Mirrored Port	Ingress	Egress
Port 1 Port 2 <b>Port 3</b> Port 4 Port 5		

Apply      Help

Mirrored Port	Ingress	Egress
Port1	Enable	Enable
Port2	Disable	Disable
Port3	Enable	Enable
Port4	Disable	Disable
Port5	Disable	Disable
Port6	Disable	Disable
Port7	Disable	Disable
Port8	Disable	Disable

Figura 4.4: Port mirroring en el TP-Link TL-SG608E

## 4.2. Honeynet

### 4.2.1. Protocolos IoT

Para abarcar las principales superficies de ataque en entornos IoT, se analizaron e implementaron los siguientes protocolos:

- **SSH (TCP 22) y Telnet (TCP 23)**

Los protocolos de acceso remoto más conocidos y utilizados en dispositivos y gateways IoT son:

- *SSH*: utiliza un canal cifrado ya que encripta todo el tráfico para evitar secuestros de sesión y otro tipo de ataques. Muy a menudo se encuentran credenciales por defecto o incluso vulnerabilidades por no tener los dispositivos actualizados.
- *Telnet*: no utiliza un canal cifrado y es más normal en dispositivos más antiguos lo que lo hace un buen sueño ya que teóricamente es más vulnerable.

- **CoAP / CoAPs (UDP 5683 / 5684)**

*Constrained Application Protocol* o *Protocolo de Aplicación Constringente* está diseñado para ser utilizado por dispositivos con recursos limitados. A diferencia de HTTP, CoAP funciona sobre UDP, lo que implica que reduce mucho el tráfico en la red y permite que se aprovechen mejor los recursos disponibles [19]. En concreto:

- *CoAP*: utiliza UDP, aun así logra cumplir con servicios que comparte con HTTP, como son los métodos GET, POST, PUT y DELETE.
- *CoAPs*: variante segura con DTLS [20], el cual permite, a las comunicaciones basadas en protocolos de datagramas (UDP) comunicarse de una manera más segura.

- **MQTT / MQTTs (TCP 1883 / 8883)**

Protocolo de publicación/suscripción muy utilizado para telemetría, es decir, para el envío de datos desde dispositivos remotos hacia un servidor central [21]. En concreto:

- *MQTT*: conexión ligera, sin cifrado por defecto, tiene el riesgo de sniffing de datos y credenciales.
- *MQTTs*: sobre TLS (canal cifrado), en la honeynet se simula un broker inseguro el cual es un intermediario entre los publicadores que envían los datos y los subscriptores que son los que reciben los datos.

- **Zigbee**

Crea redes de área personal (PAN) de corto alcance, y permite la comunicación entre dispositivos de bajo consumo en la que cada uno de los dispositivos actúa como un repetidor de señal para así expandir el alcance de la red.

- **IPP (TCP 631)**

También conocido como Protocolo de Impresión por Red (Internet Printing Protocol) está presente en dispositivos multifunción IoT. Permite identificar dispositivos expuestos con servicios de impresión o escaneo que pueden servir de puerta trasera.

- **mDNS (UDP 5353)**

En las redes locales se utiliza este servicio para anunciararse y para descubrir otros dispositivos IoT sin la necesidad de un servidor DNS [22].

- **Protocolos propietarios**

Muchos dispositivos IoT utilizan protocolos propietarios, es decir, no estandarizados y que están diseñados por un fabricante o grupo de fabricantes.

#### 4.2.2. Diseño de la honeynet

Para la honeynet se han configurado distintos dispositivos (reales o emulados), cada uno de ellos diseñado para exponer alguno o algunos de los distintos protocolos IoT. A continuación se describen cada uno de los componentes principales que la forman:

- **Cowrie**

Honeypot de interacción media-alta para simular SSH y Telnet. Este simula un dispositivo con acceso remoto mal configurado el cual tiene un sistema de UNIX con un usuario HP-Server simulado y otro con información "personal" llamado fernando. Todos los logs se registran en formato JSON donde se incluyen los intentos de conexión, los comandos ejecutados y los ficheros que son transferidos.

- **T-Pot**

Plataforma multi-honeypot que integra numerosos honeypots (Dionaea, Glutton, Conpot, etc) en contenedores docker lo que facilita el uso de los que se necesitan. Con esta plataforma se emulan distintos servicios IoT, exponiendo MQTT, MQTTS y IPP que simulan brokers mal configurados.

- **Dispositivos TRÅDFRI**

Dispositivos de IKEA, uno actuando como pasarela Zigbee/IP-CoAP y la correspondiente bombilla Zigbee. Inicialmente tras capturar paquetes del encendido y apagado de la luz desde el móvil se ha visto en las trazas que

se usan los puertos 68, 5353 y 5684, lo que revela que hay únicamente dos servicios UDP activos en la pasarela: 5353/udp (mDNS) y 5684/udp (CoAP seguro sobre DTLS) y un cliente 68/udp (cliente DHCP).

#### ■ Reloj Samsung Galaxy Active

El smartwatch de Samsung tiene conectividad tanto por Bluetooth Low Energy (BLE) como por Wi-Fi, lo que le permite sincronizar notificaciones con el teléfono móvil. Durante la fase de análisis inicial se capturaron paquetes dirigidos hacia Internet.

### Primera versión de honeynet

La red quedó inicialmente tal y como muestra la figura 4.5. El router Slate AX actúa como firewall, NAT y como servidor DHCP para la asignación de direcciones de la subred 192.168.40.X/24.

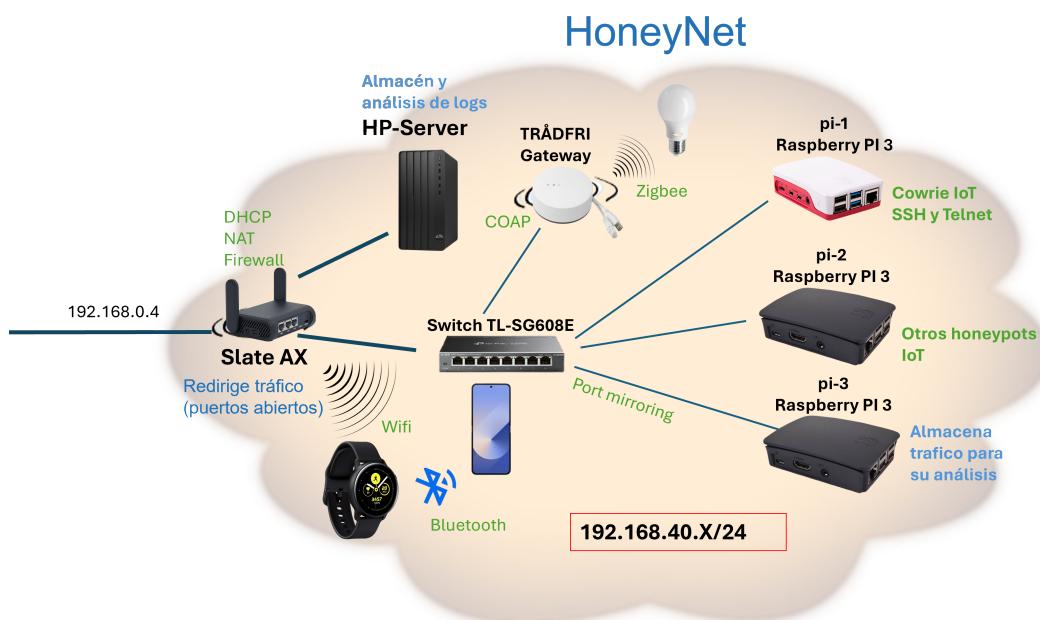


Figura 4.5: Red inicial del honeynet

En esta primera red se configuró la pi-1 con Cowrie, que funcionó adecuadamente. La idea inicial era usar la pi-2 para configurar otros honeypots. En concreto, se intentó instalar y configurar Dionaea y RiotPot. De este intento se concluyó que: (1) Dionaea y RiotPot estaban mal documentados y desactualizados, por lo que no eran una buena opción; (2) T-Pot parecía una mejor opción por la documentación y soporte; sin embargo, la pi-2 no disponía del nivel computacional adecuado para dicho honeypot. Consecuentemente, se instaló T-Pot en un ordenador, como veremos en el diseño final.

## Diseño final del honeynet

El resultado final (ver figura 4.6) es muy similar al planteado inicialmente, excepto por el cambio al Dell Inc. XPS 13 9370, el cual era si podía ejecutar T-Pot, sin ser un problema los recursos utilizados.

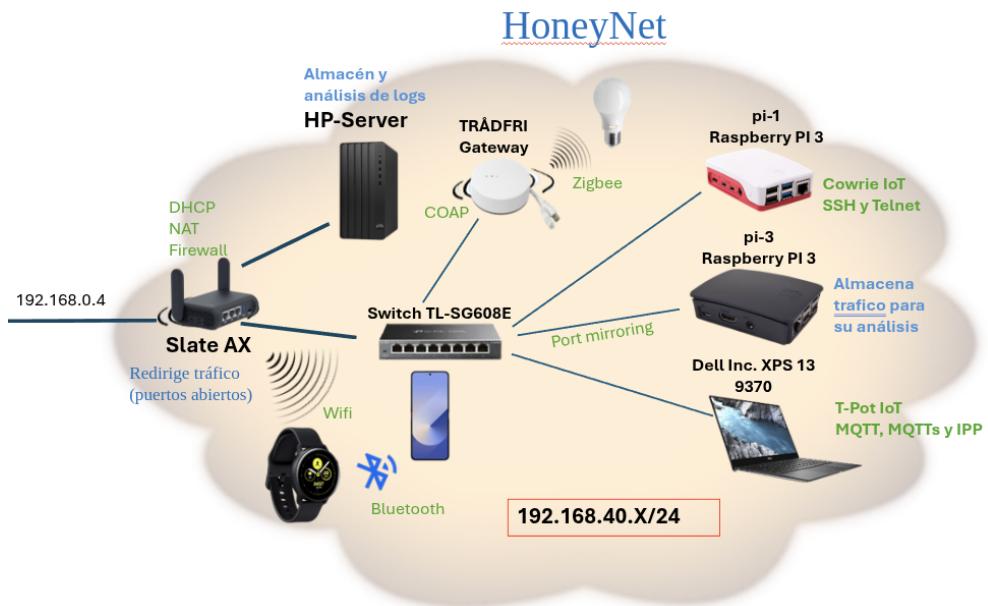


Figura 4.6: Red final del honeynet.

Los aspectos clave de esta arquitectura son: (1) el HP-Server desde donde se pueden controlar las pis (pi-1 y pi-3), además de visualizar los datos de Cowrie con ELK; (2) la pi-3 se encarga de capturar y almacenar con Wireshark todo el tráfico que pasa por la interfaz del WAN del switch (interfaz hacia Internet); (3) el reloj usa su interfaz Wi-Fi para intercambiar datos con Internet (IoT AWS); (4) el móvil puede controlar, utilizando las correspondientes aplicaciones, el reloj y el Trådfri; para controlar el reloj, usa su interfaz Bluetooth y para controlar el Trådfri, su interfaz Wi-Fi.

Cabe recalcar que el hecho de que todas las interfaces de los dispositivos tuvieran direcciones IP estáticas ayudó al desarrollo al saber siempre la dirección asignada a cada dispositivo (ver figura 4.8). La figura 4.7 muestra el plan de direccionamiento en la honeynet, direcciones asignadas mediante DHCP por el router de la subred.

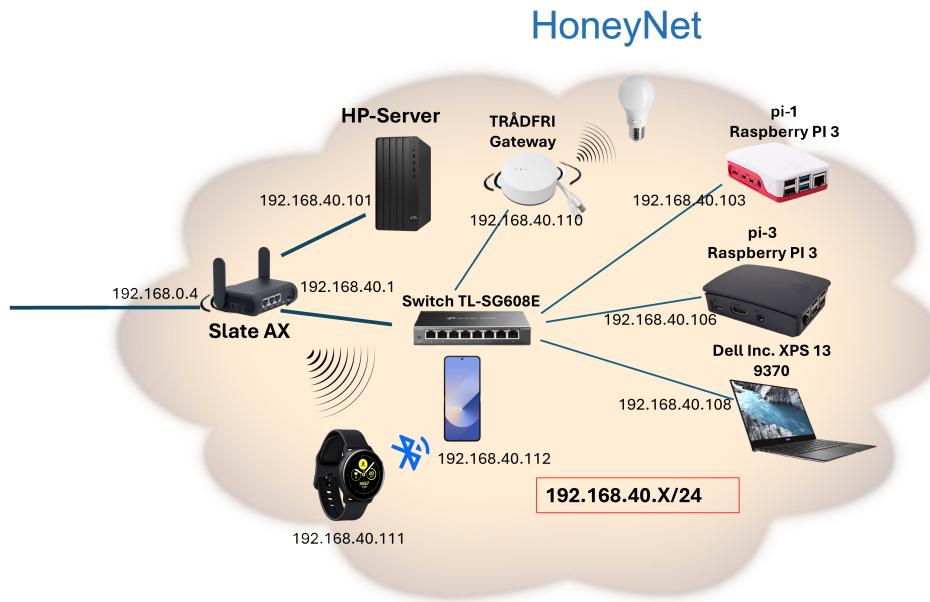


Figura 4.7: Plan de direccionamiento en la honeynet

Este panel de administración muestra la lista de clientes conectados a través del router Slate AX. Los datos capturados son:

Nombre	IP + MAC	Velocidad	tráfico	Bloquear	Acción
Switch TL-SG608E	192.168.40.100 8C:90:2D:9D:1E:92	↓ 0.00 B/s ↑ 0.00 B/s	↓ 0.00 B ↑ 647.58 KB	OFF	...
HP-Server	192.168.40.101 9C:7B:EF:3C:0E:E8	↓ 749.00 B/s ↑ 1.59 KB/s	↓ 583.48 MB ↑ 991.85 MB	OFF	...
raspberrypi-1	192.168.40.103 B8:27:EB:4F:2C:0C	↓ 1.00 B/s ↑ 6.00 B/s	↓ 2.10 MB ↑ 7.70 MB	OFF	...
raspberrypi-1	192.168.40.103 B8:27:EB:1A:79:59	↓ 0.00 B/s ↑ 0.00 B/s	↓ 1.25 MB ↑ 1.14 MB	OFF	...
raspberrypi-3	192.168.40.106 B8:27:EB:AE:2B:C8	↓ 0.00 B/s ↑ 0.00 B/s	↓ 573.93 KB ↑ 629.68 KB	OFF	...
raspberrypi-3	192.168.40.107 B8:27:EB:FB:7D:9D	↓ 0.00 B/s ↑ 0.00 B/s	↓ 4.30 KB ↑ 680.39 KB	OFF	...
Dell-Server	192.168.40.108 34:99:71:E8:EA:BA	↓ 44.00 B/s ↑ 28.00 B/s	↓ 333.57 MB ↑ 120.38 MB	OFF	...
Luz Zigbee	192.168.40.110 44:91:60:2B:E9:C5	↓ 0.00 B/s ↑ 0.00 B/s	↓ 5.55 MB ↑ 3.07 MB	OFF	...
GalaxyWatchActive-D90D	192.168.40.111 DC:F7:56:D5:D9:0E	↓ 0.00 B/s ↑ 0.00 B/s	↓ 29.96 KB ↑ 80.50 KB	OFF	...

Figura 4.8: Clientes DHCP Slate AX.

### 4.2.3. Despliegue y configuración de los honeypots

En esta sección se explica cómo se han instalado y configurado los honeypots conectados a la honeynet. El punto inicial ha sido la instalación en el servidor HP, usado como punto central para preparar y gestionar el resto de los nodos de la honeynet. Los detalles se muestran en el apéndice A. Seguidamente, se prepararon las Raspberry PIs para poder usarse de forma remota; los detalles se muestran en el apéndice B.

Como aparece en la figura 4.6 cada una de las Raspberry PIs se configura para soportar una funcionalidad. En concreto, en la **pi-1** se han instalado Cowrie y Filebeat para la sincronización de archivos entre la pi-1 y el HP-Server; en la **pi-2** inicialmente se instaló Dionaea y RioTPot, en lo que fue la honeynet en su primera versión, donde se observó la escasez de recursos de la pi-2 para ejecutar este software; como consecuencia, en la honeynet definitiva no se usó la pi-2, y esta fue sustituida por el portátil Dell XPS en el que se ha configurado T-Pot; en la **pi-3** se ha instalado Wireshark para realizar el análisis de las trazas de tráfico capturadas.

El proceso de instalación de Cowrie en la pi-1 ha incluido la creación de un usuario sin privilegios de root, la instalación de dependencias en un entorno virtual y la clonación del repositorio oficial de Cowrie. Se ha configurado el entorno, se han instalado las librerías necesarias y se han realizado pruebas iniciales para asegurar el correcto funcionamiento del honeypot. Además, se ha personalizado el archivo de configuración para simular el entorno de un servidor real, ajustando parámetros como el hostname, los puertos de acceso y los usuarios señuelo, y se han redirigido los puertos estándar del sistema a los puertos usados por Cowrie para engañar a posibles atacantes. Para mejorar elrealismo, se ha replicado la estructura de archivos de un servidor real utilizando rsync, poblando los directorios con archivos falsos y configurando usuarios ficticios con información simulada. Finalmente, se ha generado un archivo especial que permite a Cowrie mostrar esta estructura al atacante, simulando tanto la organización como los permisos de un sistema real, lo que incrementa la efectividad del honeypot al captar y registrar intentos de intrusión de manera más convincente. Los detalles de la instalación y configuración de Cowrie se pueden encontrar en el apéndice C.

Tipo T-Pot	RAM	Almacenamiento	Descripción
Hive	16GB	256GB SSD	Cuantos más honeypots, sensores, y datos, más RAM y almacenamiento es necesario.

Tabla 4.1: Requisitos del sistema para instalar T-Pot

En cuanto a la instalación de T-Pot, se ha usado un ordenador con altos recursos computacionales como es el Dell XPS 13 9370, ya que por la cantidad

de servicios que utiliza, una Raspberry Pi no es capaz de soportarlo. La tabla 4.1 muestra los requisitos que tiene T-Pot para poder ejecutarse.

El proceso de instalación de T-Pot implica descargar e instalar el repositorio desde el directorio personal del usuario, estableciendo credenciales que luego permitirán acceder a la interfaz gráfica. Durante la instalación y configuración, es común encontrar conflictos de puertos debido a la cantidad de servicios involucrados, por lo que es necesario identificar los procesos que los ocupan y decidir si se detienen o se cambian de puerto para evitar problemas en el funcionamiento de los contenedores.

Una vez instalado, T-Pot ofrece una interfaz web accesible desde la red local, donde se pueden visualizar ataques en tiempo real y analizar eventos mediante Kibana. La administración de los servicios se realiza principalmente a través de Docker y archivos de configuración, permitiendo arrancar, detener, reiniciar servicios y modificar la configuración de los contenedores para adaptar el entorno a las necesidades del usuario y mejorar elrealismo de la simulación. Los detalles de la instalación y configuración del T-Pot se pueden encontrar en el apéndice D.

### 4.3. Despliegue de los dispositivos IoT reales

Antes de su incorporación a la honeynet, se ha verificado cómo es el comportamiento normal de los dos dispositivos reales seleccionados (Trådfri y Samsung Galaxy Watch Active). Para ello, se han realizado las siguientes pruebas, configuraciones y despliegue:

#### IKEA TRÅDFRI

La figura 4.9 muestra los dispositivos utilizados, que incluyen el gateway, la bombilla y el móvil utilizados:



Figura 4.9: Dispositivos Trådfri

#### 4.3. Despliegue de los dispositivos IoT reales

- Verificación previa: desde la aplicación oficial instalada en el móvil se envía el comando de encendido/apagado, y se observa los paquetes que se envían entre los dispositivos.

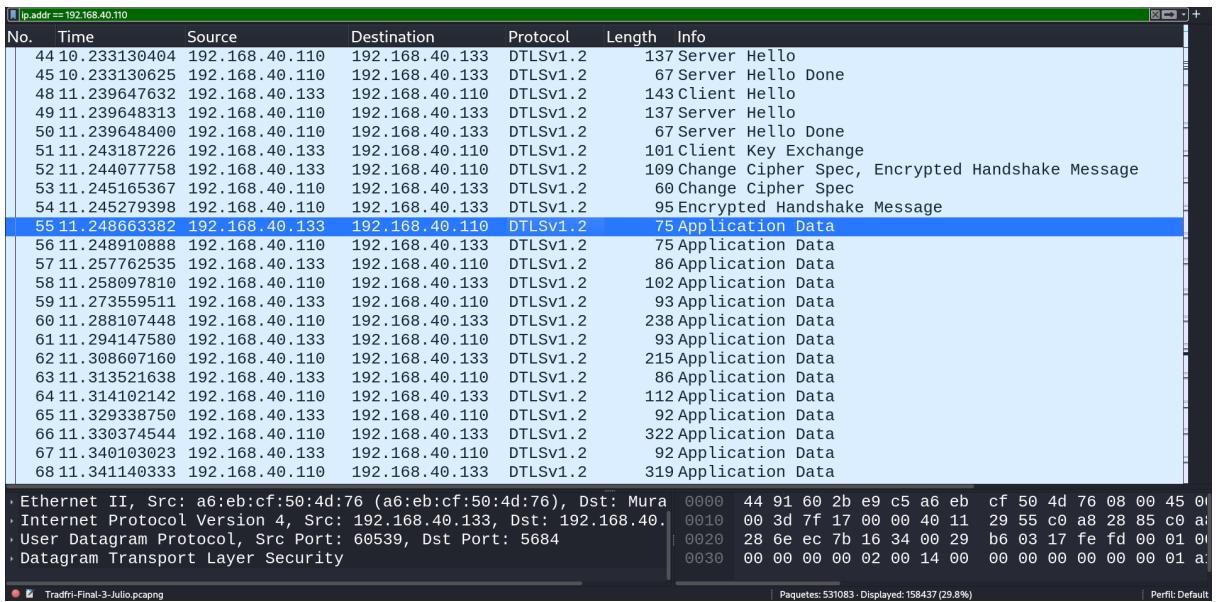


Figura 4.10: Captura Trådfri con puerto CoAPs

Además se realiza un escaneo de puertos para saber que servicios tiene escuchando:

```
(fernandoalonso@HP-Server)-[~]
└─$ sudo nmap -sU -p 0-65535 192.168.40.110
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-29 18:31 CEST
Nmap scan report for GW-4491602BE9C5.lan (192.168.40.110)
Host is up (0.00076s latency).

Not shown: 65533 closed udp ports (port-unreach)
PORT      STATE          SERVICE
68/udp    open|filtered  dhcpc
5353/udp  open|filtered  zeroconf
5684/udp  open           coaps
MAC Address: 44:91:60:2B:E9:C5 (Murata Manufacturing)

Nmap done: 1 IP address (1 host up) scanned in 5.19 seconds
```

Figura 4.11: Escaneo de puertos Trådfri

- Despliegue en honeynet: se le ha asignado la IP estática 192.168.40.110 y se ha conectado a un puerto del switch gestionable.
- Captura de todo el tráfico Wireshark mediante el port mirroring del switch filtrando por la IP 192.168.40.110.

### Samsung Galaxy Watch Active

En la figura 4.12 se muestra el reloj y el móvil utilizados:



Figura 4.12: Dispositivos Samsung Galaxy Watch Active

El reloj es un wearable IoT que se conecta con el teléfono mediante Bluetooth cuando ambos dispositivos se encuentran a corta distancia, pero además dispone de conectividad Wi-Fi y NFC. A través del Wi-Fi puede conectarse tanto a la cloud de Samsung como a otros servidores web en Internet.

- Verificación previa: se configuró el reloj para que funcione de manera autónoma para no estar conectado al móvil. Se activó el internet del reloj conectándolo a la banda Wi-Fi 2.4 GHz y se hizo la prueba de la conectividad con Amazon.com (ver figura 4.13).



Figura 4.13: Conectividad Samsung Galaxy Watch Active

- Despliegue en honeynet: se le ha asignado la IP estática 192.168.40.111 y se ha expuesto en la DMZ (exponer todos los puertos abiertos del reloj 4.14).

```
PS C:\Users\FernandoWin> nmap -sT -p 0-65535 192.168.0.4
Starting Nmap 7.97 ( https://nmap.org ) at 2025-07-03 17:46 +0200
Nmap scan report for 192.168.0.4
Host is up (0.017s latency).
Not shown: 65531 closed tcp ports (conn-refused)
PORT      STATE     SERVICE
0/tcp      filtered  unknown
7080/tcp   open      empowerid
8080/tcp   open      http-proxy
8081/tcp   open      blackice-icecap
26101/tcp  open      unknown
MAC Address: 94:83:C4:69:0A:16 (GL Technologies (Hong Kong) Limited)

Nmap done: 1 IP address (1 host up) scanned in 29.69 seconds
```

Figura 4.14: Escaneo de puertos reloj

- Captura de todo el tráfico usando tcpdump en el router de la honeynet y filtrando por la IP 192.168.40.111

## 4.4. Comprobación del funcionamiento de la red

En esta sección se describen las pruebas realizadas para la verificación tanto de la infraestructura de la red honeynet como de los requisitos de seguridad establecidos para las otras redes. Estas verificaciones, definidas en los objetivos del proyecto, son esenciales para garantizar el aislamiento entre las distintas redes y asegurar que cualquier vulnerabilidad detectada en la honeynet no pueda ser explotada para comprometer a usuarios reales.

### 4.4.1. Aislamiento de subredes domésticas

La topología de la red doméstica incluye un router principal (Sercomm) conectado a Internet y tres routers secundarios (Slate AX, TL-WR841N y Archer C50) configurados cada uno operando como un router NAT independiente con su propio servidor DHCP. Esto genera un esquema de doble NAT:

- **Router Sercomm (192.168.0.1)**: actúa como puerta de enlace principal y asigna direcciones estáticas a cada uno de los routers en la red 192.168.0.0/24.
- **Router Archer C50 (192.168.0.2)**: conectado al Sercomm en su puerto WAN, crea una nueva subred en el piso de abajo 192.168.20.0/24 y opera su propio servidor DHCP, asignando direcciones en ese rango.

- **Router TL-WR841N (192.168.0.3)**: también conectado al Sercomm, genera la subred en el piso de arriba 192.168.30.0/24 con su propio DHCP.
- **Router Slate AX (192.168.0.4)**: crea la subred 192.168.40.0/24 y gestiona direcciones en ese rango a través de DHCP todas ellas asignadas estáticamente según la MAC de los dispositivos en la honeynet.

Con este diseño:

1. Cada subred está aislada, pues los routers secundarios traducen las direcciones privadas hacia la red principal mediante NAT.
2. No existen rutas directas entre dispositivos de distintas subredes, a menos que se añadan reglas de enrutamiento específicas.
3. Cada servidor DHCP sólo ve los clientes de su subred, evitando conflictos de direcciones y facilitando el control de asignaciones.

Para verificar este aislamiento se realizaron pruebas de conectividad desde la honeynet 192.168.40.X:

- Ping a un dispositivo real de otra red, por ejemplo:

```
ping -c 4 192.168.20.10
```

Resultado: tiempo de espera agotado, lo cual confirma que no se pueden “ver” dispositivos de otra subred.

- Ping hasta puerta de enlace de otro router, por ejemplo:

```
ping -c 4 192.168.0.2
```

Resultado: si que llegaron todos los paquetes a la puerta de enlace correspondiente.

Solución: deshabilitar el ping en la configuración de los routers para evitar ataques DoS o detección de dispositivos ya que comandos como nmap usan pings.

- Traceroute para comprobar el número de saltos a un dispositivo en otra subred:

```
traceroute 192.168.30.5
```

Resultado: alcanza sólo el router Sercomm y no continúa hacia la subred destino.

Estas pruebas confirman que estas subredes (Archer C50 - 192.168.20.0/24, TL-WR841N - 192.168.30.0/24, Slate AX - 192.168.40.0/24) están correctamente segmentadas y aisladas entre sí y que se encuentran todas dentro de la red 192.168.0.0/24 del Sercomm.

#### 4.4.2. Acceso a los honeypots desde la red doméstica

Para permitir el acceso a los honeypots desde la red doméstica es necesario abrir los puertos del router Slate AX y que los servicios honeypot queden correctamente expuestos. Para ello se configuraron las reglas mostradas en la figura 4.15.

Índice	Protocolo & Descripción	Origen (zona, puerto) -- Destino (zona, IP, puerto)	Activar
1	TCP Cowrie SSH	● [WAN] 22 → [LAN] 192.168.40.102:2222	<input checked="" type="checkbox"/>
2	TCP Cowrie Telnet	● [WAN] 23 → [LAN] 192.168.40.102:2223	<input checked="" type="checkbox"/>
3	TCP T-Pot IPP	● [WAN] 631 → [LAN] 192.168.40.108:631	<input checked="" type="checkbox"/>
4	TCP T-Pot MQTT	● [WAN] 1883 → [LAN] 192.168.40.108:1883	<input checked="" type="checkbox"/>
5	TCP T-Pot secure-MQTT	● [WAN] 8883 → [LAN] 192.168.40.108:8883	<input checked="" type="checkbox"/>

Figura 4.15: Port forwarding Slate AX

Posteriormente, se realizaron las comprobaciones oportunas, primero desde dentro de la red doméstica para posteriormente exponerlo a Internet. La figura 4.16 muestra la topología usada para las pruebas desde la propia red doméstica.

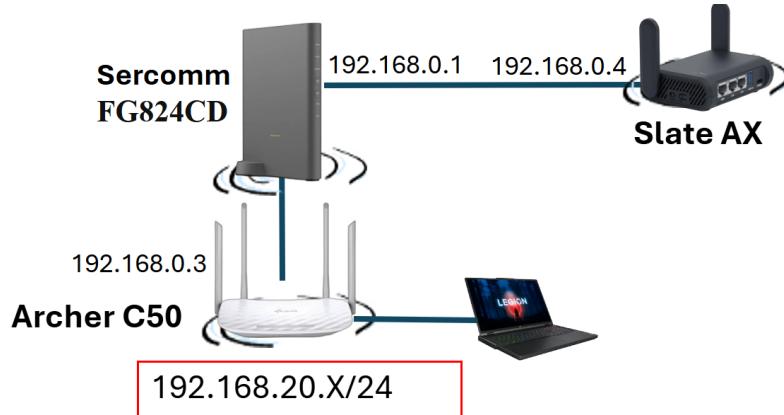


Figura 4.16: Esquema pruebas de redirección desde red doméstica

Se comprueba la conectividad y se realiza un escaneo de puertos, comprobando que los puertos están correctamente abiertos (ver figura 4.17).

```

PS C:\Users\FernandoWin> ping 192.168.0.4

Haciendo ping a 192.168.0.4 con 32 bytes de datos:
Respuesta desde 192.168.0.4: bytes=32 tiempo=5ms TTL=63
Respuesta desde 192.168.0.4: bytes=32 tiempo=5ms TTL=63
Respuesta desde 192.168.0.4: bytes=32 tiempo=5ms TTL=63
Respuesta desde 192.168.0.4: bytes=32 tiempo=6ms TTL=63

Estadísticas de ping para 192.168.0.4:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
                (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 5ms, Máximo = 6ms, Media = 5ms
PS C:\Users\FernandoWin> nmap -Pn -sT -sU -p 22,23,631,1883,8883 192.168.0.4
Starting Nmap 7.97 ( https://nmap.org ) at 2025-06-30 09:12 +0200
Nmap scan report for 192.168.0.4
Host is up (0.0051s latency).

PORT      STATE      SERVICE
22/tcp    open       ssh
23/tcp    open       telnet
631/tcp   filtered ipp
1883/tcp  filtered mqtt
8883/tcp  filtered secure-mqtt
22/udp   open|filtered ssh
23/udp   open|filtered telnet
631/udp  open|filtered ipp
1883/udp open|filtered ibm-mqisdp
8883/udp open|filtered secure-mqtt

Nmap done: 1 IP address (1 host up) scanned in 10.80 seconds

```

Figura 4.17: Prueba ping y nmap desde red doméstica

#### 4.4.3. Acceso externo a los honeypots desde Internet

Para permitir el acceso desde Internet se abren los puertos en el Sercomm (ver figura 4.18) para dejar la honeynet totalmente expuesta.

## Redirección de Puertos

La redirección de puertos permite que los equipos remotos se conecten a un dispositivo específico dentro de una LAN privada

### Redirección de puertos

Servicio	Dirección IP	Protocolo	Puerto LAN	Puerto público		
TCP	192.168.0.4	TCP	22	22		
TCP	192.168.0.4	TCP	23	23		
TCP	192.168.0.4	TCP	631	631		
TCP	192.168.0.4	TCP	1883	1883		
TCP	192.168.0.4	TCP	8883	8883		

Figura 4.18: Port forwarding Sercomm

La figura 4.19 muestra la topología usada para hacer la prueba de acceso desde Internet. Para ello, se habilita la conexión de 5G del móvil al que se conecta un ordenador.

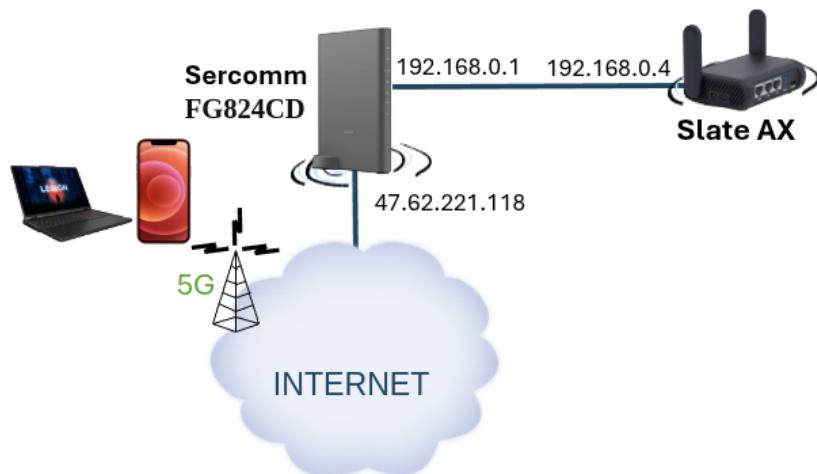


Figura 4.19: Esquema pruebas de redirección desde Internet

Para finalizar se realiza de nuevo un escaneo de puertos para comprobar que los puertos están realmente abiertos (Ver figura 4.20):

```
PS C:\Users\FernandoWin> nmap -Pn -sT -sU -p 22,23,631,1883,8883 47.62.221.118
Starting Nmap 7.97 ( https://nmap.org ) at 2025-06-29 20:28 +0200
Nmap scan report for 47-62-221-118.red-acceso.airtel.net (47.62.221.118)
Host is up (0.085s latency).

PORT      STATE     SERVICE
22/tcp    open      ssh
23/tcp    open      telnet
631/tcp   open      ipp
1883/tcp  open      mqtt
8883/tcp  open      secure-mqtt
22/udp   open|filtered ssh
23/udp   open|filtered telnet
631/udp  open|filtered ipp
1883/udp open|filtered ibm-mqisdp
8883/udp open|filtered secure-mqtt

Nmap done: 1 IP address (1 host up) scanned in 7.41 seconds
```

Figura 4.20: Prueba nmap desde Internet

# 5

## Recogida de datos y análisis

En este capítulo se describe cómo ha sido el proceso de recogida y almacenamiento de los datos de la honeynet y cómo han sido analizados. Se persigue lograr dos objetivos:

1. **Medir la exposición** de la infraestructura frente a agentes externos, utilizando observadores públicos (*Shodan*) y también ver la telemetría interna.
2. **Caracterizar la actividad maliciosa** dirigida a los servicios IoT reales y a los honeypots señuelos, a partir de los logs y las capturas de red.

Para alcanzar estos objetivos, se ha diseñado una recolección de datos mediante port mirroring de los dispositivos reales usando Wireshark y la utilización de ELK para la visualización ordenada de los voluminosos flujos de datos como son los de Cowrie y T-Pot. Los logs de los honeypots están normalizados a JSON y enriquecidos con datos de geolocalización de GeoIP, y son indexados por días. El tráfico de los dispositivos reales es capturado a fuerza bruta, pero se aplican filtros para un análisis más sencillo, ya sea por protocolos o por direcciones IP.

Se analizan las trazas de tráfico para descubrir patrones, procedencia y las credenciales más utilizadas. Se observará si hay evidencia de vulnerabilidades (CVE) que hayan sido intentadas ser potencialmente explotadas. Varios de los binarios interceptados serán sometidos a ingeniería inversa para observar su funcionamiento.

## 5.1. Shodan

Shodan es un motor de búsqueda cuya principal función es indexar dispositivos y servicios conectados como Google [23]. Hace un escaneo continuo de todas las direcciones IP públicas para catalogar los servicios que tienen los puertos abiertos. Recoge una “huella” denominada banner que incluye información como número de puerto, protocolos utilizados, versión del software y, en muchos de los casos, metadatos del dispositivo.

### Motivación y utilidad de Shodan en la honeynet

La inclusión de Shodan en la metodología parte de la necesidad de disponer de un escáner externo que confirme que los servicios de la honeynet también son detectables para agentes externos desde Internet. A diferencia de los escáneres locales, Shodan registra el banner devuelto por cada uno de los puertos que responden al menos una vez a la semana. Además, también puede detectar si se trata de un honeypot, que es lo que se intenta evitar para que los atacantes no ignoren la honeynet.

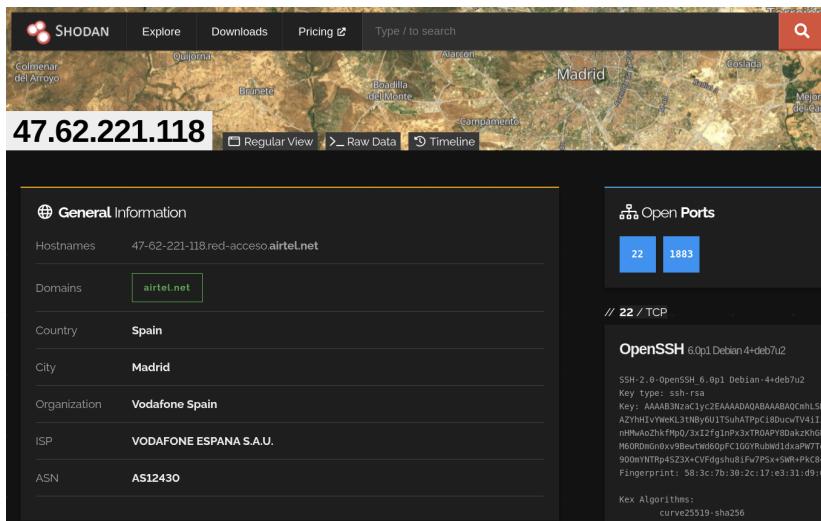


Figura 5.1: Resultado Shodan (47.62.221.118) para la honeynet

El panel de Shodan muestra la última vez que ha sido escaneado y la tabla de los puertos abiertos, donde en este caso aparecen únicamente el 22 y 1883. Esto confirma que, aunque haya más puertos abiertos (23, 631, 5684), algunos escaneos no llegan a detectar algunos de los puertos abiertos, ya sea por la capacidad computacional de los dispositivos o por la pérdida de paquetes de respuesta de los dispositivos.

## Ejemplo comparativo: otro honeypot detectado en Madrid

Para contextualizar la visión de Shodan de nuestra honeynet, la Figura 5.2 muestra otro host (5.188.181.190) que Shodan localiza en Madrid y etiqueta directamente como *honeypot* por lo que cualquier atacante podría buscarlo y ignorarlo.

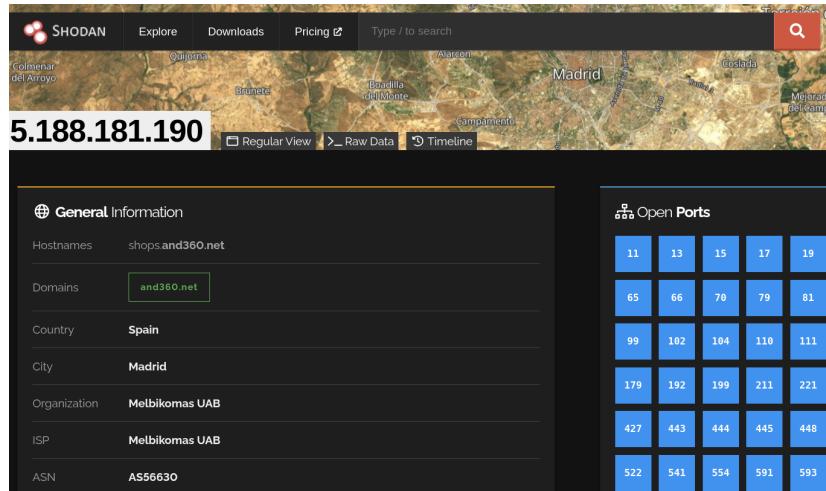


Figura 5.2: Ejemplo honeypot identificado en Madrid (5.188.181.190)

## 5.2. Recogida de datos de los honeypots

La recolección de eventos se realiza durante 10 días. Se apoya en dos honeypots **Cowrie** con *SSH* y *Telnet* y **T-Pot**, que a pesar de disponer de muchos honeypots distintos, en este trabajo se han activado *Dionaea* (MQTT), *IPPHoney* (IPP) y *Honeytrap* (MQTTs) por ser los protocolos más usados en entornos IoT. El propósito de esta sección es describir en detalle el flujo de cómo se captan los eventos, la volumetría que se ha generado y la estructura que se ha creado con estos eventos durante el período experimental.

### Arquitectura de envío y almacenamiento

Todos los honeypots escriben en sus salidas estándar (fichero de log) y esos ficheros se envían en tiempo (casi) real mediante el uso de **Filebeat**, que es un servicio muy rápido y efectivo para el envío y sincronización de archivos.

En Cowrie todo el tráfico se indexa en archivos con la estructura de logs `cowrie-logs-2025.MM.DD` que se almacenan en la carpeta por defecto. Esta carpeta ha sido configurada con un enlace simbólico hacia una unidad externa (USB)

por lo que en realidad todo se almacena en la propia unidad. La unidad se sincroniza en tiempo real y envía los archivos al HP-Server (192.168.40.101) que recibe todo con Logstash, el cual es utilizado como pipe para la visualización de datos con Elasticsearch y Kibana.

El caso de T-Pot es muy similar, ya que todos los logs generados por los contenedores Docker, Logstash los envía al Dell con la forma `logstash-2025.MM.DD`. Utilizando Elasticsearch y Kibana se pueden ver los dashboards por defecto con una gran cantidad de información.

## Volumen y granularidad de los datos

La tabla 5.1 muestra el número de documentos por cada día observado. Durante los 10 días de observación se indexan **220.200 documentos en Cowrie** con un tamaño total de *107 MiB*. En ellos, se incluyen los intentos de conexión, cada una de las contraseñas introducidas, todos los inputs, los cierres de conexión, archivos descargados, etc. Además, durante el paso de los días, las medidas de seguridad se fueron “endureciendo” ya que menos contraseñas eran aceptadas, siendo esta la principal causa del aumento de logs a lo largo del tiempo.

Tabla 5.1: Volumen diario de eventos en los índices `cowrie-logs`

Fecha	Documentos	Tamaño
2025-06-22	3.326	3,1 MiB
2025-06-23	4.193	3,3 MiB
2025-06-24	5.601	4,5 MiB
2025-06-25	5.544	4,6 MiB
2025-06-26	2.972	3,6 MiB
2025-06-27	5.825	7,8 MiB
2025-06-28	6.528	5,0 MiB
2025-06-29	86.351	34,4 MiB
2025-06-30	98.453	40,6 MiB

En paralelo, **T-Pot** ha generado **390.000 documentos (250 MiB)**. Pese a que ha recibido muchos menos ataques y conexiones que Cowrie, al tener una arquitectura tan compleja genera muchos mensajes internos de log que no tienen por qué ser de tráfico hacia el honeypot. Por esta razón, toda la información de la volumetría recibida durante la captura es menos “interesante” o descriptiva que la de Cowrie.

### Distribución de conexiones por servicio

El análisis de conexiones por servicio muestra la preferencia de los atacantes por los servicios clásicos de administración remota y, en menor medida, por servicios y protocolos propios del ecosistema IoT que generalmente son menos interactivos (ver tabla 5.2).

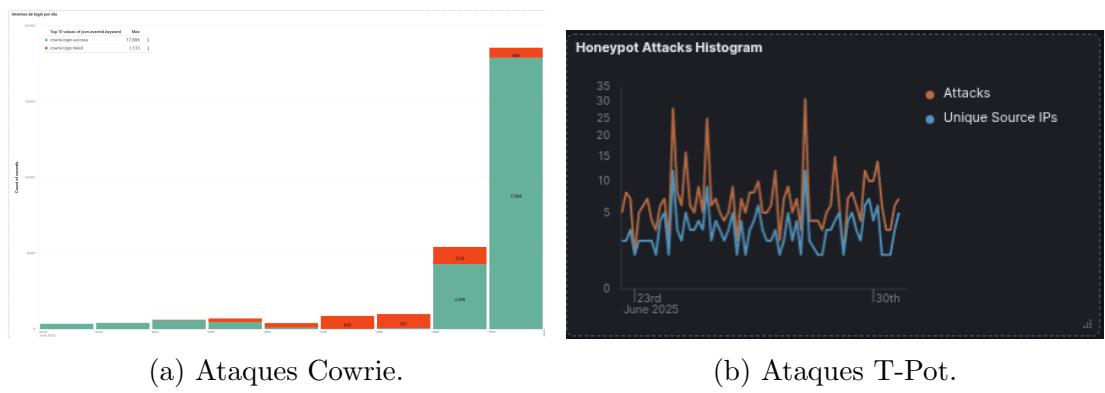
Honeypot	Servicio	Conexiones/Sesiones	IPs únicas
Cowrie	SSH	28,452 sesiones	—
Cowrie	Telnet	2,942 sesiones	—
T-Pot	Dionaea (MQTT)	77 conexiones	54
T-Pot	IPPhoney (IPP)	209 conexiones	31
T-Pot	Honeytrap (MQTTs 8883/tcp)	189 conexiones	73

Tabla 5.2: Conexiones por servicio

## 5.3. Análisis de datos de los honeypots

En el presente apartado se interpretan todos los eventos almacenados durante el periodo experimental, con el fin de identificar patrones, buscar las vulnerabilidades que se intentan explotar y evaluar la interacción que alcanzan los servicios expuestos. Las métricas seleccionadas corresponden a los paneles de Kibana que mayor poder explicativo ofrecen sobre la actividad observada.

### 5.3.1. Evolución temporal de los intentos de acceso



En la figura 5.5a se pueden observar cómo, con el tiempo, se reciben más ataques y cómo empiezan a fallar las contraseñas más frecuentemente debido al

endurecimiento de las contraseñas válidas. El último día es el más destacado debido al ataque de fuerza bruta o posible denegación de servicio que se recibió desde Estados Unidos (24.29.174.141) en el cual se recopilaron 17.780 logs con el usuario root y contraseñas al azar.

En el caso de T-Pot, ver figura 5.3b, el número de ataques diarios está más equilibrado y el día que más ataques se han recibido han sido 31 desde 12 direcciones IP únicas. Esto nos lleva a deducir que, una vez que los atacantes encuentran un servicio abierto, no se limitan a ejecutar un solo comando, sino que tratan de explotar el servicio abierto con una diversidad de pruebas.

### 5.3.2. Credenciales de fuerza bruta en Cowrie

La figura 5.4 muestra que los usuarios suelen tener persistencia en las credenciales usadas en sus intentos. En la mayoría de los casos, el atacante fija una credencial de usuario, principalmente **root**, y posteriormente ejecuta un ataque de fuerza bruta sobre la contraseña probando las combinaciones sucesivas. Este comportamiento se corrobora con el elevado predominio del usuario **root**, cuya frecuencia se puede atribuir al ataque recibido desde Estados Unidos. Por otro lado, la distribución de las contraseñas está mucho más dispersa ya que la categoría “Otros” concentra la mayoría de los valores, lo que confirma la estrategia de prueba exhaustiva que caracteriza a la fuerza bruta. En el resto de los casos se puede observar que se utilizan credenciales “típicas” como admin, user o ubuntu para los usuarios, y password, 1234 o 123 para las contraseñas.

Top 10 values of json.username.keyword	Count of records	Top 10 values of json.password.keyword	Count of records
root	23,037	admin123	1,049
admin	2,124	123456	494
support	250	support	230
test	131	password	131
user	122	admin	99
ubuntu	109	1234	87
ubnt	54	12345	62
ftpuser	45	123	60
guest	44	abc123	54
postgres	42	P@ssw0rd	49
Other	2,525	Other	25,800

Figura 5.4: Credenciales más utilizadas

### 5.3.3. Origen de los ataques

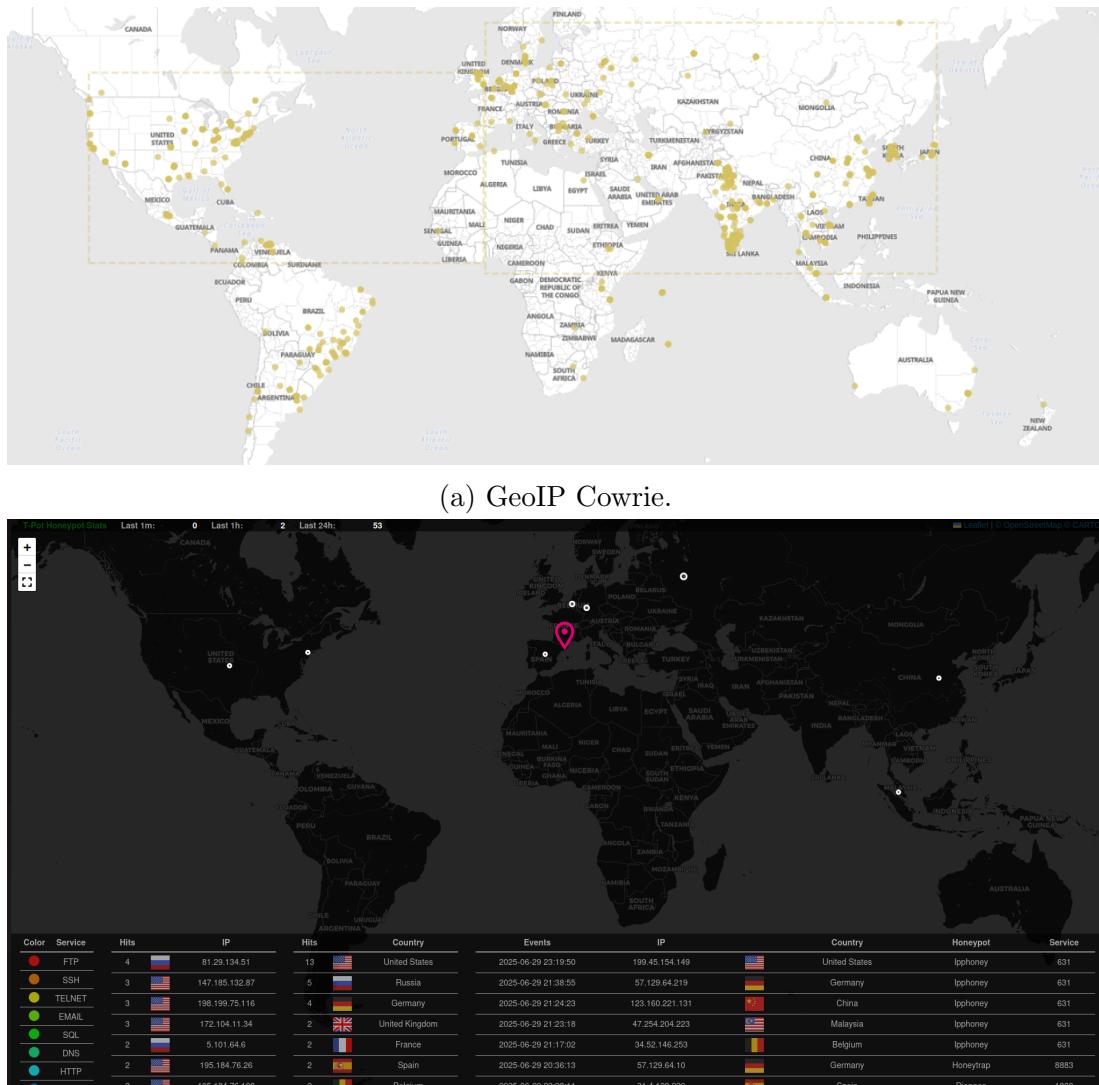


Figura 5.5: Comparativa del origen de ataques a Cowrie y T-Pot.

Tanto en Cowrie como en T-Pot, el lugar desde el que se han ejecutado más ataques, con diferencia, ha sido Estados Unidos. Cowrie ha recibido muchos ataques desde muchos lugares del mundo (independientemente del ataque de fuerza bruta de Estados Unidos) siendo otros de los más destacados Australia, Países Bajos y Vietnam. Por otro lado, T-Pot ha recibido gran parte de los ataques desde los Estados Unidos.

### 5.3.4. Comandos y archivos más comunes

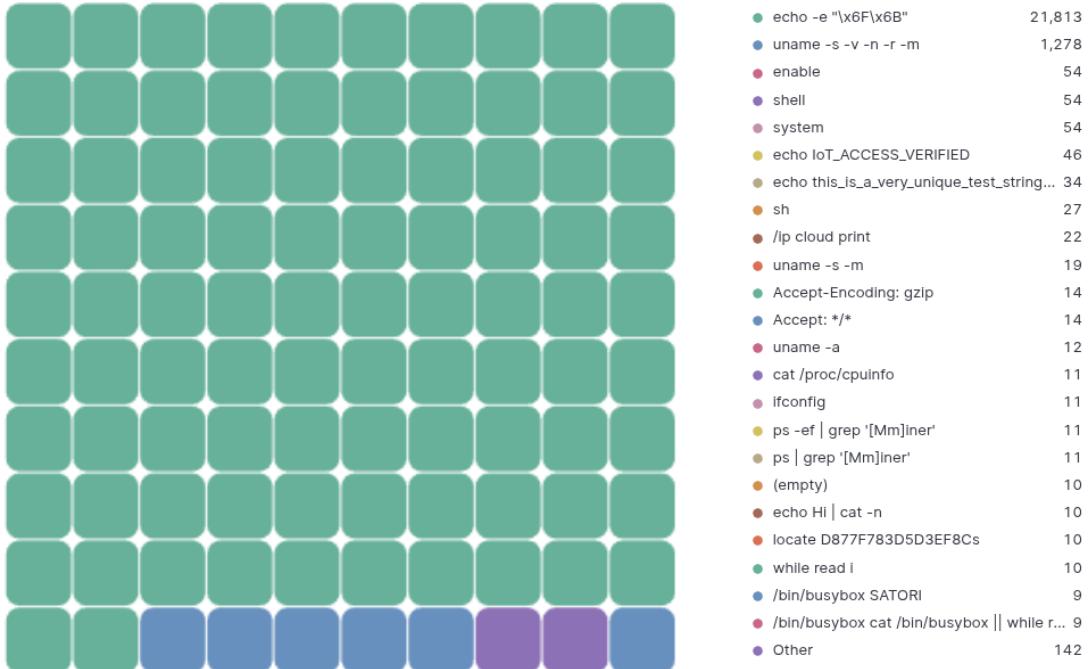


Figura 5.6: Comandos más utilizados

Una vez establecida la sesión en Cowrie, los atacantes suelen ejecutar al menos un comando y, en ocasiones, intentan descargar y lanzar algún malware. Los comandos más frecuentes son echo, cat y uname (ver figura 5.6), empleados principalmente para obtener información del sistema o leer archivos específicos.

Tabla 5.3: Archivos descargados y frecuencia de aparición

Nombre de archivo	Veces descargado
sshd	16
clean.sh	6
redtail.arm7	6
redtail.arm8	6
redtail.i686	6
redtail.x86_64	6
setup.sh	6

En cuanto a la interacción con ficheros (5.3), destacan en mayor medida los intentos de cambio de configuración de los archivos de SSH con la finalidad de facilitar accesos posteriores. Además, la descarga de binarios de la familia Redtail se intenta en numerosas ocasiones, siendo estos últimos, troyanos diseñados para

comprometer al sistema, evadir mecanismos de detección e instalar software de minería de criptomonedas [24].

### 5.3.5. Caso práctico - Análisis malware

Para evaluar el impacto real de los binarios descargados a través del honeypot Cowrie, se ha utilizado la muestra `306c4e97...45ed` [25] en el entorno controlado ANY-RUN (Ubuntu 22.04), cuyo panel de análisis se muestra en la Figura 5.7. Los antivirus lo clasifican como *Linux.CoinMiner* y está dirigido a arquitecturas x86-64 y ARM típicas de dispositivos IoT.

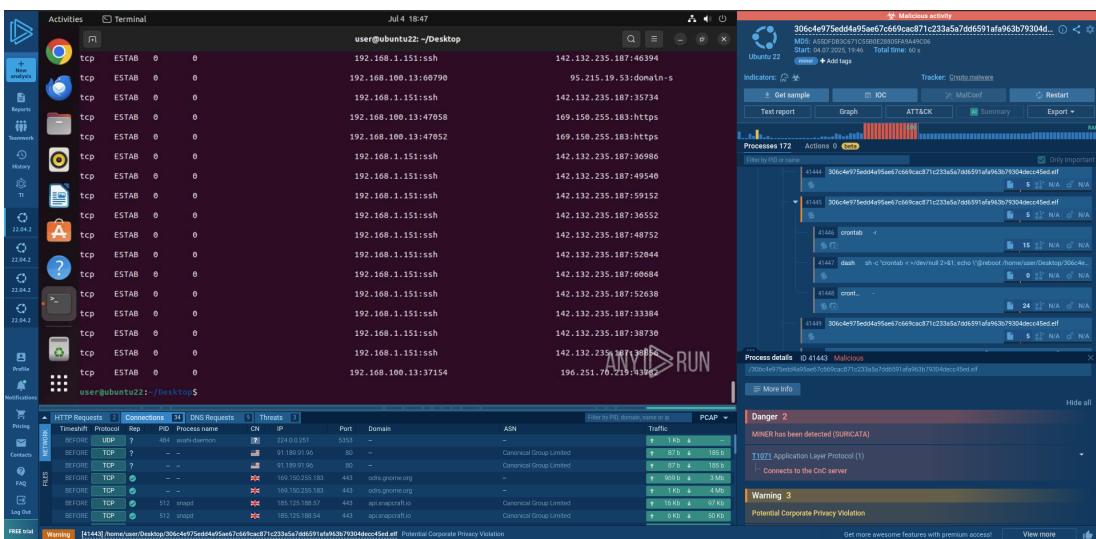


Figura 5.7: Ejecución en ANY-RUN: procesos, conexiones y alertas.

A continuación se resume el comportamiento observado del archivo tras su ejecución en la sandbox.

## 1. Inicialización

- Tras iniciarse se le conceden permisos de ejecución +x al archivo y se ejecuta el .elf.
- El binario se desempaquetá en memoria y empieza a generar copias en distintos lados para asegurarse que siempre haya una de esas copias funcionando.
- Añade una tarea automática en el planificador del sistema *cron* de modo que, cada vez que el ordenador es reiniciado, se vuelve a ejecutar de nuevo el sistema minero.

- Para dificultar el ser descubierto, además establece la tarea como “solo lectura” para evitar que un administrador del sistema lo borre por error.

## 2. Comportamiento en red

- Una vez iniciado, el programa se conecta a servidores externos utilizando los puertos 3333 y 5555, siendo estos los más habituales para las granjas de la criptomoneda Monero.
- No envía información personal de ningún tipo ni tampoco descarga más ficheros, su único objetivo es pasar desapercibido y aprovechar la potencia de un dispositivo para minar.

## 3. Impacto sobre el sistema

- Mantiene el procesador trabajando casi al máximo, por lo que otros servicios podrían volverse más lentos.
- Consume poca memoria, de esta manera podría pasar desapercibido para aquellos que solo vigilan el uso de RAM del sistema.
- Puede ser complicado de hacer desaparecer, ya que si no se elimina la tarea que lo automatiza y las copias de los archivos, el malware reaparece tras cada reinicio.

## 4. Principales riesgos

- Uso abusivo de CPU: el minero mantiene los núcleos cerca del 90 % de carga de forma continuada. Esto supone un bajo rendimiento en los servicios legítimos del dispositivo y un aumento en la temperatura y consumo interno del equipo.
- Persistencia encubierta: la tarea cron marcada como inmutable (`chattr +i`) dificulta su eliminación.
- Reputación de la dirección IP: el tráfico puede provocar que la IP se incluya en listas negras, bloqueando correos u otros servicios salientes asociados al mismo host.

## Conclusión

En resumen, la muestra no te roba ni destruye datos del sistema, pero convierte tu dispositivo en un nodo de un *sistema de minería clandestina*. El mayor

perjuicio que provoca son las consecuencias económicas (coste eléctrico) y operativas (pérdida de rendimiento del sistema y inclusión en listas de abuso). La erradicación resulta sencilla si se detecta a tiempo y se logra eliminar la tarea programada (cron), se borran las copias del binario y se refuerza la configuración del sistema. No obstante, en esta sección se subraya la importancia de la utilización de credenciales robustas y monitorización de los recursos para así evitar compromisos que puedan derivar a problemas mayores.

La figura 5.8 muestra los diferentes procesos ejecutados por el malware en una línea temporal, donde se puede observar su complejidad.

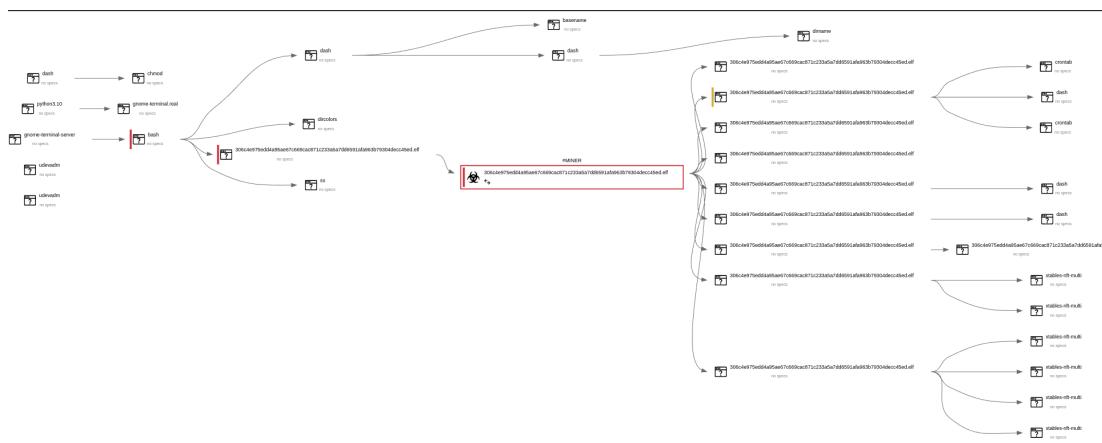


Figura 5.8: Línea temporal procesos ejecutados por el malware.

## 5.4. Recogida de datos de los dispositivos IoT reales

La honeynet contaba con dos dispositivos reales: la pasarela TRÅDFRI de Ikea junto a la correspondiente bombilla, y el reloj Samsung Galaxy Watch Active.

En el caso de la pasarela TRÅDFRI se han estado recogiendo datos 30 horas y el fichero con las trazas ocupa 50 MBytes. El número de paquetes recogidos en la traza es de 531.083 paquetes.

El reloj Samsung estuvo expuesto 48 horas mientras se recogía el tráfico y el fichero con las trazas ocupa 73,3 MBytes. El número de paquetes almacenados en la traza es de 857.000 paquetes.

## 5.5. Análisis de datos de los dispositivos IoT reales

Para el análisis de las trazas capturadas se ha usado wireshark versión 4.4.5.

### 5.5.1. Dispositivos TRÅDFRI

Consultando la aplicación Ikea Home Smart 1 en el móvil, se ha averiguado que la versión del firmware de la pasarela TRÅDFRI es la 1.21.44. En primer lugar, se ha consultado en INCIBE vulnerabilidades de estos dispositivos TRÅDFRI encontrándose la (CVE-2022-39065) consistente en el envío de un comando defec-tuoso que hace que la puerta de enlace deje de funcionar adecuadamente y deje de poder controlar las luces conectadas. Dicha vulnerabilidad era para versiones de firmware anteriores a la versión 1.19.26 [26], por lo que esta gateway no es sensible a dicha vulnerabilidad.

La pasarela TRÅDFRI en su funcionamiento habitual usa el protocolo CoAPs (CoAP usando DTLS) que es el protocolo de aplicación usado entre el móvil y la pasarela para el envío de las órdenes de encendido y apagado de la bombilla. Adicionalmente, la pasarela TRÅDFRI se conecta periódicamente a un servidor AWS IoT para permitir la gestión de la bombilla (encendido/apagado de forma remota) y su integración con asistentes de voz. Esta comunicación utiliza como protocolo el HTTPS (HTTP usando TLS).

#### Trazas CoAPs

En el puerto 5684/UDP reservado para CoAP seguro (sobre DTLS) se observa siempre el mismo patrón (ver figura 5.9), todos los clientes externos envían el paquete sin la clave necesaria para poder conectarse al TRÅDFRI. Los paquetes son para exploración, no van dirigidos específicamente y el servicio les responde como visible, pero no es vulnerable, ya que no intercambia nada sin la clave.

Uno de los paquetes contiene un SNI (Server Name Indicator) que es un indicador del nombre de servidor apuntando a 127.0.0.1 (que es la dirección de loopback o localhost). La cabecera SNI, no es una cabecera HTTP, sino una extensión del protocolo TLS que permite que un cliente (como un navegador web) indique el nombre del dominio al que quiere conectarse durante el primer paso del establecimiento de una conexión segura (el “handshake” TLS). Los servidores devuelven respuestas distintas ante diferentes SNI y en este caso, al no tratarse de una dirección específica, debería devolver un error que daría información al atacante sobre el sistema.

## 5.5. Análisis de datos de los dispositivos IoT reales

Ese paquete con el SNI y otro tienen IPs que pertenecen a la misma ASN 398324 (Autonomous System Number), lo que revela que ambas pertenecen a Censys, pero utilizan patrones distintos. Censys es otro escáner que ofrece información de los servicios en la red, al igual que Shodan. El hecho de que varios hosts del mismo proveedor usen patrones que se complementan entre ellos muestra una estrategia organizada para escanear y enumerar servicios IoT.

Tir	Source	Destination	Protocol	Ler Info
6...	192.168.0.2	192.168.40.110	DTLSv1.2	1... Client Hello
6...	192.168.40.110	192.168.0.2	DTLSv1.2	60 Alert (Level: Fatal, Description: Handshake Failure)
6...	47.62.221.118	192.168.40.110	TFTP	1... Unknown (0x1fe)
6...	192.168.40.110	47.62.221.118	TFTP	60 Unknown (0x1fe)
6...	47.62.221.118	192.168.40.110	DTLSv1.2	1... Client Hello
6...	192.168.40.110	47.62.221.118	DTLSv1.2	60 Alert (Level: Fatal, Description: Handshake Failure)
-	167.94.138.138	192.168.40.110	DTLSv1.2	1... Client Hello (SNI=127.0.0.1)
	192.168.40.110	167.94.138.138	DTLSv1.2	60 Alert (Level: Fatal, Description: Handshake Failure)
-	167.94.138.138	192.168.40.110	ICMP	85 Destination unreachable (Port unreachable)
1...	162.142.125.114	192.168.40.110	DTLSv1.2	1... Client Hello
1...	192.168.40.110	162.142.125.114	DTLSv1.2	60 Alert (Level: Fatal, Description: Handshake Failure)
1...	162.142.125.114	192.168.40.110	DTLSv1.2	1... Client Hello
1...	192.168.40.110	162.142.125.114	DTLSv1.2	60 Alert (Level: Fatal, Description: Handshake Failure)
1...	185.73.23.133	192.168.40.110	DTLS	63 Continuation Data
↓ Frame 492567: 181 bytes on wire (1448 bits), 181 bytes captured (1448 bits) on interface eth0, id 0				
Ethernet II, Src: GLTechnologi_69:0a:17 (94:83:c4:69:0a:17), Dst: MurataManufa_2b:e9:c5 (44:91:60:2b:e9:c5)				
Internet Protocol Version 4, Src: 167.94.138.138, Dst: 192.168.40.110				
User Datagram Protocol, Src Port: 23542, Dst Port: 5684				
↓ Datagram Transport Layer Security				
↓ DTLSv1.2 Record Layer: Handshake Protocol: Client Hello				

Figura 5.9: Ataques CoAPs TRÅDFRI.

## Trazas mDNS

La Figura 5.10 muestra un extracto de las capturas que han sido obtenidas sobre mDNS en el puerto 5353/UDP. Es un protocolo que los dispositivos domésticos utilizan para anunciararse al resto sin necesidad de un DNS central. Las maneras en las que los escáneres intentan extraer información utilizando los servicios mDNS son muy variadas.

Source	Destination	Protocol	Leng Info
162.142.125.81	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
146.88.241.35	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
104.195.12.37	192.168.40.110	MDNS	71 Standard query 0x492e[Malformed Packet]
64.62.197.187	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
198.235.24.68	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
104.195.12.37	192.168.40.110	MDNS	78 Unknown operation (9) 0xb2e[Malformed Packet]
111.176.22.8	192.168.40.110	MDNS	182 Inverse query 0x1293 Unknown <Unknown extended label>, "QM" question[Malformed Packet]
185.242.226.31	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
45.155.99.140	192.168.40.110	MDNS	112 Unknown operation (7) 0x492e[Malformed Packet]
14.225.211.134	192.168.40.110	MDNS	89 Unknown operation (7) 0x382e[Malformed Packet]
162.19.193.158	192.168.40.110	MDNS	129 Unknown operation (3) 0x162e[Malformed Packet]
123.58.209.112	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
123.58.209.112	192.168.40.110	MDNS	90 Standard query 0x0300 Unused <Root>, "QM" question Unused
94.162.49.193	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
45.156.128.86	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
204.76.293.41	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
8.211.45.55	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
64.62.156.51	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
187.189.119.83	192.168.40.110	MDNS	88 Unknown operation (10) 0x662e[Malformed Packet]
162.19.192.3	192.168.40.110	MDNS	87 Unknown operation (7) 0x4b2e Unknown (26987) <Unknown extended label>, "QM" question[Malformed Packet]
70.153.193.28	192.168.40.110	MDNS	78 Unknown operation (3) 0x342e[Malformed Packet]
192.168.40.110	70.153.193.28	ICMP	70 Destination unreachable (Port unreachable)
198.235.24.195	192.168.40.110	MDNS	88 Standard query 0x0000 PTR _services._dns-sd._udp.local, "QM" question
↓ Frame 384654: 90 bytes on wire (720 bits), 90 bytes captured (720 bits) on interface eth0, id 0			
Ethernet II, Src: GLTechnologi_69:0a:17 (94:83:c4:69:0a:17), Dst: MurataManufa_2b:e9:c5 (44:91:60:2b:e9:c5)			
Internet Protocol Version 4, Src: 167.94.138.138, Dst: 192.168.40.110			
User Datagram Protocol, Src Port: 23542, Dst Port: 5353			
↓ Multicast Domain Name System (query)			

Figura 5.10: Ataques mDNS TRÅDFRI.

Los paquetes piden el registro `_services._dns-sd._udp.local` [27] por lo que solicitan que el dispositivo enumere todos los servicios mDNS que hay en la red. Es una forma de que un atacante mida si el host sirve como un amplificador para ayudar a realizar un ataque de DDoS.

Wireshark además detecta paquetes que aparecen como *Malformed Packet*. Los escáneres de red mandan a propósito este tipo de paquetes para ver cómo responden los dispositivos IoT, ya que les puede dar información útil para poder clasificar el dispositivo e incluso buscar vulnerabilidades debido a firmwares antiguos.

A pesar de ser el puerto 5353 perteneciente al multicast (dirección 224.0.0.251), los paquetes capturados son unicast y van dirigidos directamente al TRÅDFRI, lo que prueba que son sondas de escaneo externo deliberado.

La IP de Hong Kong (123.58.209.112) aparece dos veces: primeramente pidiendo la enumeración de los dispositivos y luego otra con una query con Root, demostrando estar “probando” el funcionamiento del dispositivo y la búsqueda de una posible mala configuración.

## Conexiones con AWS IoT

El gateway TRÅDFRI inicia periódicamente varias sesiones TLS (puerto 443) hacia distintas IP pertenecientes a Amazon. Todas muestran la misma cabecera SNI que es un indicador del nombre de servidor, es decir, dentro de Amazon el gateway se quiere conectar específicamente a AWS IoT Core. IKEA utiliza este servidor para que sus dispositivos:

- Publiquen/reciban mensajes MQTT (cambios de estado, telemetria).
- Descarguen actualizaciones de firmware.
- Manden mensajes de keep-alive cada cierto tiempo para comprobar el funcionamiento correcto del dispositivo.

### 5.5.2. Reloj Samsung Galaxy

La versión del firmware del reloj es Tizen 5.5.0.2, una plataforma desarrollada por Samsung basada en Linux. A lo largo del análisis se ha observado que el reloj expone varios servicios TCP (están abiertos los puertos 7080, 8080, 8081 y 26101) que podrían formar parte de mecanismos de depuración o de sincronización con la aplicación móvil asociada (Galaxy Wearable).

Este comportamiento no es exclusivo del Galaxy Watch sino de otros dispositivos con Tizen como ciertos modelos de televisores inteligentes, que muestran un conjunto similar de puertos abiertos, tal y como se muestra en la figura 5.11. En ella, se muestran servicios como WebSocket o HTTP, muchos de ellos asociados a la comunicación y sincronización entre dispositivos Samsung.

```
Not shown: 65523 closed ports
Reason: 65523 resets
PORT      STATE SERVICE      REASON      VERSION
7678/tcp   open  upnp        syn-ack ttl 64  Samsung Allshare upnpd 1.0 (UPnP 1.1)
8001/tcp   open  vcom-tunnel?  syn-ack ttl 64
8002/tcp   open  ssl/teradataordbms?  syn-ack ttl 64
8080/tcp   open  http        syn-ack ttl 64  lighttpd
8187/tcp   open  upnp        syn-ack ttl 64  Samsung Allshare upnpd 1.0 (UPnP 1.1)
9012/tcp   open  websocket   syn-ack ttl 64  WebSocket++ 0.5.1
9119/tcp   open  upnp        syn-ack ttl 64  Samsung Allshare upnpd 1.0 (UPnP 1.1)
9197/tcp   open  upnp        syn-ack ttl 64  Samsung Allshare upnpd 1.0 (UPnP 1.1)
26101/tcp  open  tcpwrapped  syn-ack ttl 64
```

Figura 5.11: Puertos abiertos televisión Samsung (Fuente: [28]).

### Análisis de las trazas 7080

El puerto TCP 7080 del Galaxy Watch responde a sondas externas con un bloque fijo de 472 bytes, cuyo encabezado coincide con la respuesta **CONNECT\_OK** del protocolo **SDB (Smart Development Bridge)** utilizado por dispositivos Tizen para la depuración remota [28]. Esta respuesta confirma la presencia del servicio, pero no filtra información adicional ni acepta comandos posteriores, lo que limita su riesgo inmediato. No obstante, dejar este puerto abierto expone el dispositivo a escáneres globales como Censys o Expanse, que pueden indexarlo como un servicio de depuración activo [29].

### Análisis de las trazas 8080 y 8081

Los puertos 8080 y 8081 funcionan como servidor web interno para conectarse al reloj a través del móvil cuando el Bluetooth no está disponible. El puerto 8080 probablemente está a la escucha de peticiones HTTP de la aplicación móvil u otros servicios Samsung. Por otro lado, el puerto 8081 funcionaría de manera

188.212.103.171	192.168.40.111	HTTP	230 GET /locales/locale.json?locale=.../config/&namespace=database	HTTP/1.1
87.236.176.241	192.168.40.111	HTTP	270 GET /	HTTP/1.1
185.247.137.220	192.168.40.111	HTTP	90 GET /	HTTP/1.0
185.247.137.220	192.168.40.111	HTTP	94 OPTIONS /	HTTP/1.0
185.247.137.220	192.168.40.111	HTTP	125 GET /nice%20ports%2C/Tri%6Eity.txt%2ebak	HTTP/1.0
87.236.176.244	192.168.40.111	HTTP	335 OPTIONS sip:47.62.221.118 SIP/2.0	
185.247.137.244	192.168.40.111	HTTP	103 OPTIONS *	RTSP/1.0
196.251.83.136	192.168.40.111	HTTP	99 CONNECT 196.251.83.136:80	HTTP/1.0
193.34.212.116	192.168.40.111	HTTP	230 GET /locales/locale.json?locale=.../config/&namespace=database	HTTP/1.1
134.209.98.0	192.168.40.111	HTTP	230 GET /Locales/locale.json?locale=.../config/&namespace=database	HTTP/1.1
147.185.132.240	192.168.40.111	HTTP	422 GET /	HTTP/1.1
142.93.39.6	192.168.40.111	HTTP	678 GET /	HTTP/1.1
185.218.84.178	192.168.40.111	HTTP	501 GET /	HTTP/1.1
204.76.203.206	192.168.40.111	HTTP	116 GET /	HTTP/1.1
196.251.117.173	192.168.40.111	HTTP	180 CONNECT 45.61.137.126:7227	HTTP/1.1
196.251.117.173	192.168.40.111	HTTP	180 CONNECT 45.61.137.126:7227	HTTP/1.1
196.251.117.173	192.168.40.111	HTTP	180 CONNECT 45.61.137.126:7227	HTTP/1.1

```

> Frame 4103: 116 bytes on wire (928 bits), 116 bytes captured (928 bits)
> Linux cooked capture v2
> Internet Protocol Version 4, Src: 45.135.193.162, Dst: 192.168.40.111
> Transmission Control Protocol, Src Port: 55644, Dst Port: 8080, Seq: 1, Ack: 1, Len: 44
> Hypertext Transfer Protocol
```

Figura 5.12: Ataques a los puertos 8080 y 8081.

La captura de las conexiones entrantes al puerto 8080 muestra que el reloj confirma conexiones entrantes desde Internet y las identifica como peticiones HTTP (ver figura 5.12). Por ejemplo, múltiples clientes realizaron peticiones *GET* / *HTTP/1.1* lo que sugiere que efectivamente ejecuta algún servicio web o API HTTP en ese puerto. No obstante, al no tratarse de un servicio especializado, no responde con páginas web habituales ni con respuestas comunes, sino que parece que está a la espera de consultas más específicas por parte de un cliente autorizado.

En cuanto al puerto 8081, los clientes registraron intentos de handshake usando TLS. En concreto, se pueden ver paquetes *Client Hello* de TLS, lo que indica que los escáneres de la red tratan de negociar una conexión. El reloj recibe esos handshakes pero no completa la negociación, ya que no devuelve ningún *Server Hello* lo que sugiere que el dispositivo cierra la conexión tras el saludo inicial. Posiblemente el puerto 8081 esté destinado a comunicaciones cifradas, pero con comunicación previa, ignorando así handshakes de orígenes no reconocidos.

## Puerto 26101

A pesar de no haber capturado ningún paquete asociado a dicho puerto, es interesante resaltar que se trata del “Samsung debug bridge” puerto propietario del propio Samsung para hacer pruebas y despliegues para el reloj. En el artículo [28] se menciona que el uso de comandos que realizan ejecuciones en el sistema, mencionando que un mal uso o comprobación de estos comandos podría llevar a una inyección de comandos por parte de un atacante en este puerto.

# 6

## Conclusiones y trabajos futuros

El trabajo realizado ha demostrado que los servicios IoT expuestos a Internet son un objetivo constante de escaneos y de ataques automatizados, especialmente aquellos servicios más tradicionales de acceso remoto como son SSH y Telnet.

La honeynet desplegada, compuesta por honeypots y varios dispositivos IoT reales, ha permitido observar con detalle tanto la actividad maliciosa como el funcionamiento habitual de estos sistemas. Además, se han confirmado estrategias sistemáticas de fuerza bruta sobre credenciales comunes y el despliegue de malware como *RedTail*, orientado a minería no autorizada de criptomonedas. Con el trabajo se ha visto que la utilización de honeypots combinados con herramientas de análisis como ELK y Wireshark ha demostrado ser muy efectivas para entender las tácticas empleadas por los atacantes.

Por otro lado, aunque los dispositivos reales no fueron comprometidos, sí fueron objeto de exploración por atacantes y escáneres automatizados como Shodan y Censys. La gateway TRÅDFRI a pesar de recibir paquetes de protocolos como CoAPs y mDNS, no expuso datos sensibles ni permitió conexiones sin autenticación previa. El Galaxy Watch Active, por su parte, mantuvo servicios por defecto abiertos que responden de manera muy limitada (típico de dispositivos headless con poca capacidad), pero sin aceptar comunicaciones no autorizadas. Estos resultados muestran que un buen diseño y una configuración segura pueden resistir a ataques pasivos de la red, aunque la exposición sigue siendo un riesgo.

Una de las lecciones más importantes extraídas de este trabajo es que los atacantes actúan sobre lo que es visible. La simple exposición de un servicio puede ser suficiente para atraer escaneos o intentos de acceso, incluso aunque no

haya vulnerabilidades explotables. Este comportamiento reafirma la importancia de limitar la superficie de ataque mediante firewalls, listas de control de acceso y desactivación de servicios innecesarios.

El trabajo confirma que los entornos IoT requieren de una atención continua en cuanto a seguridad: limitar el número de puertos abiertos, cambiar las credenciales a unas seguras, mantener los dispositivos actualizados y monitorizar la actividad son prácticas fundamentales.

Cabe destacar que, en el contexto doméstico habitual, la mayoría de dispositivos IoT no representan un riesgo elevado para los usuarios medios. Esto se debe a que los dispositivos suelen operar en redes Wi-Fi privadas protegidas por autenticación, y establecen conexiones cifradas con servidores de confianza. Además, muchos de estos dispositivos están diseñados con sistemas cerrados (firmware propietario, interfaces mínimas) y utilizan mecanismos como certificados, claves precompartidas o tokens para asegurar la autenticación mutua. Por lo tanto, si el entorno está correctamente configurado, resulta improbable que sufren un compromiso real sin que haya una amenaza interna o una mala configuración previa.

Como trabajo futuro, se propone ampliar la duración del experimento así como mejorar la interacción de los dispositivos para dar lugar a que los atacantes interactúen e intenten utilizar más métodos. También sería relevante profundizar más en la ingeniería inversa como la realizada con el malware de RedTail para entender y explorar técnicas de mitigación activas que puedan responder de una forma más eficaz a amenazas en evolución.

# Bibliografía

- [1] L. S. Vailshery. (2024) Number of internet of things (iot) connections worldwide from 2022 to 2023, with forecasts from 2024 to 2033. [Online]. Available: <https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/>
- [2] A. Z. Tabari, X. Ou, and A. Singhal, “What are attackers after on iot devices? an approach based on a multi-phased multi-faceted iot honeypot ecosystem and data clustering,” 2021. [Online]. Available: <https://arxiv.org/abs/2112.10974>
- [3] INCIBE. (2025) Botnet. [Online]. Available: <https://www.incibe.es/aprendeciberseguridad/botnet#:~:text=El%20Concepto,conjunta%20para%20realizar%20actividades%20maliciosas>
- [4] INCIBE. (2025) CVE-2017-9765. [Online]. Available: <https://www.incibe.es/incibe-cert/alerta-temprana/vulnerabilidades/cve-2017-9765>
- [5] Cloudflare Learning Center. (2023) Mirai botnet: A brief overview. [Online]. Available: <https://www.cloudflare.com/learning/ddos/glossary/mirai-botnet/>
- [6] Wikipedia contributors. (2025) Vpnfilter. [Online]. Available: <https://es.wikipedia.org/wiki/VPNFilter>
- [7] Akamai Security Research. (2021) Corona-mirai botnet infects zero-day sirt routers. [Online]. Available: <https://www.akamai.com/es/blog/security-research/corona-mirai-botnet-infects-zero-day-sirt>
- [8] Kaspersky Resource Center. (2024) ¿qué es un honeypot? [Online]. Available: <https://latam.kaspersky.com/resource-center/threats/what-is-a-honeypot>
- [9] Fortinet. (2024) What is a honeypot? [Online]. Available: <https://www.fortinet.com/lat/resources/cyberglossary/what-is-honeypot>
- [10] M. Oosterhof. (2025) Cowrie ssh/telnet honeypot. [Online]. Available: <https://github.com/cowrie/cowrie>
- [11] Deutsche Telekom Security GmbH. (2025) T-pot community edition (multi-honeypot platform). [Online]. Available: <https://github.com/telekom-security/tpotce>
- [12] AAU Network Security. (2025) Riotpot: A honeypot for riot os-based iot devices. [Online]. Available: <https://github.com/aau-network-security/riotpot>
- [13] Conpot Project. (2025) Conpot: Ics/scada honeypot. [Online]. Available: <http://conpot.org/>
- [14] Dionaea Project. (2025) Dionaea honeypot documentation. [Online]. Available: <https://dionaea.readthedocs.io/en/latest/>
- [15] G. yon. (2024) Nmap reference guide. [Online]. Available: <https://nmap.org/book/man.html>

## BIBLIOGRAFÍA

---

- [16] GL.iNet. (2023) Slate ax (gl-axt1800) datasheet. [Online]. Available: [https://static.gl-inet.com/www/images/products/datasheet/axt1800\\_datasheet\\_20230602.pdf](https://static.gl-inet.com/www/images/products/datasheet/axt1800_datasheet_20230602.pdf)
- [17] OpenWrt Documentation. (2025) Dnsmasq: Dhcp and dns server. [Online]. Available: <https://openwrt.org/docs/guide-user/base-system/dhcp.dnsmasq>
- [18] ABC Xperts. (2024) Port mirroring: clave para la visibilidad y seguridad en redes. [Online]. Available: <https://abcxperts.com/port-mirroring-clave-para-la-visibilidad-y-seguridad-en-redes/>
- [19] Wallarm Labs. (2024) ¿qué es el protocolo coap? significado y arquitectura. [Online]. Available: <https://lab.wallarm.com/what/que-es-el-protocolo-coap-significado-y-arquitectura/?lang=es>
- [20] Ringover. (2023) Dtls: Qué es y cómo funciona. [Online]. Available: <https://www.ringover.es/blog/dtls>
- [21] Amazon Web Services. (2024) What is mqtt? [Online]. Available: <https://aws.amazon.com/what-is/mqtt/>
- [22] IONOS Digital Guide. (2024) Multicast dns (mdns): qué es y cómo funciona. [Online]. Available: <https://www.ionos.es/digitalguide/servidores/know-how/multicast-dns/>
- [23] S. H. Center. (2025) On-demand scanning. Consulta: 5 julio 2025. [Online]. Available: <https://help.shodan.io/the-basics/on-demand-scanning>
- [24] J. Sandbox. (2025) Análisis dinámico #1368061. Consulta: 5 julio 2025. [Online]. Available: <https://www.joesandbox.com/analysis/1368061/0/html>
- [25] (2025) Virustotal report for sha-256 306c4e975edd4a95ae67c669cac871c233a5a7dd6591afa963b79304decc45ed. Consulta: 5 julio 2025. [Online]. Available: <https://www.virustotal.com/gui/file/306c4e975edd4a95ae67c669cac871c233a5a7dd6591afa963b79304decc45ed>
- [26] INCIBE-CERT. (2025) Cve-2022-39065. Consulta: 5 julio 2025. [Online]. Available: <https://www.incibe.es/incibe-cert/alerta-temprana/vulnerabilidades/cve-2022-39065>
- [27] stammen, “dnssd-uwp issue #8: mdns queries being sent to lan without any response,” <https://github.com/stammen/dnssd-uwp/issues/8>, 2025, consulta: 7 julio 2025.
- [28] W. Labs. (2025) Opening up the samsung q60 series smart tv. Consulta: 7 julio 2025. [Online]. Available: <https://labs.withsecure.com/publications/samsung-q60r-smart-tv-opening-up-the-samsung-q60-series-smart-tv>
- [29] DataDome. (2025) Expanse (palo alto networks). Consulta: 7 julio 2025. [Online]. Available: <https://datadome.co/bots/expanse/>
- [30] G. Saurel and A. Fougères, “Ocultación avanzada en honeypots basados en cowrie,” Inria, Tech. Rep., 2022. [Online]. Available: <https://inria.hal.science/hal-03746028/document>
- [31] D. T. S. GmbH. (2025) Issue #1084: Ipp port conflict with cups prevents honeypot from binding to port 631. [Online]. Available: <https://github.com/telekom-security/tpotce/issues/1084>



# Apéndices



# A

## Configuración Servidor HP

El HP-Server actúa como servidor central para preparar y coordinar los nodos de la honeynet, en dicho servidor se ha instalado Kali Linux. A continuación, se detallan los pasos seguidos:

### 1. Instalación de Kali Linux

Se instaló *Kali Linux* ya que dispone de muchísima personalización y ayuda a la administración de las redes. En concreto decidió Kali Linux al ser la distribución más famosa para pruebas de penetración.

### 2. Actualización del sistema

Una vez arrancado, se ejecutaron los siguientes comandos para garantizar que todos los paquetes se mantengan al día:

```
sudo apt update  
sudo apt full-upgrade -y
```

### 3. Generación de claves SSH

Para acceso seguro sin la necesidad de introducir contraseña en las Raspberry Pis, se creó un par de claves RSA cuya clave pública se copiará en dispositivos para configurarlos remotamente:

```
ssh-keygen
```

# B

## Preparación de las Raspberry PIs

Las Raspberry Pi se configuraron para poder usarse remotamente; a continuación, se detallan los pasos seguidos:

### 1. Inicio y configuración Internet

Se iniciaron por primera vez conectadas por HDMI y se configuraron las conexiones a Internet tanto por interfaz Wi-Fi como por Ethernet.

### 2. Actualización del sistema operativo

Se actualizó la distribución Raspbian con:

```
sudo apt update  
sudo apt full-upgrade -y
```

### 3. Instalación del servidor SSH

Se instaló el paquete `openssh-server`:

```
sudo apt install -y openssh-server
```

### 4. Configuración de SSH

En `/etc/ssh/sshd_config` se ajustaron los siguientes parámetros:

```
Port 2000  
ListenAddress 192.168.40.102  
PasswordAuthentication no  
AuthorizedKeysFile .ssh/HP-Server  
PermitRootLogin no  
AllowUsers pi-1
```

Reinicio del servicio SSH:

```
sudo systemctl restart ssh
```

Conexión desde HP-Server mediante SSH a las Raspberry Pi:

```
ssh -p 2000 pi-<N>@<IP-RPI>
```

# C

## Instalación y configuración de Cowrie

A continuación, se presentan en detalle los pasos seguidos para la instalación de Cowrie en la pi-1:

### 1. Requisitos iniciales

Se creó un usuario Cowrie sin permisos de root y se instalaron las dependencias necesarias, entre ellas python, con el que se utilizará un entorno virtual para tener aislamiento de las dependencias y para no requerir privilegios de sistema ya que se guardan en la propia carpeta del entorno.

### 2. Clonación del repositorio

Se clonó y se siguió la configuración inicial de Cowrie:

```
https://github.com/cowrie/cowrie
```

### 3. Creación de entorno virtual y dependencias

Se accede a la carpeta correspondiente y se crea el entorno virtual donde se instalan las librerías de python usando requirements.txt del propio Cowrie:

```
cd /home/cowrie/cowrie
python3 -m venv cowrie-env
source cowrie-env/bin/activate
pip install --upgrade pip
pip install -r requirements.txt
```

### 4. Prueba inicial

Tras tener la configuración basica se inicia por primera vez Cowrie utilizando los archivos binarios que se te ofrecen en la carpeta /cowrie/bin:

```
bin/cowrie start #Se inicia
bin/cowrie status #Se comprueba su estado
```

Se observa que hay un sistema simple de Unix con algunas carpetas como bin, home, var, etc. En el home hay un usuario phil y apenas archivos.

---

## 5. Ajustes en cowrie.cfg

Para los ajustes clave de Cowrie esta la carpeta /etc donde están los archivos de configuración cowrie.cfg.dist con los datos por defecto. Incluye tambié userdb.example que contiene una serie de combinaciones usuario:contraseña con las que se puede hacer login. La documentación te hace copiar esos archivos a otros con nombre cowrie.cfg y userdb.txt. Estos archivos son muy importantes ya que serán los primeros que busque Cowrie para buscar los ajustes y si falta algún parámetro lo buscará en los archivos iniciales. Se realizaron los siguientes cambios en cowrie.cfg [30]:

- **Hostname** - Se puso el mismo nombre que el ordenador central HP-Server ya que el objetivo es simularlo al completo.
- **Puerto SSH** - Puerto 22 por defecto al 2222 para no estar dentro de los well-known ports.
- **Puerto Telnet** - Puerto 23 por defecto al 2223.
- **userdb\_file = etc/userdb.txt** - Archivo con usuarios señuelo y sus contraseñas.
- **logfile = cowrie.json** - Ruta donde se almacenan los logs JSON aunque el archivo es un enlace simbólico que apunta a un USB donde se almacena todo.

Para que el atacante piense que esta atacando a los verdaderos puertos 22 y 23 en la documentación indica que se redirigan a los de Cowrie:

```
sudo iptables -t nat -A PREROUTING \
    -p tcp --dport 22 -j REDIRECT --to-ports 2222

sudo iptables -t nat -A PREROUTING \
    -p tcp --dport 23 -j REDIRECT --to-ports 2223
```

## 6. Configuración de sistemas de archivos

Para simular al máximo los archivos se utilizó un comando propio de linux que es rsync que permite la sincronización de archivos y directorios entre dispositivos por lo que se clonaron todas las carpetas del HP-Server. El comando utilizado fue:

```
sudo rsync -av -d / /Cowrie/kali_skeleton /
```

De esta manera se tienen todos los directorios los cuales luego se poblaron con archivos falsos. Unas carpetas tenían más archivos que otras pero las que más atención recibieron son:

- **/bin** - Contiene los archivos binarios que son los comandos que el atacante “podría” llegar a ejecutar en un sistema real. No todos esos comandos están integrados en Cowrie por lo que a pesar de estar el archivo binario no funciona el comando.
- **/home** - Donde se tiene la estructura típica de un directorio de usuario. El HP-Server sería un usuario servidor con archivos de configuración de una empresa. Se creó el usuario fernando donde hay información personal incluso una tarjeta virtual de Revolut real sin dinero que podría ser robada y utilizada.

Para mostrar esta estructura de ficheros se utiliza el comando:

```
python3 bin/createfs \
    -l /Cowrie/kali_skeleton/
    -o fs.pickle
```

Este comando genera un archivo fs.pickle que es el que Cowrie utiliza para mostrar al atacante. Este archivo trae tanto la estructura como los permisos de la carpeta que se le indique. Además, contiene todos la ruta de los archivos reales para cuando el atacante quiera hacer un cat o descargar algún archivo.

# D

## Instalación y configuración de T-Pot

La instalación del T-Pot se ha realizado en un Dell XPS 13 9370 tras comprobar que disponía de la capacidad computacional requerida por T-Pot. Además, es necesario disponer de docker y docker compose y de los instaladores de apt y curl.

### 1. Clonación e instalación del repositorio

Descarga e instalación como no root desde el directorio \$HOME:

```
env bash -c "$(curl -sL \
https://github.com/telekomsecurity/tpotce/raw/master/install.sh)"
```

#### Credenciales por defecto:

Durante la instalación de T-Pot se te piden que pongas credenciales que serán las utilizadas posteriormente para acceder a la interfaz gráfica.

### 2. Resolver conflictos

Al utilizar tanta cantidad de servicios, es muy común que haya ciertos conflictos [31] por puertos que ya están siendo utilizados, por lo que hay que detectarlos primero con:

```
docker ps
```

Esto te muestra los contenedores con problemas y, en el caso de que sea por un puerto que ya está siendo utilizado, se detecta buscando su PID:

```
sudo netstat -tulpn | grep <PUERTO>
sudo lsof -i <PUERTO>
```

A partir del PID y del proceso es necesario decidir si es posible “matar” el proceso o hay que iniciararlo en otro puerto (esto no afecta ya que luego se pueden redireccionar los puertos). Además, también aparece el tipo de servicio que es, por lo que también tienes la opción de pararlo. Se pueden usar estos comandos:

```
Matar el proceso: sudo kill <PID>
Detener el servicio: sudo systemctl stop <SERVICE>
```

---

### 3. Acceso interfaz web de T-Pot

Una vez el stack de T-Pot esta completamente funcionando, se puede utilizar la interfaz que te ofrece para acceder a distintas herramientas. Entre ellas está un mapa con los ataques sufridos en tiempo real y lo más importante la interfaz de Kibana (ELK) de la que dispone, que te permite visualizar todos los eventos capturados. Desde cualquier navegador de la red local se puede acceder a su interfaz con:

```
https://<IP_DE_TPOT>:64297
```

- **Discover:**

Contiene todos los logs JSON los cuales se pueden visualizar y filtrar según quiera el usuario.

- **Dashboards:**

T-Pot incluye paneles preconfigurados para MQTT, Telnet, SSH, etc.

### 4. Uso de Docker

Desde la carpeta de instalación que está en el directorio \$HOME se pueden controlar los contenedores gracias al archivo docker-compose.yml.

- **Arrancar y detener** sensores y servicios con un único comando:

```
docker compose up -d      % segundo plano  
docker compose down       % detener
```

- **Reiniciar** un servicio puntual sin afectar al resto:

```
docker-compose restart <SERVICE>
```

- **Ver logs** en tiempo real de un contenedor concreto:

```
docker compose logs -f elasticsearch
```

- **Probar cambios de configuración.** Editando docker-compose.yml o los ficheros .yml de cada servicio se pueden realizar modificaciones para recrear mejor los servicios con más realismo y reactivando los contenedores con docker-compose up -d. Un ajuste interesante es quitar el reinicio periódico que sufre cada día.