GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

HONEPOT PARA EL ANÁLISIS DE ATAQUES EN DISPOSITIVOS IoT

Autor: Ernesto Hidalgo Felipe

Director: Dr. Gregorio López López

Co-Director: Dr. Rafael Palacios Hielscher

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

HONEYPOT PARA EL ANÁLISIS DEATAQUES EN DISPOSITIVOS IoT

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2022/23 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Ernesto Hidalgo Felipe Fecha: ..04../ ..07../ ..23..

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: RAFAEL PALACIOS Fecha: ..04../ ..07../ ..23..

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

HONEPOT PARA EL ANÁLISIS DE ATAQUES EN DISPOSITIVOS IoT

Autor: Ernesto Hidalgo Felipe

Director: Dr. Gregorio López López

Co-Director: Dr. Rafael Palacios Hielscher

Madrid

A mis padres, por darme la infancia más feliz y su amor incondicional.

A mi hermano, por cogerme de la mano cuando no sabía andar.

A mis amigos, por hacer de mi vida una experiencia preciosa.

A Stefano, por cuidarme. A Giovanni, por cuidar de quien me cuida.

A Elena, por quererme, tanto y tan bien.

A Dios. A la matriz.

HONEYPOT PARA EL ANÁLISIS DE ATAQUES EN DISPOSITIVOS IOT

Autor: Hidalgo Felipe, Ernesto.
Director: López López, Gregorio; Palacios Hielscher, Rafael.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas.

RESUMEN DEL PROYECTO

El proyecto implementa honeypots en una Raspberry Pi para simular un entorno IoT y analizar los ciberataques que recibe. Los datos recopilados se visualizan en búsqueda de tendencias en sus tácticas y cualquier otra información que nos pueda ser útil. Los hallazgos destacan la importancia, creciente, de la seguridad en este sector, y la eficacia del método de anticipación y respuesta propuesto.

Palabras clave: Honeypot, IoT, Ciberseguridad, Visualización, Protocolos, Puertos

1. Introducción

El proyecto se centra en el análisis de ataques a redes de dispositivos del Internet de las Cosas (IoT). El IoT es un campo que está cobrando cada vez más relevancia; los dispositivos conectados a la red y los datos que estos generan aumentan a un ritmo exponencial y muy acelerado.

El IoT ha cambiado la forma en la que entendemos las telecomunicaciones. El uso de la tecnología ha evolucionado de su pasado, consciente y activo, a un paradigma pasivo y constante, siempre presente. Su proliferación ha venido acompañada, de forma inevitable, de un aumento de los ciberataques, comprometiendo la seguridad de nuestra información a causa de malas prácticas en su uso y desarrollo.

Los honeypots son sistemas de seguridad informática diseñados para actuar como señuelos, atrayendo a los ciberdelincuentes para que ataquen estos sistemas en lugar de los objetivos reales. Al simular ser sistemas legítimos y vulnerables, los honeypots pueden recopilar información valiosa sobre los métodos, tácticas y motivaciones de los atacantes. Este enfoque proactivo de la ciberseguridad nos puede permitir anticiparnos a las amenazas emergentes, mejorar las defensas existentes y desarrollar estrategias de respuesta más efectivas.

2. Definición del proyecto

El TFG consiste en la simulación de un entorno IoT cotidiano, un señuelo. A través de la implementación de dos honeypots, Cowrie y Dionaea, en una Raspberry Pi, podremos actuar como intermediarios de la red y medir así la frecuencia, cantidad y tipo de ataques que esta recibe para, posteriormente, estudiar las tendencias de los hackers y trazar posibles líneas de anticipación y respuesta.

El proyecto se esfuerza por mantenerse relevante y actualizado en el contexto de la creciente prevalencia de los dispositivos IoT en la vida cotidiana. A medida que estos se vuelven cada vez más omnipresentes, también lo hacen las vulnerabilidades y amenazas asociadas a ellos. Por lo tanto, este proyecto no solo busca entender y analizar las tácticas actuales de los ciberdelincuentes, sino también anticipar y prepararse para los peligros futuros.

Además, el trabajo se inspira y basa en investigaciones previas en el campo (1), aprovechando los avances y descubrimientos ya realizados, y construye sobre ellos una nueva línea de acción. El enfoque que le hemos dado hace especial hincapié en la visualización y análisis de los datos, parte fundamental del proceso y a partir de la cual inferiremos nuestras conclusiones.

3. Descripción del modelo

El sistema que implementamos se basa en una red compuesta por una Raspberry Pi 3+ Modelo B, varios dispositivos IoT y un router que proporciona acceso a Internet. Los dispositivos IoT, que incluyen *wearables*, una webcam y un *smartphone*, se conectan a una subred wifi creada específicamente para este fin.

Los honeypots Cowrie y Dionaea se instalan en la Raspberry Pi. Cowrie se configura para emular los servicios Telnet y SSH, mientras que Dionaea ofrece una amplia gama de protocolos, incluyendo FTP, HTTP o SMB, entre otros. Ambos funcionarán durante 45 días, ininterrumpidamente.

Finalmente, los puertos del router asociados a los servicios descritos se abren al exterior, permitiendo que los honeypots sean detectados por otros equipos desde otras partes del mundo. El sistema se encarga de analizar, registro tras registro, los ataques que recibimos.

Una vez anotadas todas las conexiones entrantes, nos valemos de herramientas de gestión, como SQLite, para interpretar los datos obtenidos, y de otras de visualización, como Tableau, para proyectar gráficamente los resultados. A partir de las visualizaciones que creamos, podremos ser capaces de intuir patrones y tendencias.

4. Resultados

El proyecto comenzó con la puesta en marcha de los honeypots y la espera de los primeros ataques para confirmar su funcionamiento. Los ataques no tardaron en llegar, con intentos de inicio de sesión a través de SSH desde varias direcciones IP, incluyendo una de China y otra de Estados Unidos.

Después de 45 días dedicados a la recopilación de datos, se obtuvieron resultados significativos. El honeypot Dionaea proporcionó una base de datos SQLite de 679MB de registros de la interacción de los atacantes con el equipo. A pesar de los desafíos iniciales que nos supuso la capacidad de la CPU para manejar la gran cantidad de datos, pudimos minarlos y visualizarlos, y se encontraron patrones interesantes. Por ejemplo, el protocolo más explotado fue, con diferencia, SipSession, seguido de smbd y FTP.

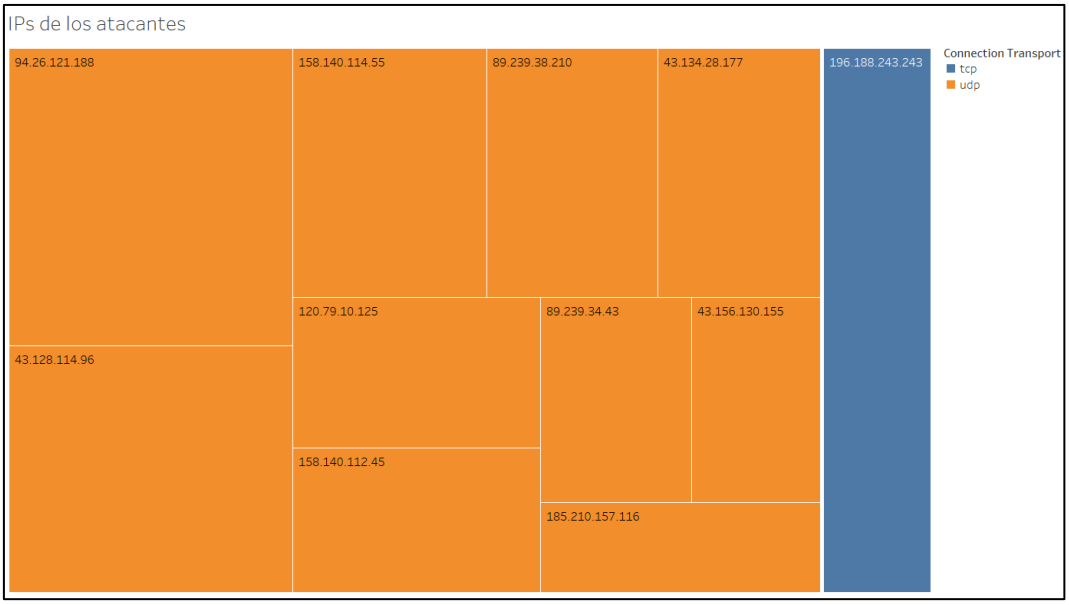

Ilustración 1 – IPs más comunes de los atacantes -Dionaea- y sus protocolos de transporte

Por otro lado, el honeypot Cowrie recopiló 333MB de información en ficheros diarios JSON. Los resultados estadísticos mostraron que las direcciones IP más comunes de los atacantes provenían de una variedad de países que no guardaban una relación concreta, y se observaron intentos de inicio de sesión con credenciales predeterminadas habituales, lo que subraya la importancia de configurar nuestros equipos de forma segura y cauta.

5. Conclusiones

Las conclusiones de este proyecto subrayan la importancia de la seguridad en los entornos del Internet de las Cosas (IoT). La implementación de honeypots ha demostrado ser una herramienta eficaz para medir y analizar la frecuencia, cantidad y tipo de ataques que puede recibir una red.

Este estudio materializa también la utilidad de las herramientas de visualización de datos para analizar y manejar conjuntos de información masivos.

Los resultados obtenidos nos han proporcionado una visión valiosa de las tácticas utilizadas por los ciberdelincuentes, incluyendo la explotación de protocolos comunes, puertos inseguros y el uso de credenciales fácilmente adivinables; un boceto de su *modus operandi*.

6. Referencias

(1) Tabari, A. Z., Ou, X., & Singhal, A. (2021). What are Attackers after on IoT Devices? Cornell University. Obtenido de: https://arxiv.org/pdf/2112.10974.pdf

HONEYPOT FOR THE ANALYSIS OF ATTACKS ON IOT DEVICES

Author: Hidalgo Felipe, Ernesto.
Supervisor: López López, Gregorio; Palacios Hielscher, Rafael.
Collaborating Entity: ICAI – Universidad Pontificia Comillas.

ABSTRACT

The project implements honeypots on a Raspberry Pi to simulate an IoT environment and analyse the cyberattacks it receives. The collected data is visualized in search of trends in their tactics and any other information that may be useful. The findings highlight the growing importance of security in this sector, and the effectiveness of the proposed anticipation and response method.

Keywords: Honeypot, IoT, Cybersecurity, Visualization, Protocols, Ports

1. Introduction

The project focuses on the analysis of attacks on Internet of Things (IoT) device networks. IoT is a field that is gaining more and more relevance; the devices connected to the network and the data they generate are increasing at an exponential and very accelerated rate.

IoT has changed the way we understand telecommunications. The use of technology has evolved from its conscious and active past to a passive and constant paradigm, always present. Its proliferation has inevitably been accompanied by an increase in cyberattacks, compromising the security of our information due to poor practices in its use and development.

Honeypots are computer security systems designed to act as decoys, attracting cybercriminals to attack these systems instead of the real targets. By simulating legitimate and vulnerable systems, honeypots can collect valuable information about the methods, tactics, and motivations of attackers. This proactive approach to cybersecurity can allow us to anticipate emerging threats, improve existing defenses, and develop more effective response strategies.

2. Project Definition

The TFG consists of simulating a daily IoT environment, a decoy. Through the implementation of two honeypots, Cowrie and Dionaea, on a Raspberry Pi, we can act as network intermediaries and thus measure the frequency, quantity, and type of attacks it receives to later study hacker trends and draw possible lines of anticipation and response.

The project strives to remain relevant and up to date in the context of the growing prevalence of IoT devices in everyday life. As these become increasingly ubiquitous, so do the vulnerabilities and threats associated with them. Therefore, this project not only seeks to understand and analyze the

current tactics of cybercriminals but also to anticipate and prepare for future dangers.

In addition, the work is inspired and based on previous research in the field (1), leveraging the advances and discoveries already made, and builds on them a new line of action. The approach we have given it places special emphasis on data visualization and analysis, a fundamental part of the process from which we will infer our conclusions.

3. Model Description

The system we implement is based on a network composed of a Raspberry Pi 3+ Model B, various IoT devices, and a router that provides internet access. The IoT devices, which include wearables, a webcam, and a smartphone, connect to a wifi subnet created specifically for this purpose.

The Cowrie and Dionaea honeypots are installed on the Raspberry Pi. Cowrie is configured to emulate Telnet and SSH services, while Dionaea offers a wide range of protocols, including FTP, HTTP, or SMB, among others. Both will run for 45 days, uninterruptedly.

Finally, the router ports associated with the described services are opened to the outside, allowing the honeypots to be detected by other teams from other parts of the world. The system takes care of analyzing, record by record, the attacks we receive.

Once all incoming connections have been noted, we use management tools, such as SQLite, to interpret the obtained data, and other visualization tools, such as Tableau, to graphically project the results. From the visualizations we create, we will be able to intuit patterns and trends.

4. Results

The project began with the start-up of the honeypots and the wait for the first attacks to confirm their operation. The attacks did not take long to arrive, with attempts to log in via SSH from various IP addresses, including one from China and another from the United States.

After 45 days dedicated to data collection, significant results were obtained. The Dionaea honeypot provided a 679MB SQLite database of attacker interaction records with the team. Despite the initial challenges posed by the CPU's ability to handle the large amount of data, we were able to mine and visualize them, and interesting patterns were found. For example, the most exploited protocol was, by far, SipSession, followed by smbd and FTP.

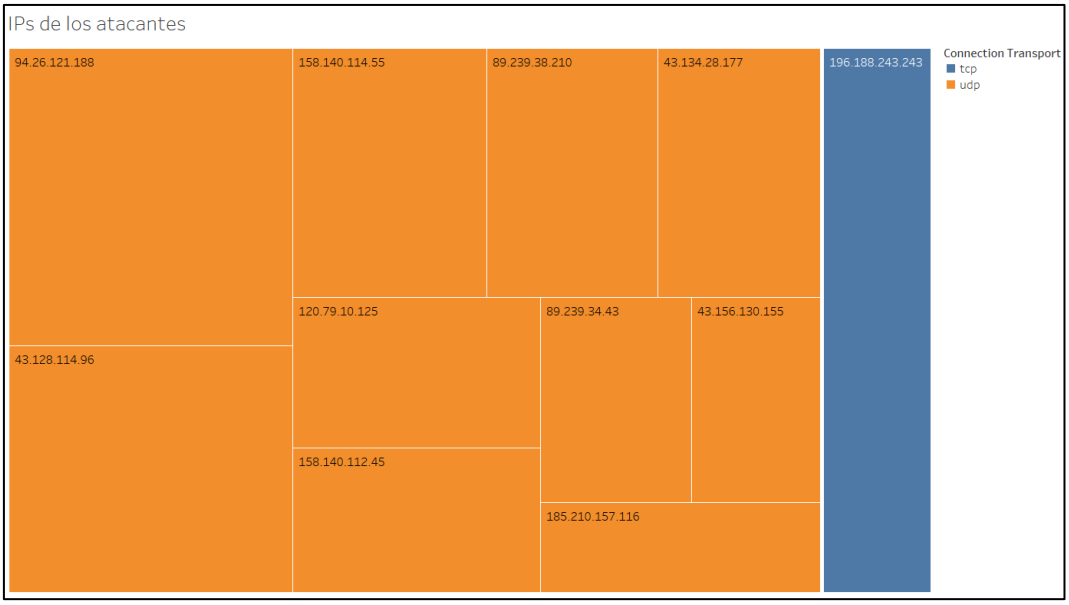

Illustration 2 – Most common attacker IPs -Dionaea- and their transport protocols

On the other hand, the Cowrie honeypot collected 333MB of information in daily JSON files. The statistical results showed that the most common attacker IP addresses came from a variety of countries that had no specific relationship, and there were attempts to log in with usual default credentials, which underscores the importance of setting up our equipment securely and cautiously.

5. Conclusions

The conclusions of this project underline the importance of security in Internet of Things (IoT) environments. The implementation of honeypots has proven to be an effective tool for measuring and analyzing the frequency, quantity, and type of attacks that a network can receive.

This study also materializes the usefulness of data visualization tools for analyzing and handling massive data sets.

The results obtained have provided us with a valuable insight into the tactics used by cybercriminals, including the exploitation of common protocols, insecure ports, and easily guessable credentials; a sketch of their *modus operandi*.

6. References

(1) Tabari, A. Z., Ou, X., & Singhal, A. (2021). What are Attackers after on IoT Devices? Cornell University. Retrieved: https://arxiv.org/pdf/2112.10974.pdf

Índice de la memoria

Índice de figuras

Índice de tablas

Capítulo 1. Introducción

1.1 Justificación y motivación del proyecto

El Internet of Things, o IoT, se define como la "interconexión de dispositivos, sensores y objetos de uso cotidiano que generan, consumen e intercambian datos bajo una intervención humana mínima" [1].

Bajo esta premisa ha nacido una nueva manera de entender las telecomunicaciones: hasta ahora, utilizábamos la tecnología de manera activa, con una participación consciente y específica; a causa del auge del IoT, el potencial de los avances más recientes se ha basado en un uso pasivo y constante de la red, dedicado a un rango de tareas muy amplio, que abarca desde los aspersores de un hogar hasta la geolocalización de ganado en una granja.

Tanto es así que existen fuentes que afirman que durante esta década se podrían alcanzar entre treinta [2] y cien mil millones de dispositivos IoT conectados, con un impacto global que superaría los 11 billones de dólares [1].

En este contexto es lógico y vital hacer hincapié en la seguridad de dichos dispositivos. Las malas prácticas que hasta ahora abundan en su desarrollo han permitido que proliferen una gran cantidad de ciberataques; crecientes de manera directamente proporcional a la tendencia del sector [3]. La privacidad de los usuarios se ve a menudo comprometida, e información que debería ser confidencial pasa a manos de hackers e intrusos con diversas intenciones, entre las que la económica se repite más a menudo.

Por todo ello, nuestra propuesta consiste en la creación de honeypots; sistemas señuelo que actuarán como man-in-the-middle analizando el tráfico en un entorno IoT que simularemos con distintos dispositivos de uso cotidiano (relojes inteligentes, SPAs, webcams, teléfonos móviles…). De esta manera, seremos capaces de medir la frecuencia, cantidad y tipo de ataques que recibe la red para así realizar un estudio sobre cómo proceden los hackers, sus tendencias y estrategias, trazando posteriormente posibles líneas de anticipación y respuesta.

Cabe destacar que la idea del proyecto nace, como se explicará en detalle en el *Estado del arte*, de la mano de Andrea Fariña Fernández-Portillo quien, en julio de 2022, desarrolló su TFM en torno a este propósito, sentando las bases del texto presente [4].

Como la autora afirmaba en su escrito, el trabajo "se alinea con los objetivos del proyecto europeo **RAYUELA** (empoweRing and educAting YoUng pEople for the internet by pLAying)". El proyecto, liderado por la Universidad de Comillas y financiado por la UE, se enfoca, de manera multidisciplinar, en entender el comportamiento en línea de los usuarios y su interacción con la ciberdelincuencia [5], y casa perfectamente con la metodología que propondremos. Su logotipo, símbolo del enfoque educativo y amigable que pretende adoptar, se aprecia en la Figura 1.

Figura 1. Logotipo del proyecto europeo RAYUELA

Una vez hecha evidente la necesidad de investigar en el campo, podemos especificar que este TFG funcionará, simultáneamente, como un ejercicio académico que estudia y sigue los pasos dados por otros autores, y como un planteamiento de alternativas y posibilidades que de estos emanen.

El trabajo se podría acercar a la idiosincrasia del ejemplo desarrollado por Tabari, Ou y Singhal en 2021 [6] (y del que hablaremos en el *Estado del arte)*, que primaba en particular la experimentación, recogiendo datos sobre los tipos de ataque, su cantidad y frecuencia y, en conclusión, las formas de actuar de los ciberdelincuentes.

Este tipo de proyectos es imprescindible para la sostenibilidad de un mundo como el de IoT, que involucra a tantísimos usuarios a lo largo del planeta. La información, en muchos casos sensible y confidencial, que exponemos a través de nuestros dispositivos nos podría

comprometer de formas inimaginables: datos médicos privados, vídeos de cárceles o guarderías, información de administraciones públicas o, sin ir más lejos, los patrones de los habitantes de un hogar.

Basta con dejar volar un poco la imaginación para darse cuenta de cómo puede afectarnos esta realidad a nivel personal y, sobre todo, económico. No en vano está a la orden del día la extorsión por filtración o robo de datos, o el fraude bancario, por ejemplo.

La motivación del TFG surge, por ende, de seguir entendiendo el irrefrenable paradigma de la seguridad en el Internet de las Cosas mediante un análisis exhaustivo del hacer del hacking; una materialización de la realidad que vivimos que servirá para sentar nuevos objetivos sobre las garantías que los dispositivos y sus empresas ejerzan.

1.2 Objetivos del TFG

Con el proyecto perseguimos desarrollar honeypots que atraerán y minarán los que inferiremos como ataques más comunes en el IoT. Para ello, seguiremos los siguientes objetivos:

1) **Estudio del concepto de honeypot y sus diferentes tipos y aplicaciones.** En primer lugar, haremos un breve análisis de los diseños de honeypot existentes para discernir cuál nos puede ser útil. También repasaremos los ataques más comunes en este ecosistema.
2) **Propuesta de dispositivos IoT.** De igual forma, estudiaremos qué dispositivos de los más comunes en IoT podemos incluir en la arquitectura a desarrollar, junto a sus características principales.
3) **Diseño de la arquitectura.** Tras decidir qué elementos van a conformar el honeypot realizaremos un diseño de la que será nuestra red señuelo, especificando sus propiedades y singularidades en materia de tráfico, conexiones…
4) **Programación de la Raspberry Pi.** El elemento principal de la red. Para que nuestra Raspberry Pi funcione como *man-in-the-middle*, deberemos programarla con ese fin.
5) **Implementación del sistema.** Una vez cumplidos los pasos previos, podremos establecer nuestra red honeypot para empezar a trabajar con ella.

6) **Recolección y medición de resultados.** Durante el que esperamos sea un periodo considerable, recogeremos todos los datos que los honeypot puedan medir en materia de los ataques que sufran a diario.
7) **Análisis de los mismos.** Una vez disponible una base de datos empírica de los ciberataques, nos dispondremos a analizar, clasificar, interpretar y describir estos.
8) **Conclusiones y margen de acción futuro.** Tras la realización del proyecto seremos capaces de sacar un diagnóstico en claro, a partir del cual se leerán propuestas de acción e ideas de cambio y mejora.

1.3 Metodología de la investigación

El problema planteado se resolverá por medio de la creación de una infraestructura que simulará la de un conjunto de dispositivos IoT al uso. Para comenzar, llevaremos a cabo una investigación exhaustiva de las posibilidades que se nos presentan y el porqué de primar unas sobre otras (tipos de honeypot escogidos, ataques y vulnerabilidades que se tienen en cuenta, arquitectura de los dispositivos, tráfico de la red…).

Cuando esta se diseñe e instale, nos valdremos de una Raspberry Pi 3 Model B+ debidamente programada para cumplir como man-in-the-middle o intermediario del tráfico, analizando y almacenando los datos de todas las conexiones que se produzcan en el entorno del Internet de las Cosas. Tanto la Raspberry como algún dispositivo se pueden observar en la Figura 2.

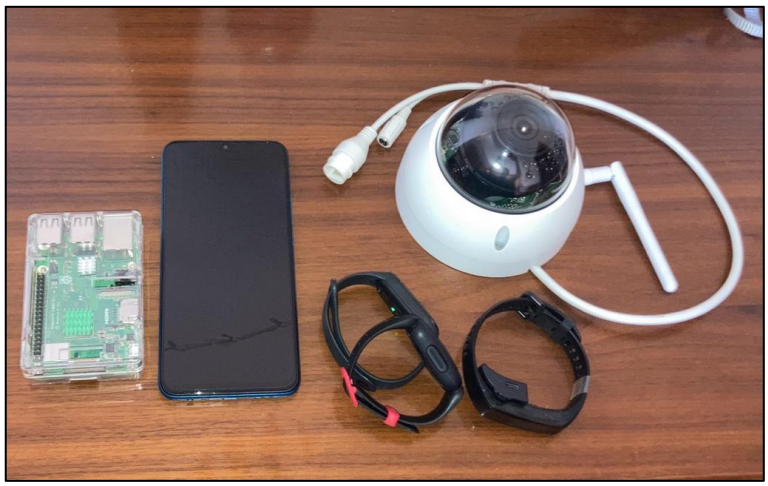

Figura 2. Algunos dispositivos de la arquitectura

Para esta sección hemos tenido ocasión de debatir posibles maneras de proceder. En un primer momento se estudió montar el honeypot en Azure [7], lo cual presentaba algunas ventajas y otras desventajas:

Ventajas:

- **Facilidad de despliegue**: Con Azure, podríamos desplegar y configurar rápidamente el honeypot sin tener que adquirir y configurar hardware adicional.
- **Elasticidad y escalabilidad**: Azure puede proporcionar más recursos a medida que un honeypot recibe más tráfico, lo que podría ser útil si estuviésemos tratando de estudiar ataques a gran escala.
- **Aislamiento**: Al alojar tu honeypot en Azure, puedes mantenerlo aislado de tu infraestructura interna, reduciendo el riesgo de que un ataque a tu honeypot se propague a tus sistemas reales.

Desventajas:

- **Coste**: Aunque Azure puede ofrecer costes iniciales bajos, estos pueden aumentar rápidamente con el uso intensivo. Los ataques DDoS o cualquier otro ataque de gran volumen pueden consumir muchos recursos, lo que podría resultar en altas tarifas.
- **Dependencia del proveedor**: Al usar Azure para el honeypot, seríamos dependientes de nuestro proveedor para el soporte y mantenimiento de la red. En general, nos limita a la hora de tomar según qué decisiones.
- **Veracidad del señuelo**: Para un atacante que utilice métodos algo más sofisticados sería muy fácil detectar que la red que planteamos se limita al plano virtual.

Finalmente, llegamos a la conclusión de que esta decisión levantaría sospechas de nuestro señuelo, haciéndolo menos atractivo para los atacantes. Además, veíamos problemas en la configuración del firewall del centro, que muy probablemente iba a bloquear parte del tráfico.

En otra línea, se planteó la posibilidad de solicitar una línea de fibra óptica dedicada específicamente para el proyecto, pero esta realidad no se pudo materializar. Aún así, creemos que este escenario habría sido el ideal.

La red de dispositivos se terminará por establecer en nuestro hogar, lo que nos facilita la monitorización continua de los resultados y su procesado. En la *Implementación del diseño* desarrollaremos mejor esta idea.

Una vez recogida la información, procederemos a estudiarla. Le aplicaremos distintas formas de filtrado y selección, con software de análisis y representación de datos con el que ya hemos podido trabajar previamente. Así, trazaremos un boceto del perfil de los atacantes y sus principales estrategias para establecer después vías de prevención, contención y respuesta.

En el diagrama de Gantt que sigue en la Figura 3 se puede ver con detalle cómo pensamos repartir las tareas descritas, por medio de un cronograma planificado y concreto.

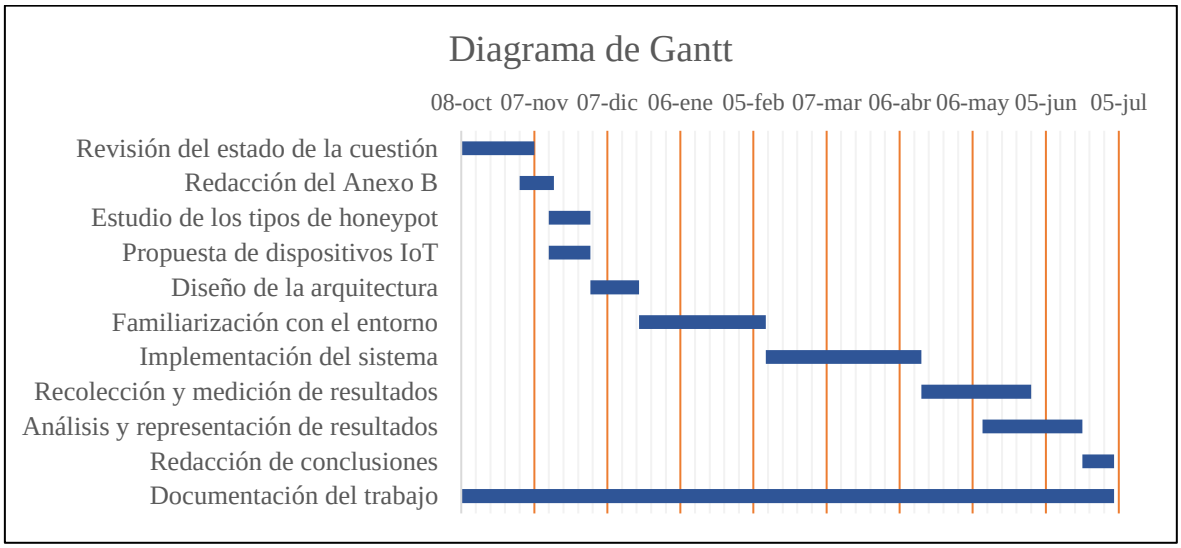

Figura 3. Planificación propuesta

Capítulo 2. Marco teórico

2.1 Contexto y antecedentes. Estado del arte

Existen bastantes ejemplos de aplicación de honeypots al mundo del IoT. Entre ellos, unos cuantos TFMs y tesis, como el de Castro Astroz [8], que utilizaba el mismo tipo de honeypots que se propondrá en este documento; o el de Armiñana Gorriz, que seguía una línea de pensamiento similar [9].

A su vez, existen *papers* académicos que se han interesado por el mismo objetivo, véase el trabajo IoT Honeypot: A Review from Researcher's Perspective [10], donde se realiza una exposición teórica de los tipos existentes de honeypot junto a sus posibles aplicaciones y, en general, el estado actual del mundo IoT; o el interesantísimo What are Attackers after on IoT Devices? [6], que explora las vulnerabilidades que explotan los atacantes e infiere sus patrones y tendencias, extrayendo conclusiones muy útiles.

Como se mencionó en la Introducción, este mismo TFG bebe del TFM de Andrea Fariña Fernández-Portillo, que en julio de 2022 expuso junto a la Universidad de Comillas el diseño de honeypots sobre el que se inspira mi proyecto.

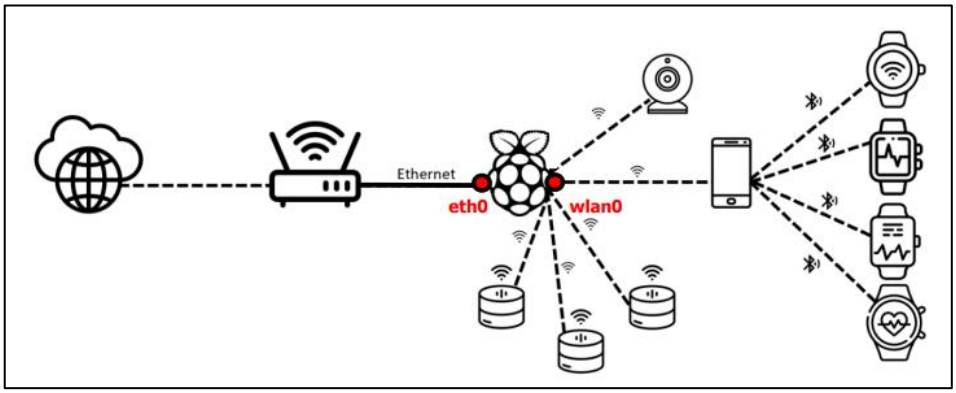

Figura 4. Arquitectura propuesta por Andrea Fariña

Se puede apreciar, entonces, que la documentación existente al respecto es inmensa. Sin embargo, este no es motivo para no plantear un estudio análogo. Como se expuso previamente, el sector del IoT se está convirtiendo en un gigante y con él, sus deficiencias.

Así, serán siempre pocas las investigaciones que se lleven en base a las vulnerabilidades de estos dispositivos, al menos de momento. Además, en un mundo tan rápidamente cambiante, no sería de extrañar que nuestras conclusiones distasen de las ya alcanzadas en el pasado.

2.2 Definición y clasificación de honeypots

Un **honeypot** es un sistema informático diseñado para atraer ciberataques. Simula ser un objetivo de interés para los atacantes, presentando vulnerabilidades ficticias que estos tratan de aprovechar en su beneficio. De esta forma, se puede analizar su proceder y recabar datos de gran interés para el desarrollo de medidas de protección contra estos [11].

2.2.1 Características

Existen muchas modalidades de honeypot. Por un lado, los podemos diferenciar según la funcionalidad que pretenden simular, ya sea un email expuesto a spam, una base de datos sensibles o la actividad de un usuario en la red [12].

Por otro lado, los honeypots se pueden definir según su nivel de interacción. Los de baja interacción consumen pocos recursos y no generan demasiados datos, solo una cantidad esencial (dirección IP del usuario, puertos, protocolos…). Por otro lado, los de alta interacción tienen como fin retener al atacante durante más tiempo, ofreciéndole una plataforma con la que interactuar y "activos" a explotar.

Nuestros señuelos presentan otras características relevantes, como su rol (inicia las conversaciones -activo- o espera a recibirlas -pasivo-), o la aplicación a la que se destina (dispositivos IoT, redes industriales…). También cabe mencionar su margen de acción, existiendo honeypots meramente informativos y otros, los *sticky*, que tratan de ralentizar un ataque para proteger el resto del sistema [13].

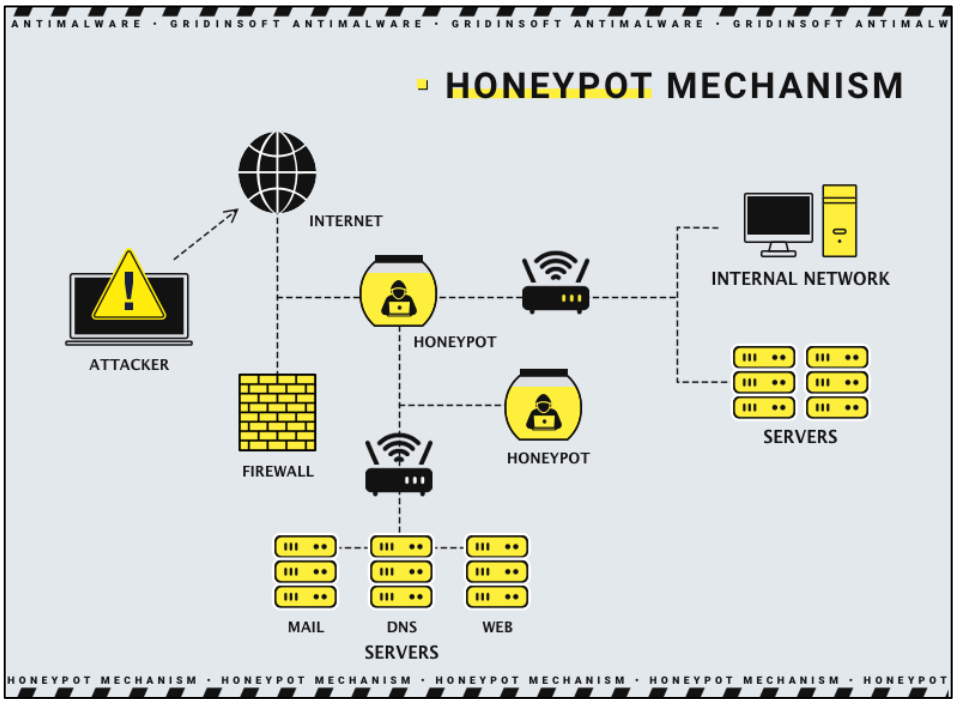

Figura 5. Mecanismo de un honeypot

2.3 Cowrie y Dionaea

Los honeypots con los que elegimos trabajar son dos: Cowrie [14] y Dionaea [15].

2.3.1 Cowrie

Cowrie es un honeypot de **interacción media-alta** que emula los servicios **Telnet** y **SSH**. Su uso nos es de gran interés, ya que le ofrece al atacante una estructura completa de directorios y ficheros con los que puede interactuar, haciendo así muy creíble la experiencia y ofreciéndonos, a cambio, una gran cantidad de información sobre la actividad del delincuente.

Michel Oosterhof desarrolló Cowrie como un *fork* de Kippo [16], otro honeypot que pensamos que habría podido ser apropiado para el proyecto. Sin embargo, Cowrie presenta algunas mejoras que nos han hecho decantarnos por él:

- Soporte para protocolo de transferencia de **archivos SFTP**: Cowrie tiene soporte integrado para el protocolo de transferencia de archivos SFTP, lo que puede dar una visión más completa de las acciones de un atacante.
- Emulación de **más comandos** del sistema Unix: Cowrie puede emular un conjunto más amplio de comandos del sistema Unix que Kippo, lo que lo hace más convincente para los atacantes y puede permitirnos recopilar más información sobre sus acciones.
- Mejoras de seguridad y **mantenimiento**: Cowrie se actualiza con más frecuencia que Kippo, lo que significa que generalmente tiene una mejor seguridad y mantenimiento. Kippo ha dejado de actualizarse activamente, lo que podría dejarlo vulnerable a ciertos tipos de ataques.

Además, es un honeypot relativamente sencillo de establecer, y resume, cada día, en un archivo JSON, todos sus registros, lo cual es muy cómodo para su posterior procesado.

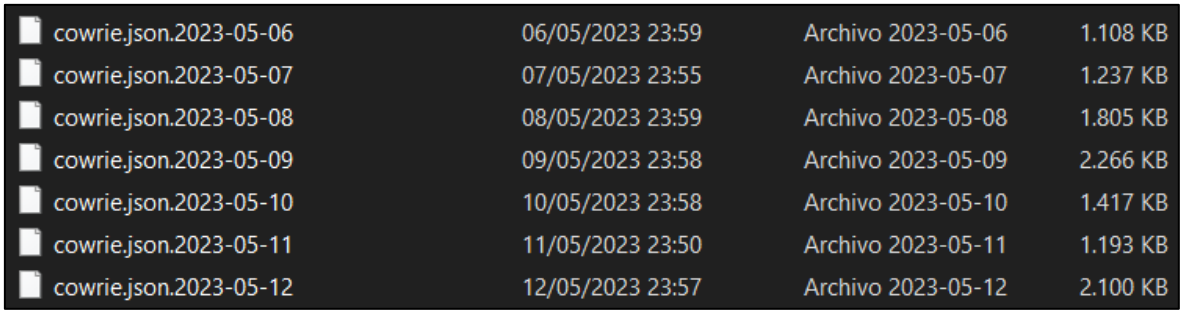

Figura 6. Ficheros de registro de Cowrie

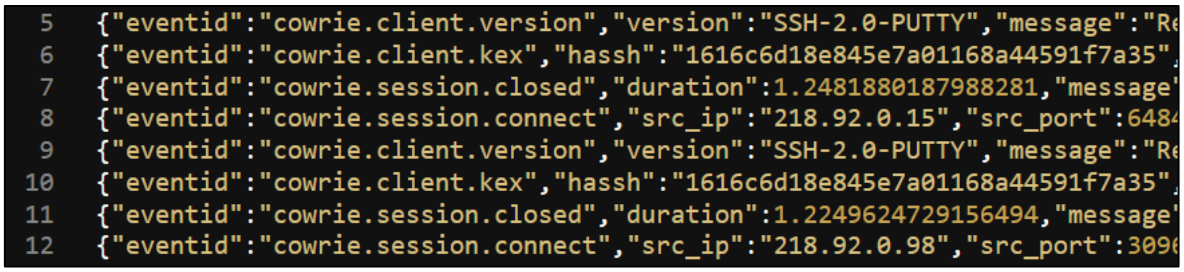

Figura 7. Ejemplo de un JSON de Cowrie

2.3.2 Dionaea

Por su parte, Dionaea es un honeypot de **interacción baja**. Una de sus principales ventajas es la gran cantidad de servicios que logra emular simultáneamente:

· blackhole	· mssql
· epmap	· mysql
· ftp	· pptp
· http	· sip
· memcache	· smb
· mirror	· tftp
· mqtt	· upnp

Además, proporciona un registro muy detallado de la interacción del atacante con el honeypot, incluyendo las vulnerabilidades que estos intentan explotar, los protocolos a los que acceden o el *malware* que utilizan. El honeypot requiere de la apertura de los puertos habituales a los que los servicios mencionados se suelen asociar, o bien de puertos a los que estos redirijan.

Registra su actividad en *bistreams* de cada interacción, pero muy apropiadamente condensa todo en un archivo *sqlite* con el que es fácil interactuar (aunque su tamaño es, como cabe esperar, de cientos de miles de líneas). Dos fragmentos se presentan en la Figura 8 y la Figura 9.

SipSession	1681667093.92997	192.168.1.168	5060	193.107.216.112	5070
SipSession	1681667157.71832	192.168.1.168	5060	193.107.216.112	5070
mssqld	1681667192.05466	192.168.1.168	1433	183.167.237.243	19028
mssqld	1681667193.15733	192.168.1.168	1433	183.167.237.243	19029
SipSession	1681667220.14147	192.168.1.168	5060	193.107.216.112	5071
SipSession	1681667283.61096	192.168.1.168	5060	193.107.216.112	5071
SipSession	1681667348.39886	192.168.1.168	5060	193.107.216.112	5074

Figura 8. Ejemplo de registro de Dionaea (conexiones)

894	31248	administrator	qwerty123456
895	31261	administrator	qazxswedc
896	31274	user	anonymous
897	31287	user	123456
898	31300	user	admin
899	31313	user	root
900	31326	user	password

Figura 9. Ejemplo de registro de Dionaea (logins)

Un ejemplo similar de honeypot es Conpot [17], y nos proponemos estudiar si elegirlo como alternativa es una buena idea. Sin embargo, encontramos algunos motivos por los que Dionaea es mejor opción:

- **Amplitud de protocolos emulados**: Dionaea atrae una gama más amplia de *malware* al ofrecer una mayor cantidad de protocolos que simular.
- **Flexibilidad de despliegue**: Conpot está específicamente pensado para entornos industriales (ICS), mientras que Dionaea no se limita a un solo contexto.
- **Soporte IPv6**: Especialmente en entornos modernos de red, el soporte de IPv6 puede sernos de gran utilidad, y Dionaea es de los pocos honeypots que lo incorpora.

Por todo ello, elegimos Dionaea como honeypot a implementar, una opción sencilla y elegante que no interfiere con Cowrie y puede funcionar de manera paralela.

2.4 Dispositivos IoT y Raspberry Pi

Como mencionábamos en la *Introducción*, cada vez son más los dispositivos IoT presentes en nuestras vidas. Están involucrados en todos los procesos que protagonizamos, tanto sanitarios como informáticos, burocráticos, o meramente cotidianos. Para dotar de realismo a nuestra arquitectura, hemos decidido conformar una microrred de dispositivos conectados a una subred wifi que será monitorizada por la Raspberry Pi. Los dispositivos que hemos elegido son una muestra variada de la tecnología que fácilmente podría encontrarse en un hogar.

2.4.1 Webcams

Las webcams son una brecha de seguridad muy habitual, y su acceso indebido por parte de un tercero puede suponer la completa invasión de nuestra privacidad. Por lo general, todas incluyen *backdoors* y configuraciones predefinidas muy sencillas de superar. Una práctica recomendable es asociar a las cámaras wifi una dirección IP estática para, acto seguido, limitar su acceso al exterior. En nuestro caso, incluimos en la arquitectura uno de los modelos de Dahua [18], proveedor chino de videovigilancia. Este, el **Vandal Proof Wi-Fi Dome**, se observa en la Figura 10.

Figura 10. Dahua Vandal Proof Wi-Fi Dome

2.4.2 Wearables

Los relojes inteligentes, o *smartwatch*, son una alternativa a los relojes de muñeca tradicionales que, además de la hora, ofrecen muchas otras opciones, tales como la toma de pulso, cuenta de pasos, información meteorológica, asistencia deportiva… Para lograr esto, los *wearables* necesitan estar conectados a una red wifi o, más concretamente, a un móvil que, a su vez, se conecte a la red. Esto los hace igual de vulnerables que cualquier otro dispositivo IoT, y deberán poseer un buen diseño de base para no verse comprometidos.

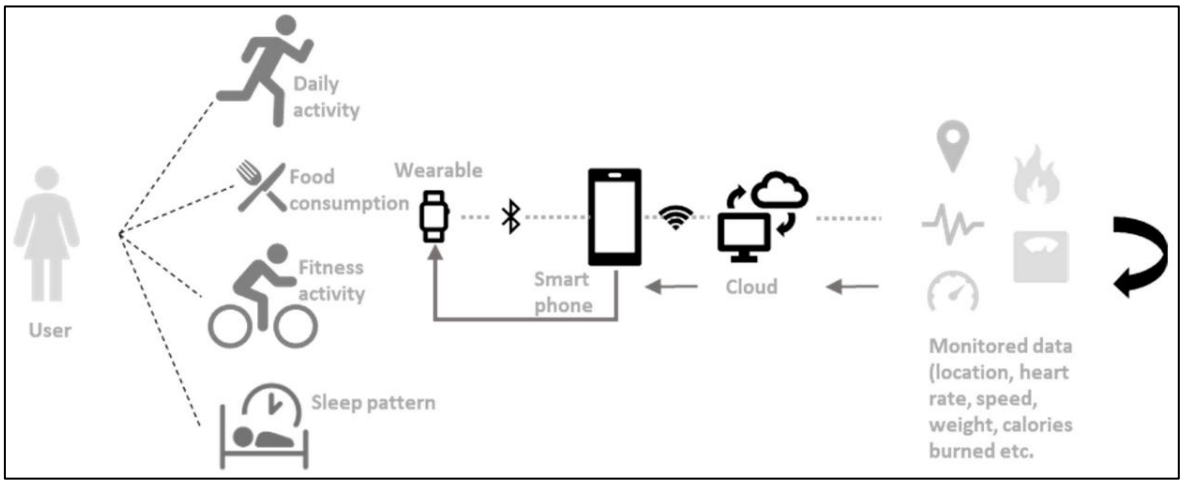

Figura 11. Funcionamiento de un wearable [19]

Contamos con tres relojes inteligentes para nuestro diseño. Estos son el **Fitbit Ace 3**, el **HONOR Band 5** y el **Mi Smart Band 5**. Los tres se parecen mucho en su funcionamiento: es necesario crear una cuenta en su app y tenerla descargada en el móvil al que van asociados. Una vez se conectan, desde el móvil se actualizará periódicamente la información que el reloj registra.

2.4.3 Smartphone

Al igual que los *wearables*, los *smartphones* nacen como una evolución de los teléfonos tradicionales, reuniendo una cantidad incalculable de posibilidades en una pantalla de unas pocas pulgadas. Hay quien los define como el nacimiento de una nueva era en la evolución humana [20], y no es para menos; el ser humano depende por completo de estos dispositivos, y es por ello por lo que es imprescindible que su uso sea seguro y confidencial.

Utilizaremos un Redmi 9 para conectarnos a la red wifi y, a su vez, para conectar a este los relojes inteligentes y la cámara.

2.4.4 Raspberry Pi

Por último, contamos con una Raspberry Pi 3+, Modelo B, que nos servirá como *main hub* del proyecto y que cumplirá el papel de *man-in-the-middle*, analizando el tráfico que pasa por la red wifi por medio de los ya mencionados honeypots Cowrie y Dionaea.

A la hora de decidir qué sistema operativo instalar, nos debatimos entre **Raspberry Pi OS** (antiguamente Raspbian), una distribución de Debian, o **Kali**, derivada también de Debian. Kali está pensada específicamente para la seguridad informática y, por tanto, ofrece ventajas muy atractivas, como su análisis exhaustivo del tráfico o el paquete de aplicaciones preinstaladas que incluye (Wireshark, Metasploit, Aircrack…). Es un OS muy habitual entre los *pen-testers*.

Sin embargo, Raspberry Pi OS es menos complejo y desafiante. Para el propósito que a este documento atañe, las herramientas de Kali son demasiado avanzadas, y dada la facilidad de uso del primero y, en especial, su menor consumo de recursos, nos terminamos decantando por elegirlo como sistema operativo.

2.5 Amenazas más frecuentes

Podemos resumir algunas de las amenazas que más se repiten entre los ataques a dispositivos IoT vulnerables. Esto nos ayudará a desarrollar un protocolo de anticipación y actuación, y también nos servirá para buscar entre los registros de los honeypots indicios de este *modus operandi*.

2.5.1 Ataques de fuerza bruta

Los ataques de fuerza bruta contra el protocolo SSH son una amenaza comúnmente observada. Este tipo de ataque intenta acceder a un sistema probando todas las combinaciones posibles de credenciales hasta que se encuentre una que funcione. Se discutirá la frecuencia de estos ataques, así como los patrones observados, como los nombres de usuario y contraseñas más comúnmente probados.

2.5.2 Malware

Algunos honeypots capturan y analizan las instancias de *malware* que los atacantes intentan instalar en el sistema. Ejemplos de este *malware* son los *botnets*, *ransomware*, *spyware*, o los troyanos.

2.5.3 Reconocimiento y escaneo

Los atacantes a menudo realizan actividades de reconocimiento y escaneo para identificar posibles objetivos y determinar qué servicios están corriendo y qué vulnerabilidades podrían estar presentes. Esta es la vía por la que los atacantes encuentran nuestro honeypot para proceder a su explotación. Suele estar completamente automatizada.

2.5.4 Ataques DDoS

Algunos atacantes pueden intentar inundar la red con tráfico para causar una denegación de servicio, ralentizando o incluso anulando la red por completo [21].

2.5.5 Amenazas específicas de IoT

Dada la presencia de dispositivos IoT en la red, es importante discutir las amenazas que son específicas para estos dispositivos, tales como intentos de acceso a través de credenciales predeterminadas o conocidas vulnerabilidades de firmware.

Capítulo 3. Diseño e implementación

3.1 Arquitectura de la red

La red está conformada por una Raspberry Pi 3+ Modelo B, eje de la comunicación; una serie de dispositivos IoT (enumerados en la sección *Dispositivos IoT*); y el router, que nos da acceso a internet.

Con el fin de conectar los dispositivos, incluido el *smartphone*, a la red del router, optamos por crear una subred reservada para el proyecto. Así, el tráfico general del hogar no se verá afectado por el honeypot. Existe un servicio de software libre llamado **dnsmasq** que cumple con lo deseado. Se trata de un redireccionador de sistema de nombres de dominio (DNS) ligero y fácil de configurar, pensado específicamente para redes de pequeña escala [22].

Combinaremos este servicio con el de **HostAPD**, que es capaz de hacer funcionar una tarjeta inalámbrica como un punto de acceso wifi [23].

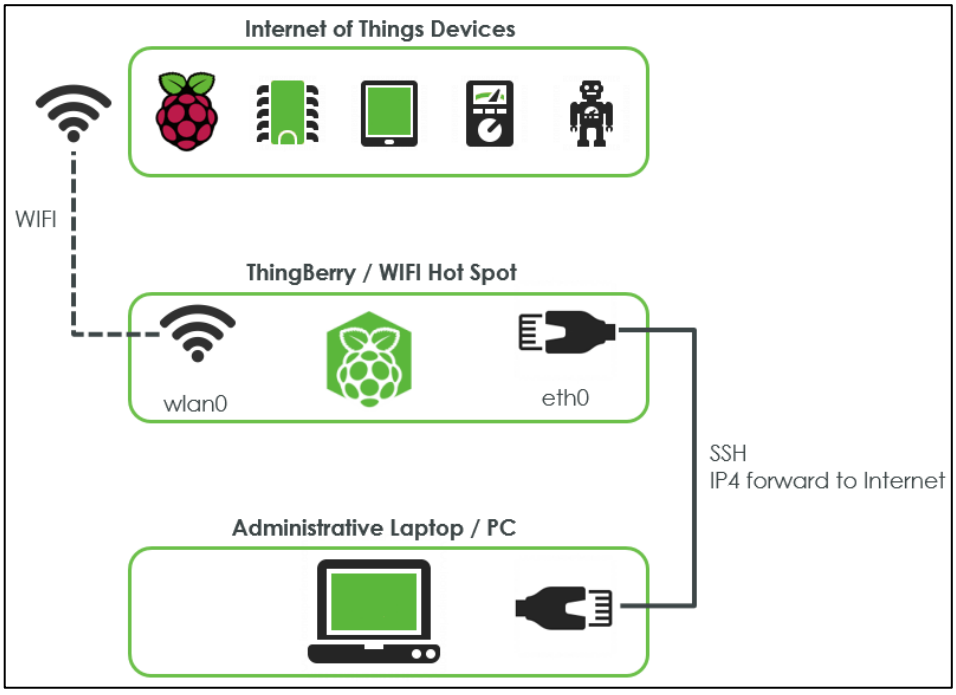

Figura 12. Raspberry Pi como wifi AP [24]

3.2 Implementación del diseño

3.2.1 Configuración de los dispositivos

Empezamos configurando los dispositivos IoT que se conectarán a la red. En primer lugar, los *wearables*, que asociamos al Redmi 9 por medio de sus aplicaciones.

Para ello, creamos una cuenta Gmail que servirá como identidad ficticia en todo nuestro ejercicio: sugarhoneyiceandpot@gmail.com; Richard Honey. A continuación, creamos también cuentas en las apps de Fitbit y Zepp Life (de la Mi Smart Band 5). Una vez asociadas a los relojes, confirmamos que la información se sincroniza con éxito.

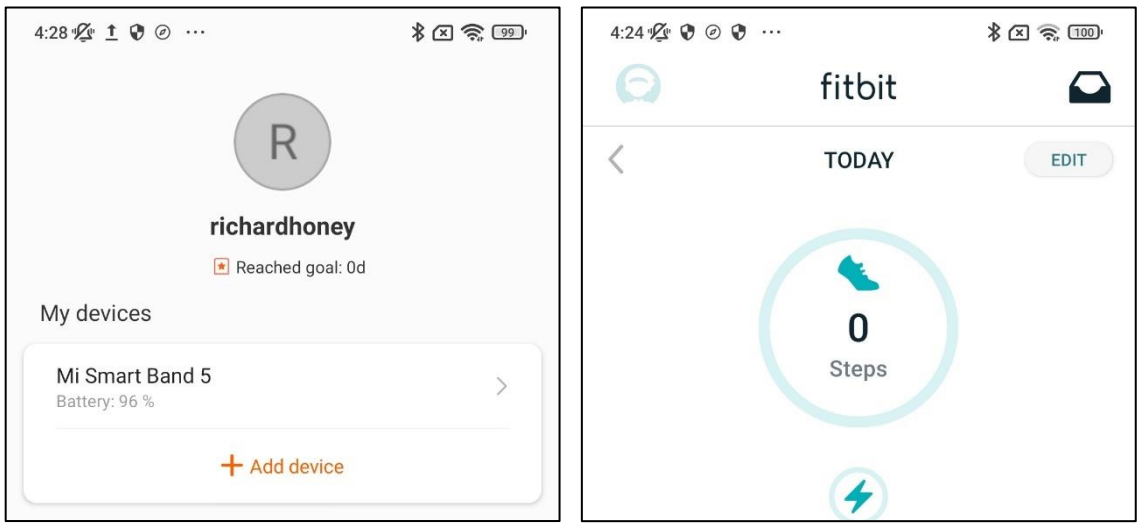

Figura 13. Capturas de las apps de los wearables

En segundo lugar, nos encargamos de la webcam. Esta debe conectarse por primera vez a través de su puerto ethernet. Una vez hecho esto, es sencillo acceder a su configuración: tan solo tenemos que buscar su dirección IP en la barra del navegador.

Para saber cuál es la dirección IP privada asociada a la cámara, nos valdremos del comando **nmap**, que resume los dispositivos dentro de un rango, como se muestra en la Figura 14.

```
C:\Users\Ernes>nmap -sn 192.168.1.1-254
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-16 00:38 Hora de verano romance
Nmap scan report for 192.168.1.1
Host is up (0.019s latency).
MAC Address: D4:7B:B0:A4:E4:A5 (Askey Computer)
Nmap scan report for 192.168.1.108
Host is up (0.013s latency).
MAC Address: A0:BD:1D:04:F1:C9 (Zhejiang Dahua Technology)
Nmap scan report for 192.168.1.139
Host is up.
MAC Address: 8C:C6:81:8B:BC:5D (Intel Corporate)
Nmap scan report for 192.168.1.147
Host is up (0.46s latency).
MAC Address: 58:B1:0F:C8:0A:76 (Samsung Electronics)
Nmap scan report for 192.168.1.170
Host is up (0.15s latency).
MAC Address: 4E:5F:7A:89:95:3B (Unknown)
Nmap scan report for 192.168.1.145
Host is up.
Nmap done: 254 IP addresses (6 hosts up) scanned in 6.77 seconds
```

Figura 14. Direcciones IP de los dispositivos

Una vez sabida, basta con conectarla a la subred wifi, cuyo establecimiento describiremos en el siguiente epígrafe. Es importante especificar que el *smartphone* estará conectado también a esta subred.

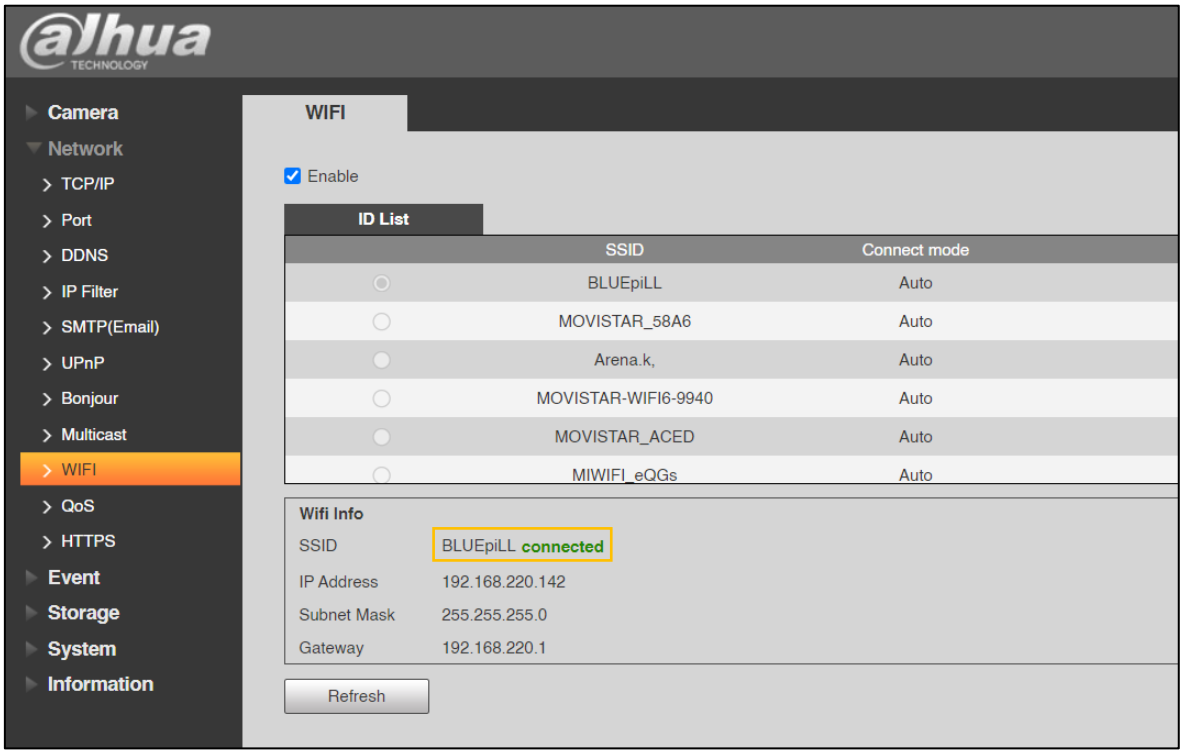

Figura 15. Webcam conectada a la subred wifi

3.2.2 Configuración de la subred wifi

Como definimos en la *Arquitectura de la red*, utilizamos HostAPD y dnsmasq para establecer una subred wifi. Elegimos como nombre **BLUEpiLL**, y le asociamos el rango de direcciones privadas **192.168.220.1-254**, que se asignarán a los dispositivos. Nos aseguramos de que la Raspberry no varíe su propia dirección IP asignándole una dirección estática, 192.168.1.168, desde el router.

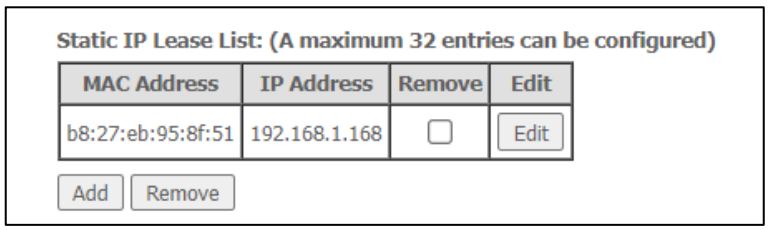

Figura 16. Dirección IP estática de la Raspberry Pi

Una vez configurada la subred, volvemos a usar **nmap** para confirmar que los dispositivos se encuentran en ella.

```
C:\Users\Ernes>nmap -sn 192.168.220.1-254
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-16 12:19 Hora de verano romance
Nmap scan report for 192.168.220.1
Host is up.
MAC Address: B8:27:EB:C0:DA:04 (Raspberry Pi Foundation)
Nmap scan report for 192.168.220.128
Host is up (0.24s latency).
MAC Address: A4:55:90:F8:DF:0B (Xiaomi Communications)
Nmap scan report for 5E0894CPAGF919A (192.168.220.142)
Host is up (0.026s latency).
MAC Address: A0:BD:1D:04:F2:F5 (Zhejiang Dahua Technology)
Nmap scan report for 192.168.220.126
Host is up.
Nmap done: 254 IP addresses (4 hosts up) scanned in 11.44 seconds
```

Figura 17. Direcciones IP de los dispositivos IoT

3.2.3 Implementación de los honeypots

Procedemos a implementar los honeypots Cowrie y Dionaea en la Raspberry. Para ello, seguimos los pasos descritos en sus respectivas documentaciones oficiales, disponibles en GitHub.

i) Cowrie

En un primer momento, parece un proceso sencillo. De hecho, establecer Cowrie no presenta mucha complicación, y en seguida está funcionando. Cabe destacar que, siguiendo el consejo del autor, iniciamos Cowrie desde un entorno virtual, aunque creemos que esto no sería necesario en este caso concreto, pues la red que hemos instaurado no tiene ningún tipo de valor.

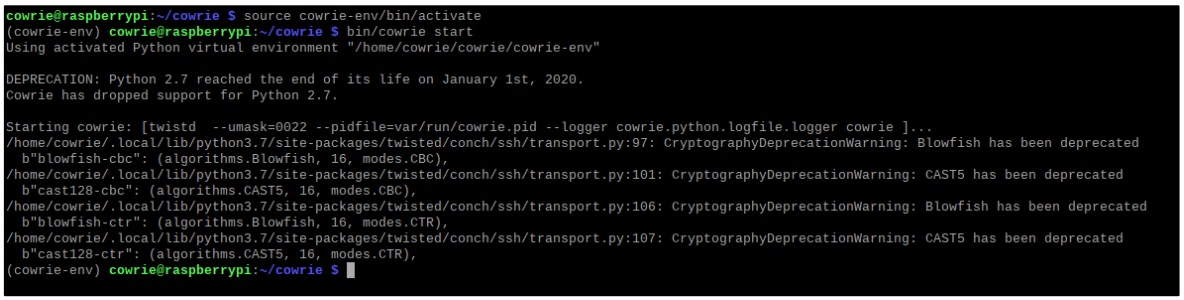

Figura 18. Cowrie funcionando

También es importante indicar que en el archivo de configuración del honeypot, **cowrie.cfg**, será donde establezcamos qué protocolos emular (SSH y Telnet) y a qué puertos asociarlos (22 y 23, para un mayor realismo, porque son los que por defecto se asocian).

Para asignar servicios a puertos por debajo de 1024, por seguridad, es necesario tener privilegios de *root*, ya que un mal uso de estos puede provocar brechas de seguridad importantes. Haciendo caso de la aportación de un usuario, @joeforker [26], ejecutamos el comando:

```
sudo setcap cap_net_bind_service=ep some-binary
```

Como él mismo indica, CAP_NET_BIND_SERVICE corresponde a la capacidad de *root* de asignar puertos <= 1024. De esta forma, ya podemos configurar Cowrie.

Otra alternativa habría sido utilizar otros puertos menos cruciales y redirigir el tráfico. Un comando ejemplo sería:

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

Como curiosidad, existe un modo de configuración por el cual el honeypot permite el acceso tras un número aleatorio de intentos. Dado que diferentes atacantes van a probar distintas combinaciones de usuarios y contraseñas, nos pareció buena idea especificar esta línea.

```
#  auth_class_parameters: <min try>, <max try>, <maxcache>
 auth_class = AuthRandom
 auth_class_parameters = 2, 5, 10
```

ii) Dionaea

Con Dionaea nos sucede todo lo contrario. Por algún motivo, la forma en la que su web indica que hay que instalar el honeypot no termina de funcionar, y al intentar iniciarlo no arranca ninguno de los servicios que debería.

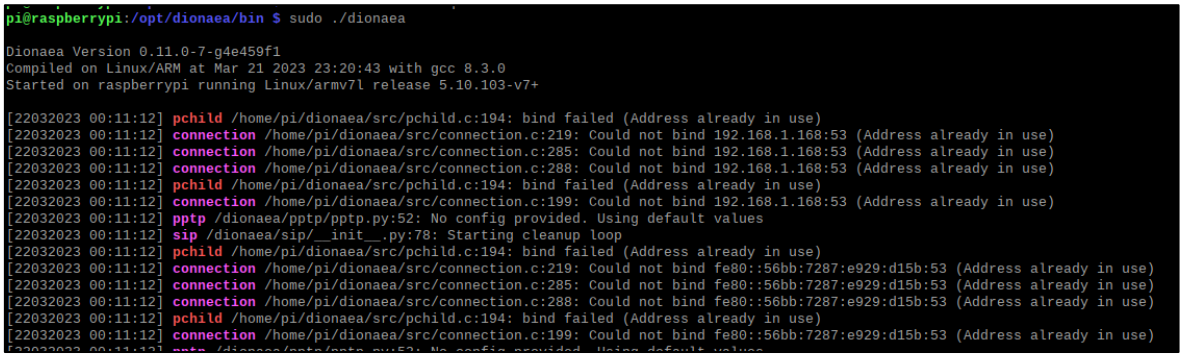

Figura 19. Dionaea presentando errores

Esta será una de las mayores complicaciones del proyecto, y nos llevará un tiempo considerable solucionarla. Tras probar muchas de las propuestas de usuarios de internet, damos con el consejo de @ahezza [25], que propone modificar el fichero *dionaea.service*:

```
nano /etc/systemd/system/dionaea.service
```

E incluir las siguientes líneas:

```
[Unit]
Description=Dionaea
[Service]
WorkingDirectory = /opt/dionaea
ExecStart = /opt/dionaea/bin/dionaea
User = pi
Group = pi
Restart = always
[Install]
WantedBy = multi-user.target
```

El usuario debe tener permisos suficientes para ejecutar el honeypot, por lo que inferimos que el problema debía tener que ver con los permisos.

Por último, cambiamos los permisos del propio fichero:

```
chmod 755 /etc/systemd/system/dionaea.service
```

Una vez seguidos los pasos, se nos ofrece una nueva línea de comando que ejecuta el honeypot y que, para nuestra sorpresa, funciona con éxito:

```
sudo systemctl start dionaea.service
```

Además, también disponemos de otro comando que devuelve el estado del servicio:

```
sudo systemctl status dionaea.service
```

```
pi@raspberrypi:~ $ sudo systemctl start dionaea.service
pi@raspberrypi:~ $ sudo systemctl status dionaea.service
● dionaea.service - Dionaea
   Loaded: loaded (/etc/systemd/system/dionaea.service; disabled; vendor preset:
   Active: active (running) since Fri 2023-04-14 21:05:32 CEST; 1s ago
 Main PID: 1037 (dionaea)
    Tasks: 2 (limit: 2059)
   CGroup: /system.slice/dionaea.service
           ├─1037 /opt/dionaea/bin/dionaea
           └─1038 /opt/dionaea/bin/dionaea

Apr 14 21:05:32 raspberrypi systemd[1]: Started Dionaea.
```

Figura 20. Dionaea funcionando

3.2.4 Configuración del router

Por último, es necesario que configuremos los puertos del router para abrirlos al exterior y que puedan ser detectados por otros ordenadores desde otras redes (ahora mismo solo son accesibles a nivel local).

Accedemos introduciendo en la barra del navegador la dirección 192.168.1.1, que suele ser la que tiene por defecto. Una vez dentro, basta con abrir los puertos de los servicios de los honeypots para la dirección IP de la Raspberry que, como recordamos, configuramos como estática.

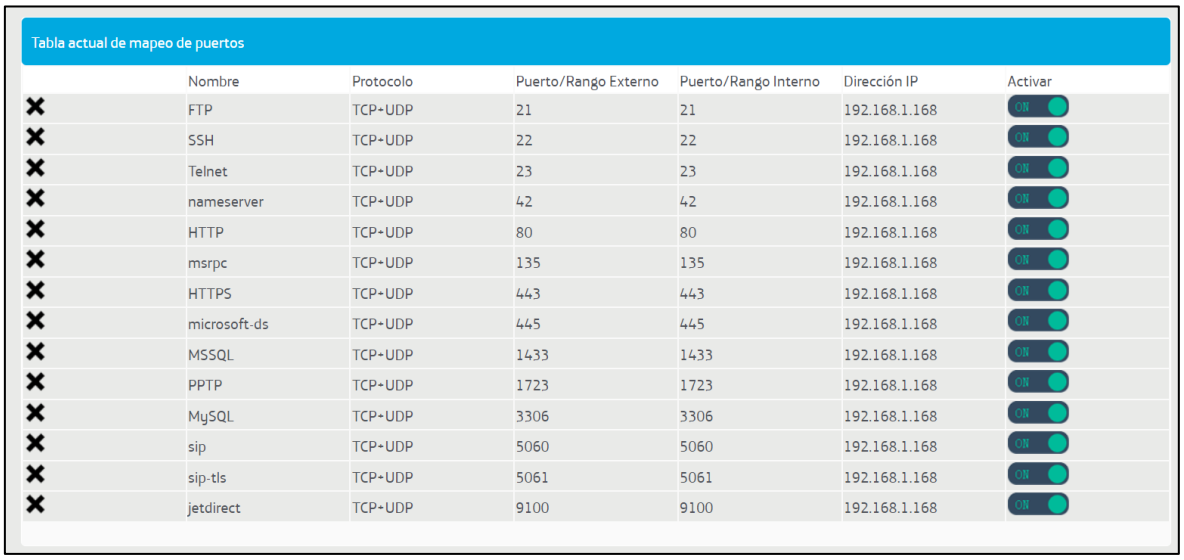

Tabla actual de mapeo de puertos

	Nombre	Protocolo	Puerto/Rango Externo	Puerto/Rango Interno	Dirección IP	Activar
✖	FTP	TCP+UDP	21	21	192.168.1.168	ON
✖	SSH	TCP+UDP	22	22	192.168.1.168	ON
✖	Telnet	TCP+UDP	23	23	192.168.1.168	ON
✖	nameserver	TCP+UDP	42	42	192.168.1.168	ON
✖	HTTP	TCP+UDP	80	80	192.168.1.168	ON
✖	msrpc	TCP+UDP	135	135	192.168.1.168	ON
✖	HTTPS	TCP+UDP	443	443	192.168.1.168	ON
✖	microsoft-ds	TCP+UDP	445	445	192.168.1.168	ON
✖	MSSQL	TCP+UDP	1433	1433	192.168.1.168	ON
✖	PPTP	TCP+UDP	1723	1723	192.168.1.168	ON
✖	MySQL	TCP+UDP	3306	3306	192.168.1.168	ON
✖	sip	TCP+UDP	5060	5060	192.168.1.168	ON
✖	sip-tls	TCP+UDP	5061	5061	192.168.1.168	ON
✖	jetdirect	TCP+UDP	9100	9100	192.168.1.168	ON

Figura 21. Puertos abiertos en el router

3.3 Comprobación del funcionamiento de la red

Tras configurar nuestro diseño, solo falta confirmar que este funciona como debe. Primero, confirmamos con **nmap** que todos los puertos están, efectivamente, abiertos:

```
pi@raspberrypi:~ $ nmap 192.168.1.168
Starting Nmap 7.70 ( https://nmap.org ) at 2023-04-14 21:08 CEST
Nmap scan report for 192.168.1.168
Host is up (0.00072s latency).
Not shown: 985 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
42/tcp   open  nameserver
53/tcp   open  domain
80/tcp   open  http
135/tcp  open  msrpc
443/tcp  open  https
445/tcp  open  microsoft-ds
1433/tcp open  ms-sql-s
1723/tcp open  pptp
3306/tcp open  mysql
5060/tcp open  sip
5061/tcp open  sip-tls
9100/tcp open  jetdirect

Nmap done: 1 IP address (1 host up) scanned in 0.37 seconds
pi@raspberrypi:~ $
```

Figura 22. Puertos de los servicios

En teoría, los puertos están abiertos y emulando los servicios que deseamos. Hacemos un par de comprobaciones rápidas.

Primero, en local, un SSH desde el ordenador a la Raspberry:

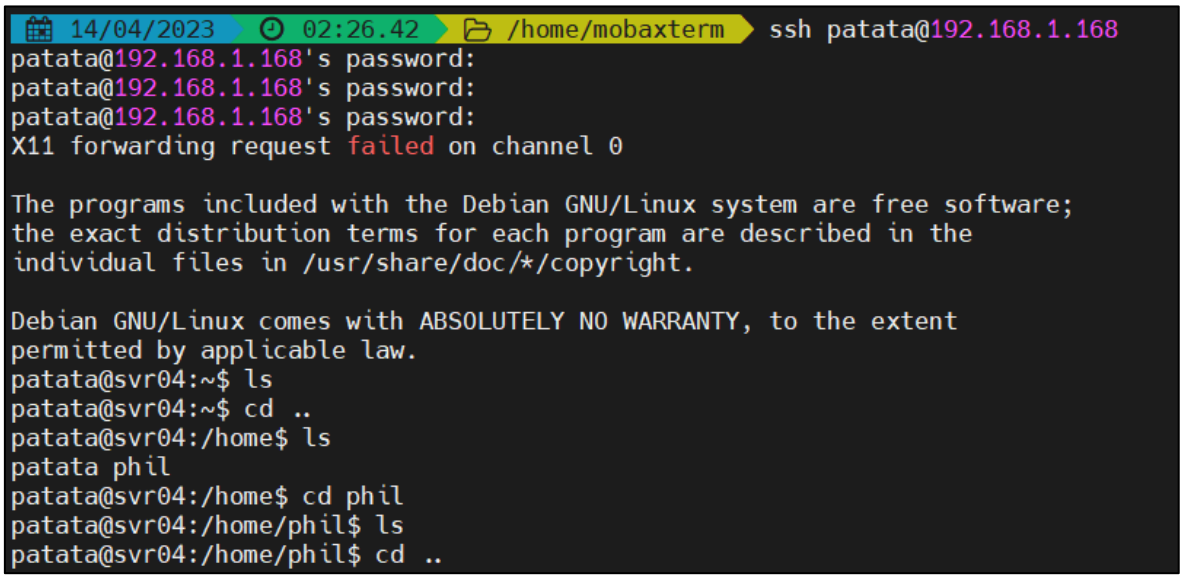

Figura 23. SSH local (emisor)

Que, como cabía esperar, se registra en el *log* del honeypot.

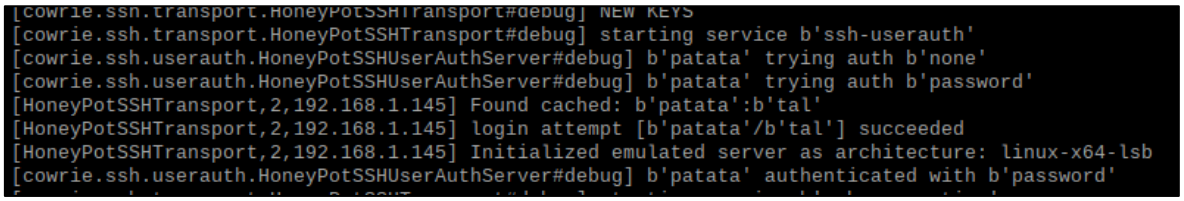

Figura 24. Log del ataque SSH local

Y probamos con un Telnet al puerto 23, que también se realiza.

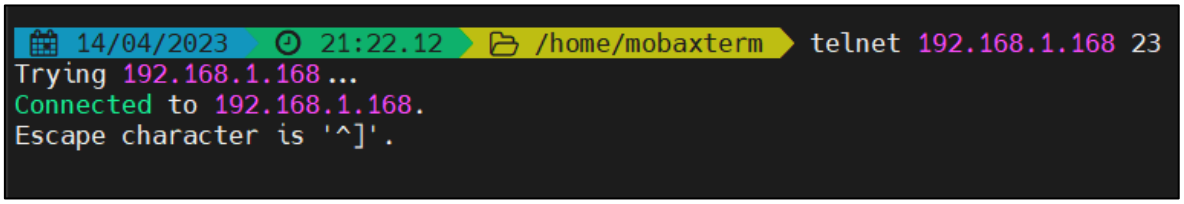

Figura 25. Telnet local (emisor)

Pero no podemos afirmar que los honeypots funcionan hasta que no lo confirmemos desde una red externa. Si funciona en local pero no pueden encontrarnos desde fuera, el propósito del honeypot se pierde.

Por tanto, conectamos el ordenador a una red pública, en este caso a la de nuestros datos móviles. Lo confirmamos comprobando la IP que se nos asigna:

Figura 26. IP pública (emisor)

De igual forma, revisamos la IP que se le ha asignado a la Raspberry.

Figura 27. IP pública (receptor)

Esta será la dirección a la que atacarán los usuarios de internet (puede cambiar en algún momento y, de hecho, lo hará un par de veces). Como vemos, es completamente distinta a la del emisor del SSH, así que, si el protocolo se establece, habremos confirmado que los honeypots están bien establecidos. Probamos:

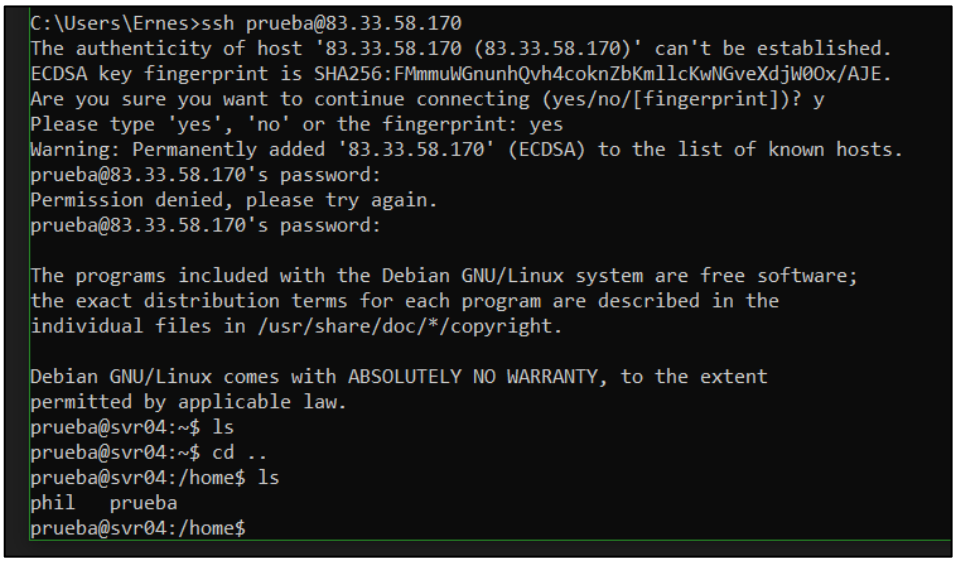

Figura 28. SSH entre redes

Y apreciamos cómo el SSH se ha establecido con éxito, probando que los honeypots ya están listos para recibir ataques.

Capítulo 4. Medición y análisis

4.1 Primeras incidencias

Procedemos a esperar un primer ataque que nos asegure la realidad de que, en efecto, los honeypots funcionan, y no tarda en llegar.

Vemos que se registran intentos de login a través de SSH por parte de una IP, 101.34.138.117. El atacante prueba con combinaciones habituales, como “admin”, “123456” o “ubuntu”.

```
[HoneyPotSSHTransport,54,101.34.138.117] login return, expect: [b'admin'/b'123456']
[HoneyPotSSHTransport,54,101.34.138.117] login attempt [b'ubuntu'/b'ubuntu'] failed
[cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'ubuntu' failed auth b'password'
[cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] unauthorized login: ()
[cowrie.ssh.transport.HoneyPotSSHTransport#info] connection lost
[HoneyPotSSHTransport,54,101.34.138.117] Connection lost after 4 seconds
[cowrie.ssh.factory.CowrieSSHFactory] New connection: 101.34.138.117:42566 (192.168.1.168:22) [session: 69e1be70474a]
[HoneyPotSSHTransport,55,101.34.138.117] Remote SSH version: SSH-2.0-Go
[HoneyPotSSHTransport,55,101.34.138.117] SSH client hassh fingerprint: 4e066189c3bbeec38c99b1855113733a
[cowrie.ssh.transport.HoneyPotSSHTransport#debug] kex alg=b'curve25519-sha256' key alg=b'ecdsa-sha2-nistp256'
[cowrie.ssh.transport.HoneyPotSSHTransport#debug] outgoing: b'aes128-ctr' b'hmac-sha2-256' b'none'
[cowrie.ssh.transport.HoneyPotSSHTransport#debug] incoming: b'aes128-ctr' b'hmac-sha2-256' b'none'
[cowrie.ssh.transport.HoneyPotSSHTransport#debug] NEW KEYS
[cowrie.ssh.transport.HoneyPotSSHTransport#debug] starting service b'ssh-userauth'
[cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'ubuntu' trying auth b'none'
[cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'ubuntu' trying auth b'password'
[HoneyPotSSHTransport,55,101.34.138.117] login attempt: 41
[HoneyPotSSHTransport,55,101.34.138.117] login return, expect: [b'admin'/b'123456']
[HoneyPotSSHTransport,55,101.34.138.117] login attempt [b'ubuntu'/b'123456'] failed
[cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'ubuntu' failed auth b'password'
[cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] unauthorized login: ()
```

Figura 29. Primer SSH externo

Si geolocalizamos la dirección IP veremos que corresponde a un equipo situado (en teoría) en China. Bastante impresionante.

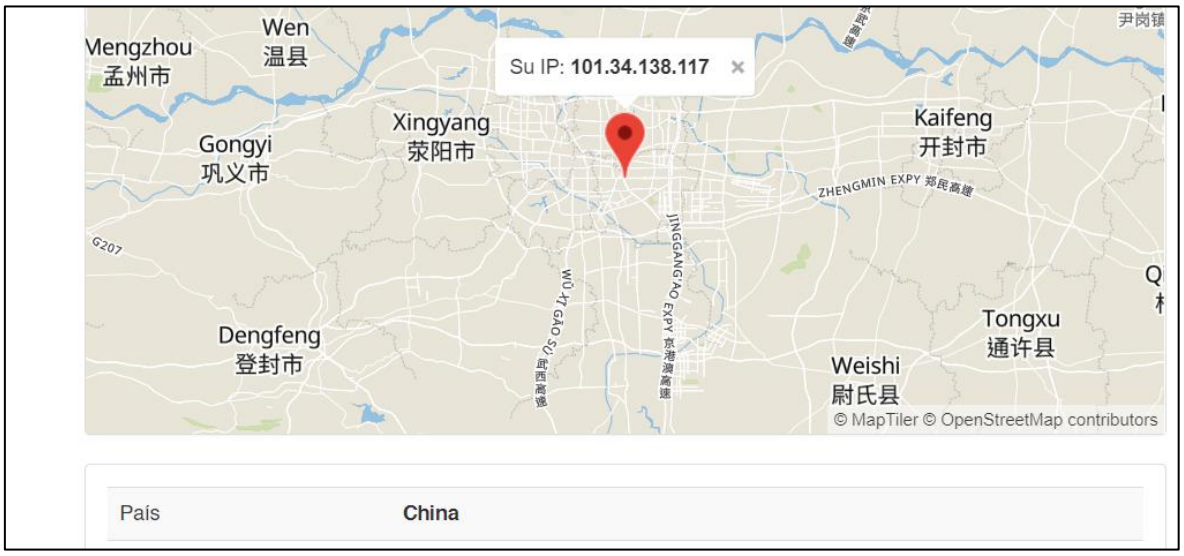

Figura 30. Ubicación de la primera IP

De nuevo, se da otro ataque SSH. En esta ocasión, la dirección es 207.229.167.36.

```
[cowrie.ssh.factory.CowrieSSHFactory] New connection: 207.229.167.36:53640 (192.168.1.168:22) [session: 2c54b1f1ade6]
[HoneyPotSSHTransport,61,207.229.167.36] Remote SSH version: SSH-2.0-HELLOWORLD
[cowrie.ssh.transport.HoneyPotSSHTransport#info] connection lost
[HoneyPotSSHTransport,61,207.229.167.36] Connection lost after 20 seconds
ypi:~/cowrie/var/log/cowrie $
```

Figura 31. Segundo SSH externo

Si tratamos de localizar esta, veremos que corresponde a EEUU.

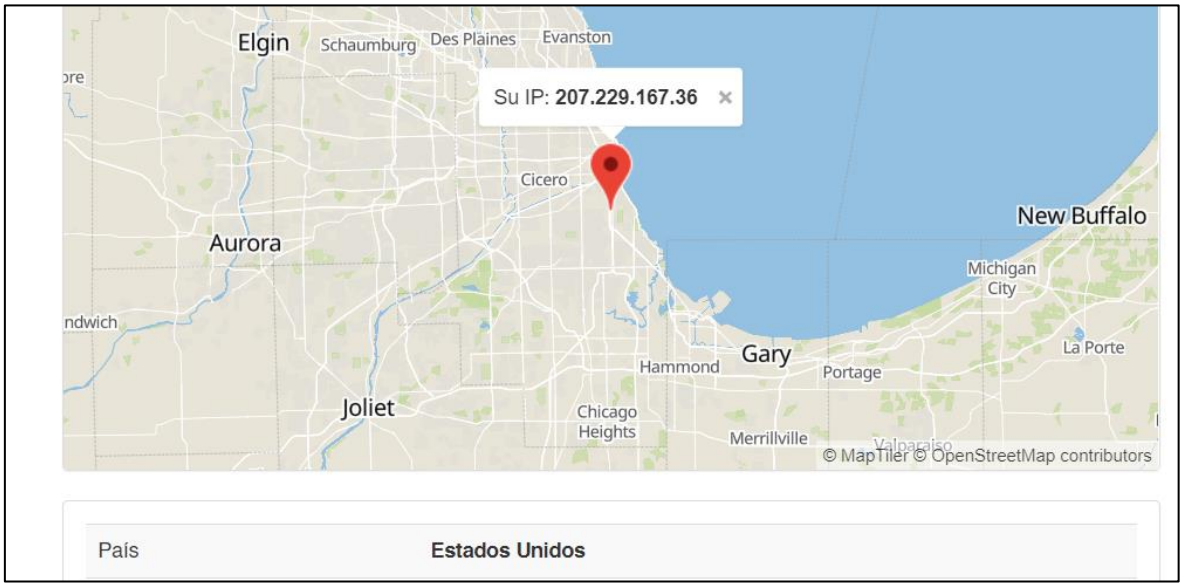

Figura 32. Ubicación de la segunda IP

De esta forma, demostramos empíricamente la efectividad de nuestro diseño.

4.2 Mediciones finales

Tras haber estado funcionando durante 45 días, desde el 15 de abril hasta el 30 de mayo, recopilamos datos suficientes como para elaborar nuestra tesis e inferir conclusiones.

4.2.1 Dionaea

Por una parte, el honeypot Dionaea nos proporciona un archivo **SQLite** [27] de 679MB, tamaño más que considerable para una base de datos que recoge nuestros ataques. SQLite es un motor de base de datos SQL incorporado en el lenguaje de programación C. Es rápido, eficiente y muy apropiado para su cometido. Además, es compatible con muchos programas de análisis de datos, y desde su versión 3 es capaz de alojar hasta 2TB de información. También permite el acceso en paralelo a la lectura, lo que lo hace aún más rápido.

En un primer momento, teníamos la intención de crear un programa en Python que se encargase de minar los datos de la base SQLite, quedándose únicamente con aquellos que nos fuesen relevantes para que su procesado posterior fuese más sencillo. Python incluye soporte para este lenguaje, y el código era relativamente fácil de programar, pero nos vimos limitados por la capacidad de la CPU. Algunas de las tablas de la base de datos sobrepasaban el millón de registros, como se puede observar en la Figura 33.

19	19	accept	tcp	Blackhole
20	20	accept	tcp	Blackhole
21	21	accept	tcp	httpd
22	22	accept	tcp	mssqld
23	23	accept	tcp	httpd
24	24	accept	tcp	httpd

1 - 24 de 1326350

Figura 33. Fragmento de la tabla connections del SQLite

Lo ideal habría sido crear un código que, a medida que detectaba nuevos registros, los hubiese ido procesando en tiempo real. No obstante, esto no nos limita a la hora de interpretar

los datos. Existe una solución sencilla por la cual se puede dotar a Tableau, el programa de visualización de datos que hemos elegido [28], de la capacidad de interpretar bases de datos SQLite.

Escogemos Tableau debido a su interfaz intuitiva y sencilla, su amplia gama de visualizaciones disponibles y la versatilidad y potencia de sus *dashboards* interactivos. Basta con descargar un *driver* que soporte el formato [29] y seleccionarlo al leer los datos.

Figura 34. Selección del controlador de SQLite3

De esta manera, tendremos los datos listos para trabajar con ellos como nos interese.

Sin embargo, nos conviene revisar primero la estructura que presentan. Un estudio preliminar puede ahorrarnos mucho tiempo cuando pasemos a la visualización.

Existe un programa sencillo, DB Browser for SQLite [30], que lee una base de datos SQLite y le ofrece al usuario una interfaz gráfica con la que es fácil trabajar. Lo instalamos e importamos la base de datos de Dionaea.

Lo primero en lo que nos fijamos es en la gran cantidad de tablas que hay. Aunque pueda parecer abrumador, lo cierto es que este reparto nos facilita mucho el trabajo.

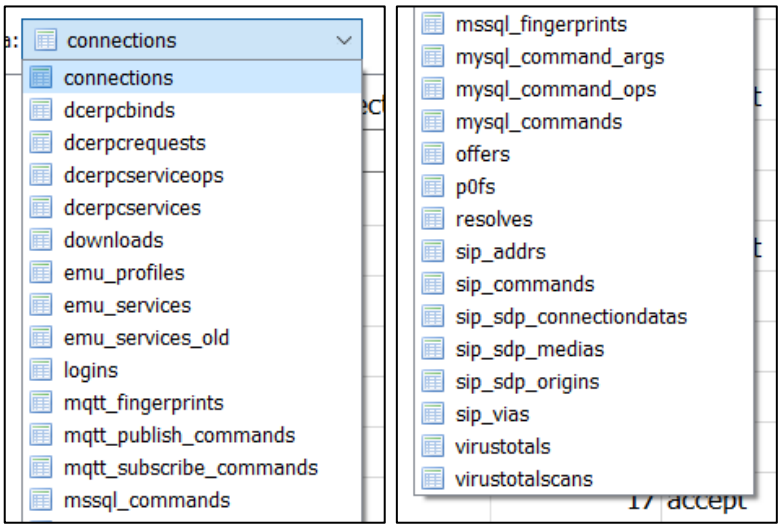

Figura 35. Tablas de la BBDD de Dionaea

Debido al alcance del proyecto, nos centraremos, únicamente, en dos de las tablas: *connections* y *logins*. Podemos prescindir del resto, que tienen que ver con la interacción del atacante con la plataforma, pero escapan a nuestro enfoque.

La tabla *connections* ofrece 1326350 registros, y tiene como campos de interés:

Tabla I. Cabeceras de connections

Cabecera	Ofrece
connection_type	Tipo de conexión: listen, connect o accept, dependiendo de la comunicación
connection_transport	Tipo de transporte: TCP o UDP
connection_protocol	Protocolo explotado: ftpd, Blackhole, SipSession, smbd, upnpd…
connection_timestamp	Día y hora del evento (en formato Unix epoch)
local_host	IP local
local_port	Puerto local
remote_host	IP remota
remote_port	Puerto remoto

Así, en Tableau sintetizaremos estos campos en algunos gráficos exploratorios y/o combinatorios.

Por su parte, *logins* presenta 13197 entradas, y sus cabeceras son:

Tabla II. Cabeceras de logins

Cabecera	Ofrece
login_username	Nombre de usuario
login_password	Contraseña

Esta tabla es más sencilla, y se asocia con los intentos de *login* que los atacantes han intentado. Es muy curiosa de analizar, y presenta combinaciones bastante habituales de cuentas *default*, por lo que debemos recalcar que es importante configurar credenciales seguras en todos los dispositivos que utilicemos. Más adelante interpretaremos con atención los datos.

4.2.2 Cowrie

El honeypot Cowrie registra sus datos de otra manera. Como mencionábamos en la *Metodología de la investigación*, cada día condensa en un archivo JSON sus eventos, y al final de los 45 días había acumulado 333MB de información. Aunque no parezca mucho, tenemos que recordar que se trata únicamente de líneas de texto, y se dan, de media, 22MB en registros de ataques diarios.

Al igual que en el caso de Dionaea, la potencia de nuestro equipo nos limita el procesamiento de los datos. Nuestra primera propuesta fue crear un pequeño programa en RStudio para manipular la información:

```
# Cargamos los paquetes necesarios
library(jsonlite)
library(dplyr)
library(httr)
library(ipaddress)

# Leemos el archivo como JSON Lines
data <- stream_in(file("cowrie2023-05-22.json"))

# Convertimos a un data.frame para poder trabajar con él en R
```

```
df <- as.data.frame(data)

# Eliminamos las columnas no deseadas
df <- df %>% select(-compCS, -encCS, -kexAlgs, -keyAlgs, -macCS, -langCS,
-message,-sensor, -session, -hassh, -hasshAlgorithms, -fingerprint, -
key,-type, -data, -id, -arch, -ttylog, -size, -shasum, -duplicate,-
destfile)

# Convertimos la columna timestamp en formato de fecha legible por
Tableau
df$timestamp <- as.Date(substr(df$timestamp, 1, 10))

# Cargamos los archivos CSV de las BBDD de las IPs
blocks <- read.csv("GeoLite2-ASN-Blocks-IPv4.csv", stringsAsFactors =
FALSE)
country <- read.csv("GeoLite2-Country-Blocks-IPv4.csv", stringsAsFactors
= FALSE)
country_locations <- read.csv("GeoLite2-Country-Locations-en.csv",
stringsAsFactors = FALSE)

# Unimos los dataframes de blocks y country
blocks <- merge(blocks, country, by = "network")

# Unimos los dataframes de blocks y country_locations
blocks <- merge(blocks, country_locations, by.x = "geoname_id", by.y =
"geoname_id")

# Añadimos columnas ASN y Country al dataframe df, inicialmente con NA
df$ASN <- NA
df$Country <- NA

# Función para comprobar si una dirección IP está en una subred
ip_in_subnet <- function(ip, subnet){
  ip_net <- ip_network(subnet)
  return(ip_address(ip) %in% ip_net)
}

top_ips <- df %>% count(src_ip) %>% top_n(1) %>% pull(src_ip)

# Para cada una de las 20 IPs más comunes, comprobamos si la IP se
encuentra en algún rango
for(ip in top_ips){
  for(j in seq_len(nrow(asn_blocks))){
    if(ip_in_subnet(ip, asn_blocks$network[j])){
      df$ASN[df$src_ip == ip] <- asn_blocks$autonomous_system_number[j]
      break
    }
  }
  for(j in seq_len(nrow(country_blocks))){
    if(ip_in_subnet(ip, country_blocks$network[j])){
      geoname_id <- country_blocks$geoname_id[j]
      df$Country[df$src_ip == ip] <-
country_locations$country_iso_code[country_locations$geoname_id ==
geoname_id]
      break
    }
```

```
    }
}

# Escribimos el DataFrame en un archivo CSV
write.csv(df, file = "datos_cow.csv", row.names = FALSE, col.names =
TRUE)
```

El código empieza leyendo un archivo JSON (más adelante, iteraríamos para leer todos juntos). Tras eliminar las columnas que no nos interesan, y adaptar la de la fecha, damos con un *dataframe* que luce así:

	eventid	duration	timestamp	src_ip	src_port	dst_ip	dst_port
1	cowrie.session.closed	1.359705448	2023-05-22	218.92.0.95	NA	NA	NA
2	cowrie.session.connect	NA	2023-05-22	64.227.177.54	61000	192.168.1.168	22
3	cowrie.session.closed	0.143246889	2023-05-22	64.227.177.54	NA	NA	NA
4	cowrie.session.connect	NA	2023-05-22	195.3.147.52	31658	192.168.1.168	22
5	cowrie.client.version	NA	2023-05-22	195.3.147.52	NA	NA	NA
6	cowrie.session.closed	0.009408236	2023-05-22	195.3.147.52	NA	NA	NA
7	cowrie.session.connect	NA	2023-05-22	195.3.147.52	26253	192.168.1.168	22
8	cowrie.client.version	NA	2023-05-22	195.3.147.52	NA	NA	NA
9	cowrie.client.kex	NA	2023-05-22	195.3.147.52	NA	NA	NA
10	cowrie.login.failed	NA	2023-05-22	195.3.147.52	NA	NA	NA
11	cowrie.session.closed	1.984613657	2023-05-22	195.3.147.52	NA	NA	NA
12	cowrie.session.connect	NA	2023-05-22	174.138.61.44	33894	192.168.1.168	22

protocol	version	username	password	input	outfile	url
NA	NA	NA	NA	NA	NA	NA
ssh	NA	NA	NA	NA	NA	NA
NA	NA	NA	NA	NA	NA	NA
ssh	NA	NA	NA	NA	NA	NA
NA	SSH-2.0_CoreLab-1.0	NA	NA	NA	NA	NA
NA	NA	NA	NA	NA	NA	NA
ssh	NA	NA	NA	NA	NA	NA
NA	SSH-2.0-paramiko_2.0.2	NA	NA	NA	NA	NA
NA	NA	NA	NA	NA	NA	NA
NA	NA	root	admin	NA	NA	NA
NA	NA	NA	NA	NA	NA	NA
ssh	NA	NA	NA	NA	NA	NA

Figura 36. Dataframe de Cowrie

Es un paso grande el poder ver qué formato muestran los registros de los JSON, ya que no todos poseen los mismos campos y era imposible interpretarlos en bruto.

Como podemos observar, se registran protocolos utilizados, credenciales, IPs y puertos, de forma parecida al honeypot anterior. De nuevo, la cantidad de registros es grande y, para un

día como este (22 de mayo de 2023), que no fue de los que más eventos tuvo, ya contamos con unas 5000 líneas.

Nuestro siguiente reto consistió en asociar cada IP atacante con una ubicación geográfica y un ASN (Autonomous System Number). La idea nace de unas visualizaciones impresionantes que Hacker Target presentó en su análisis de este mismo honeypot [31].

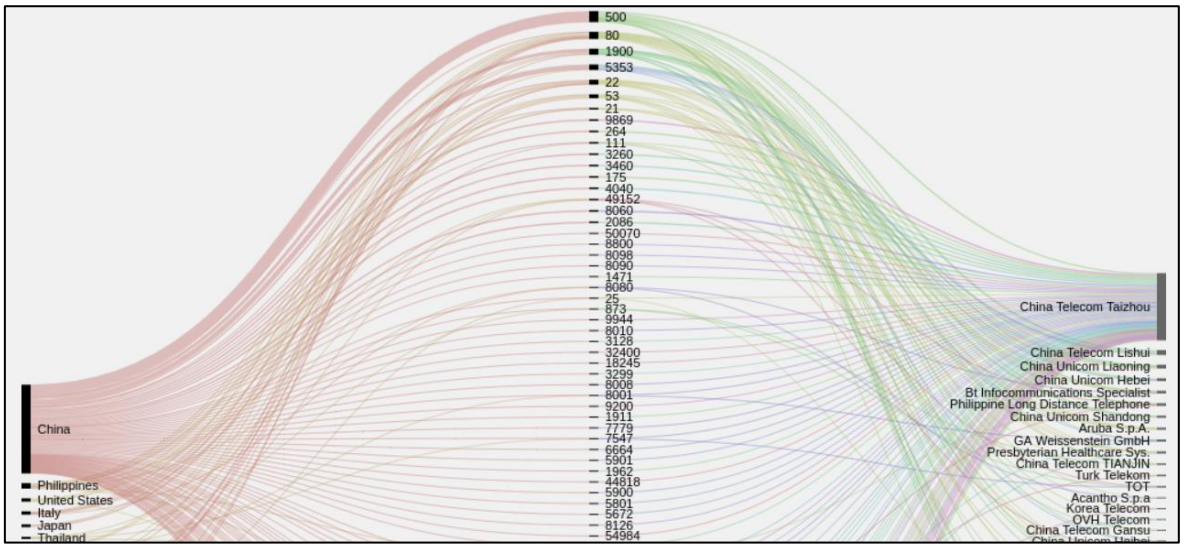

Figura 37. Puertos abiertos según país y organización (HT)

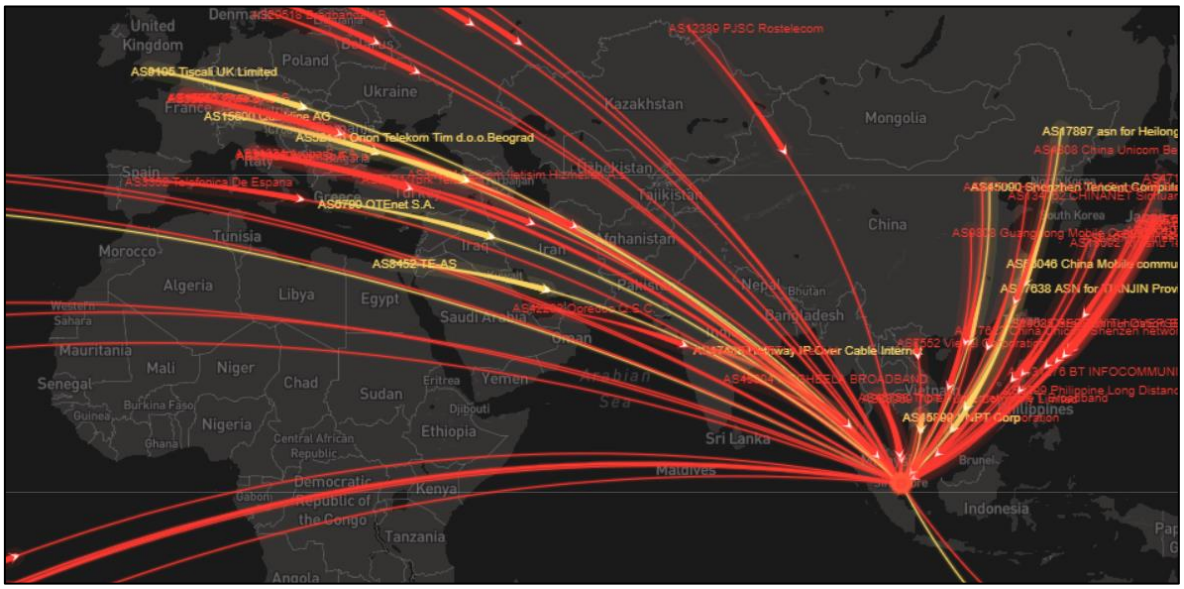

Figura 38. "Pew pew map" (HT)

El problema es que las API gratuitas que asocian IPs a ubicaciones geográficas suelen tener un límite de consultas. De nuevo, la mejor solución habría sido consultarlas a medida que se registraban, aunque es probable que el límite se hubiese sobrepasado, de todas formas. Ya que no contamos con una API de pago, se nos ocurre otra idea: descargar una base de datos que resuma rangos de direcciones y asocie estas con sus países y ciudades. Concretamente, MaxMind ofrece en su página web unos CSV que tienen este propósito (GeoLite2 [32]) y se actualizan periódicamente.

Una vez descargados, redactamos las líneas de código pertinentes para buscar las IP más comunes del JSON y asociarlas a sus países y ASNs. Es aquí donde la CPU nos capa, y no es para menos; los CSV son bastante extensos y el bucle tiene que recorrer demasiados registros.

Por suerte, encontramos en internet la aportación de un usuario, @jasonmpittman, que consiste, precisamente, en un sencillo programa que analiza registros de Cowrie y emite algunas estadísticas interesantes [33]. Optaremos, dadas nuestras limitaciones, por utilizarlo.

4.3 Visualizaciones y estadísticas

Para finalizar con el desarrollo del proyecto, vamos a observar algunas visualizaciones y estadísticas de los honeypots. A partir de ellas, elaboraremos nuestras conclusiones de los ataques y la información más relevante de sus registros.

4.3.1 Dionaea

Empezamos por Dionaea, y elaboramos unos primeros gráficos básicos en Tableau. El protocolo más explotado ha sido, con diferencia, SipSession, con más de un millón de registros. Recordamos que Sip es un protocolo especializado en tráfico multimedia (de voz, imágenes, vídeo…).

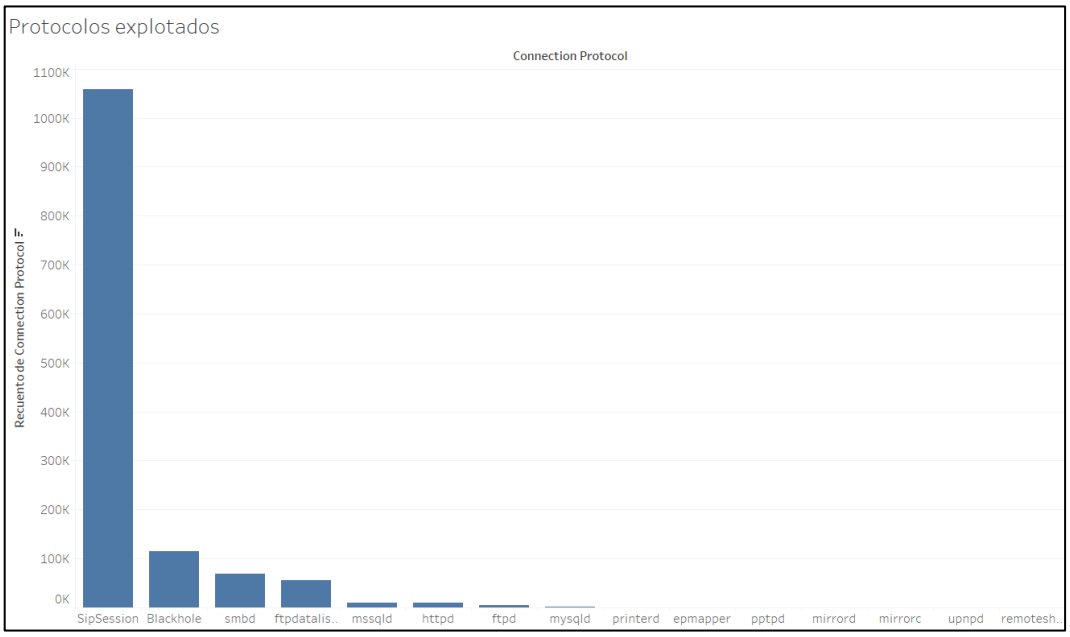

Figura 39. Protocolos explotados en Dionaea

Podemos excluirlo para tener una mejor imagen de la distribución del resto:

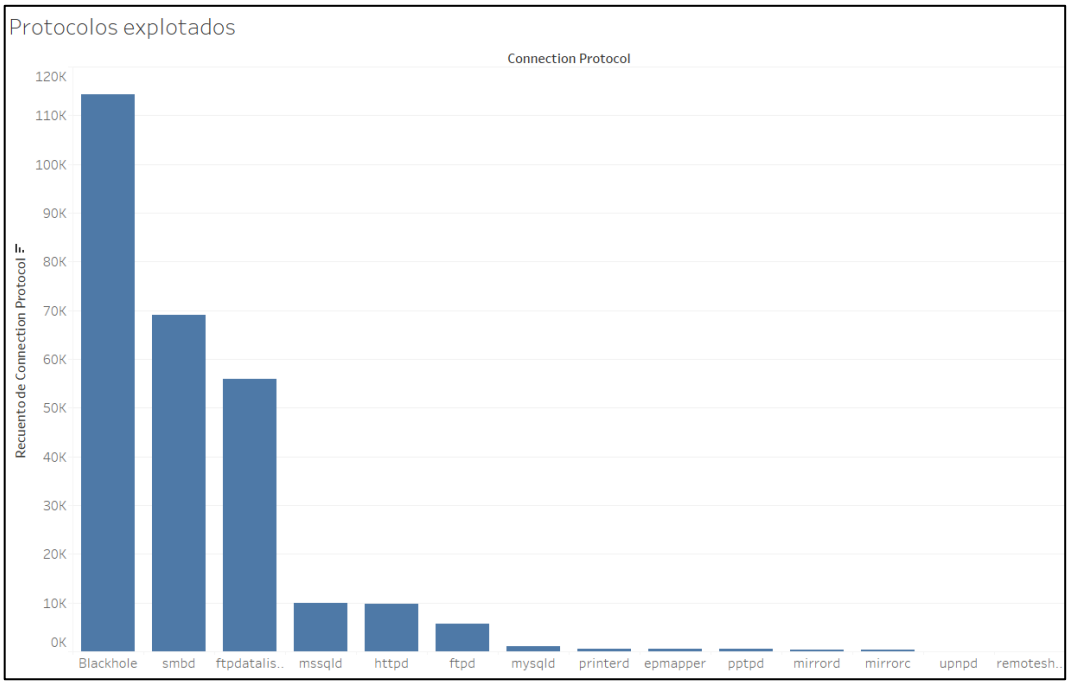

Figura 40. Protocolos exceptuando SipSession

El segundo más recurrente es Blackhole, seguido de smbd (de gestión de BBDD) y FTP.

Ahora observamos las direcciones IP de los atacantes. De nuevo, encontramos un *outlier* que sobrepasa a los demás en cantidad de registros por mucho. Se trata de la dirección 116.206.230.126, asociada a 840502 instancias.

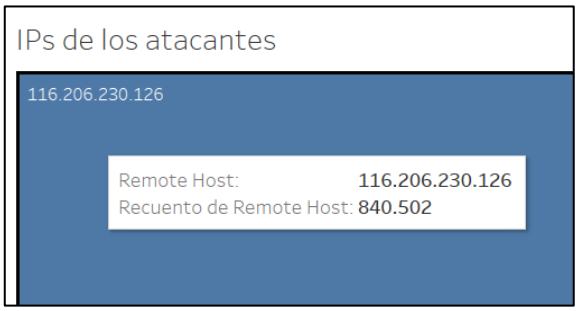

Figura 41. IP más repetida -Dionaea-

Lo más probable es que correspondan a una misma serie de ataques. La IP se ubica en Australia, pero no ha sido denunciada antes. Dichos ataques se realizan desde un vasto rango de puertos, aproximadamente 16000 distintos. De esta manera, el atacante trata de prolongar su actividad, evitando ser bloqueado tan rápidamente por los IDS. Observamos, como ejemplo, los más repetidos, situados entre el 50000 y el 66000:

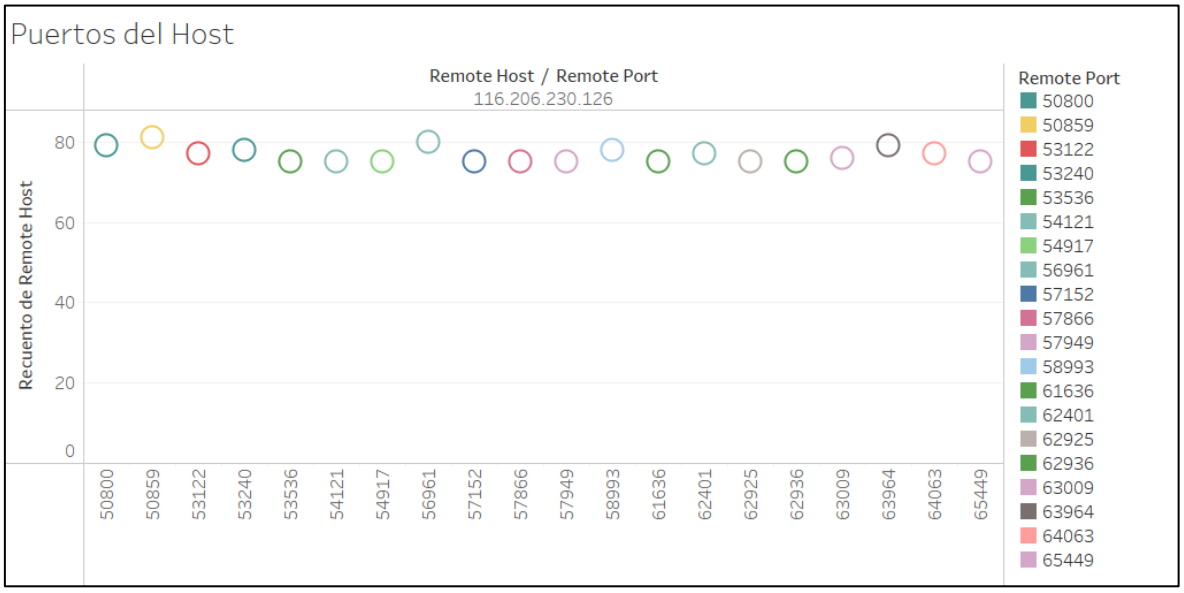

Figura 42. Puertos más repetidos del host

Podemos revisar, de entre las siguientes IPs, cuáles son las más comunes. La segunda más repetida es 94.26.121.188, con un recuento de 28160 registros. Tampoco ha sido denunciada previamente, y aparece en Palestina.

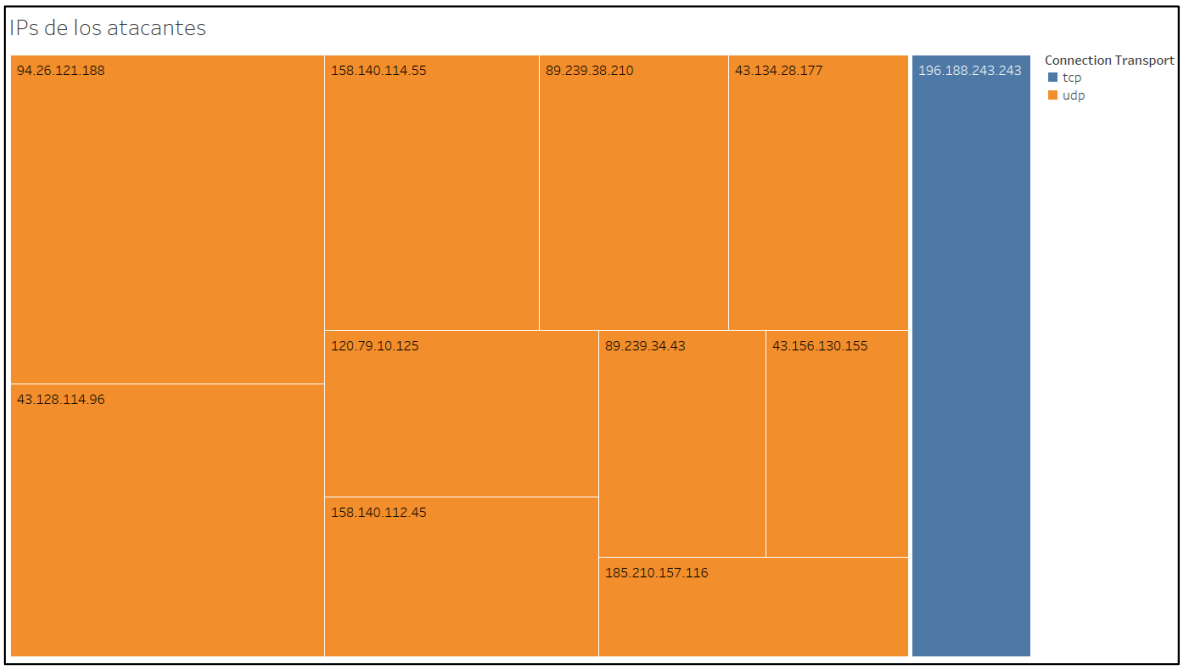

Figura 43. IPs más repetidas (exceptuando la primera) -Dionaea-

Las resumimos en una tabla:

Tabla III. IPs más repetidas -Dionaea-

Remote Host	Recuento	Denunciada	País
94.26.121.188	28.160	No	Palestina
43.128.114.96	23.316	No	Singapur
196.188.243.243	19.890	431 veces	Etiopía
158.140.114.55	16.075	No	Palestina
89.239.38.210	14.194	No	Palestina
43.134.28.177	13.498	No	Singapur
120.79.10.125	12.427	17 veces	China
158.140.112.45	11.907	No	Palestina
89.239.34.43	10.348	1 vez	Palestina

43.156.130.155	8.811	5 veces	Singapur
185.210.157.116	8.406	16 veces	Reino Unido

Sorprendentemente, casi ninguna ha sido denunciada, y las que lo han sido no las han reportado demasiadas veces. Además, se repite 5 veces Palestina como país de procedencia, y otras 3 Singapur. Es una distribución ciertamente extraña, aunque tampoco podemos afirmar una correlación directa a partir de una muestra de tan solo 10 direcciones. Lo que si vemos es que la mayoría utilizan UDP como protocolo de transporte. Estudiemos dicha distribución:

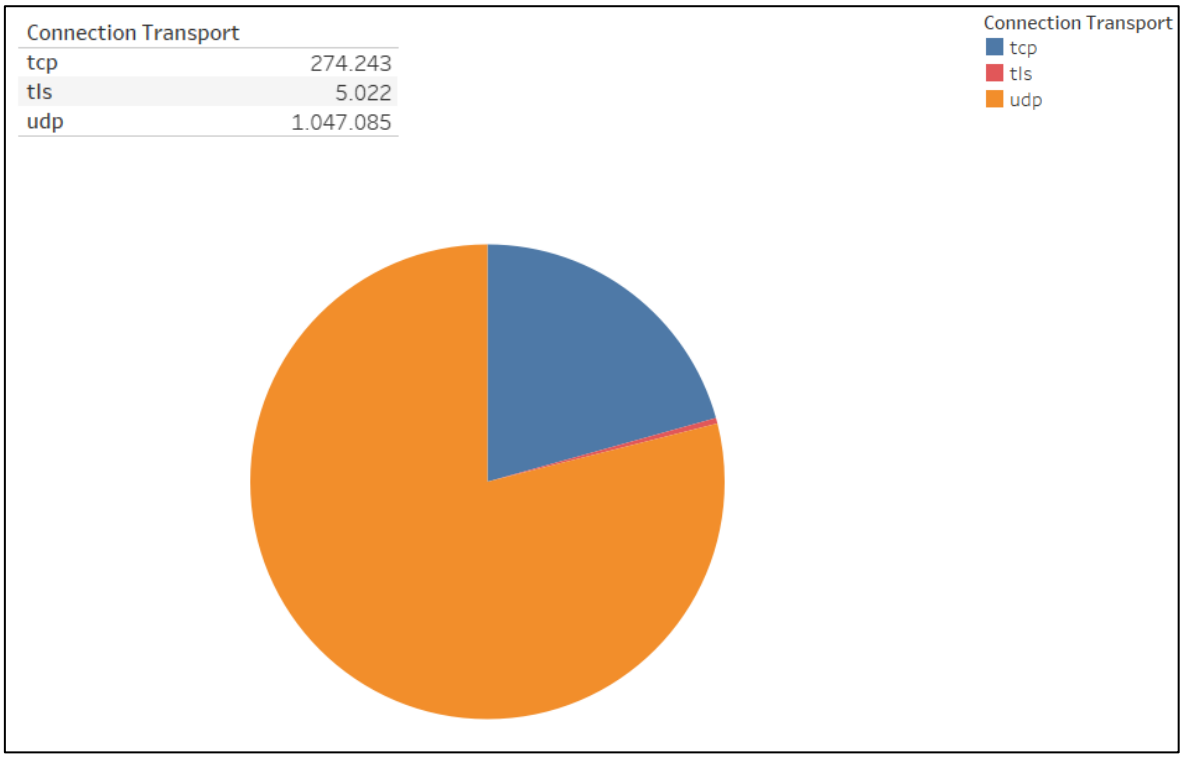

Figura 44. Distribución de los protocolos de transporte

Si incluimos todas las instancias, el 79% son UDP. Sin embargo, si excluimos las que se asocian a SipSession, el gráfico cambia considerablemente:

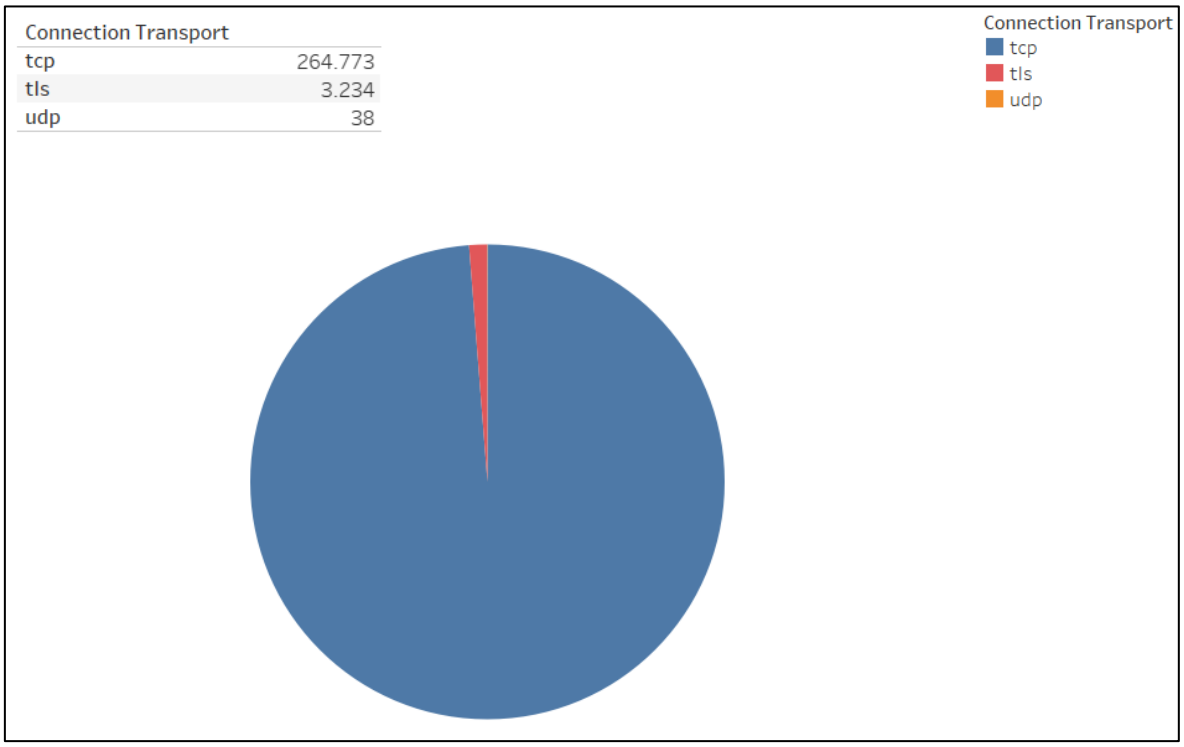

Figura 45. Distribución transp. exceptuando SipSession

Es decir, que la mayoría, que es SipSession, también es UDP.

Podemos estudiar ahora la tabla de *logins*. Empezamos viendo las contraseñas más repetidas (obviando la contraseña vacía, que es la que más se dio, con 1280 registros).

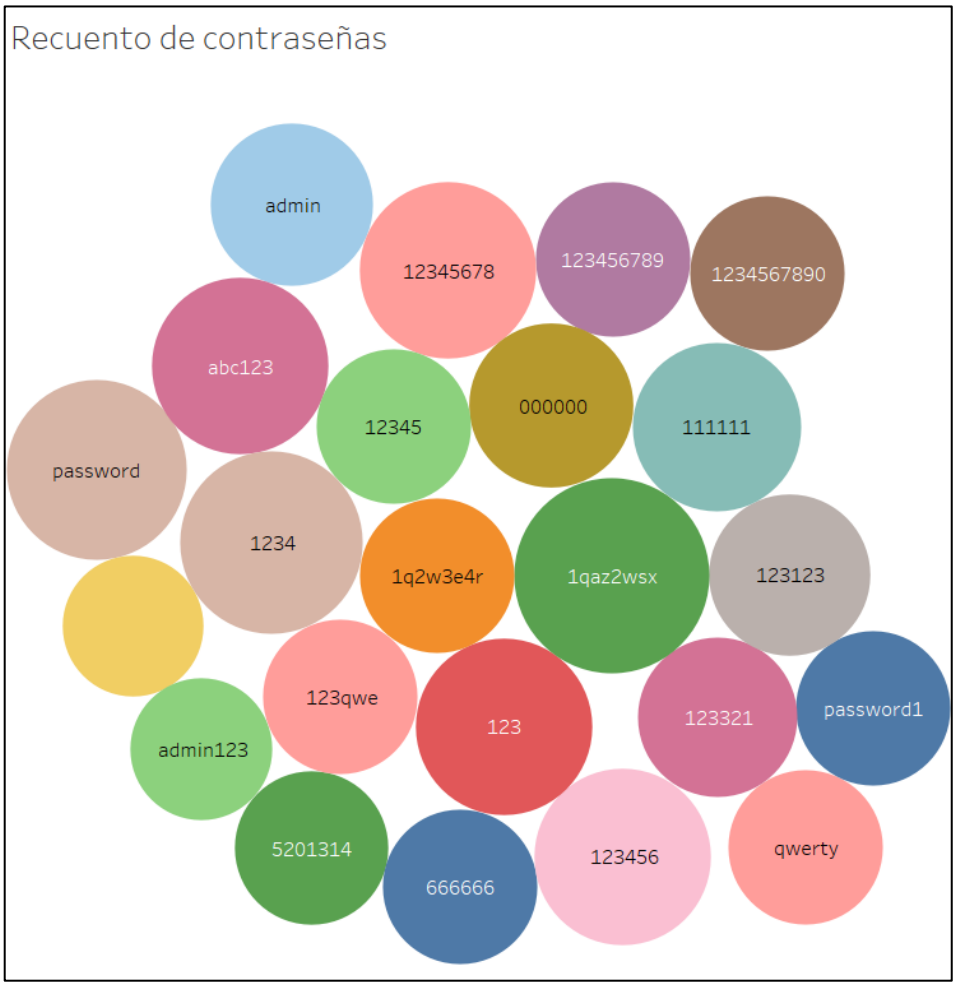

Figura 46. Contraseñas más habituales en Dionaea

Y hacemos lo mismo con los nombres de usuario más habituales:

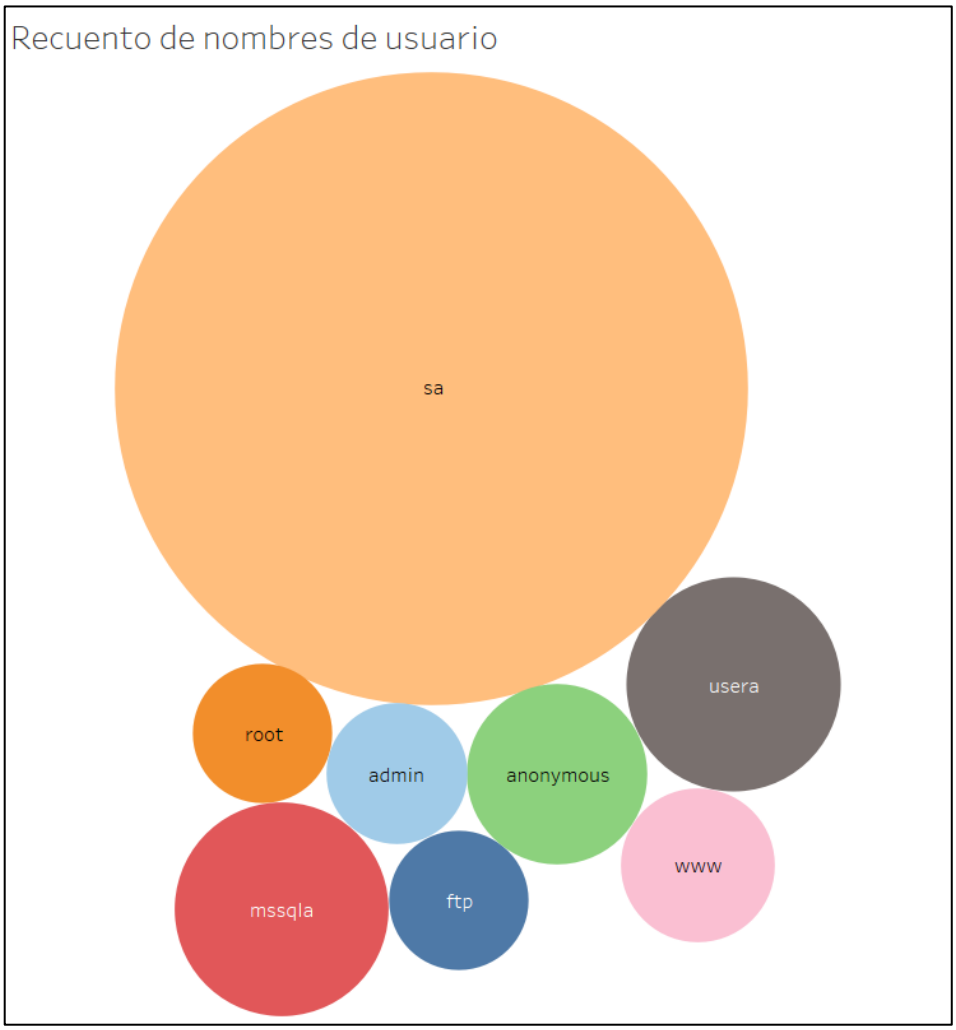

Figura 47. Nombres de usuario más habituales en Dionaea

En este caso, es "sa", con 6311 entradas, el más repetido. "sa" es la cuenta de un servidor SQL que posee privilegios de sistema, teniendo acceso a todas las tablas y poder para editar, eliminar o crear usuarios y bases de datos. Es **muy importante** limitar y proteger el acceso a esta cuenta, y no es nada recomendable usarla para gestiones para las que no sea imprescindible.

4.3.2 Cowrie

Como comentábamos en las *Mediciones finales*, para Cowrie nos hemos valido de un programa de análisis desarrollado por un usuario de GitHub. El programa es muy simple, pero nos es suficiente para el cometido que nos atañe.

Tomamos una muestra de todos los registros, ya que la capacidad de nuestro ordenador nos hace prohibitivo su análisis completo. Los resultados que devuelve el *script* figuran en forma de tablas y gráficos. Por ejemplo, podemos comprobar las direcciones IP que más se han repetido y, como hicimos con Dionaea, revisar si han sido denunciadas.

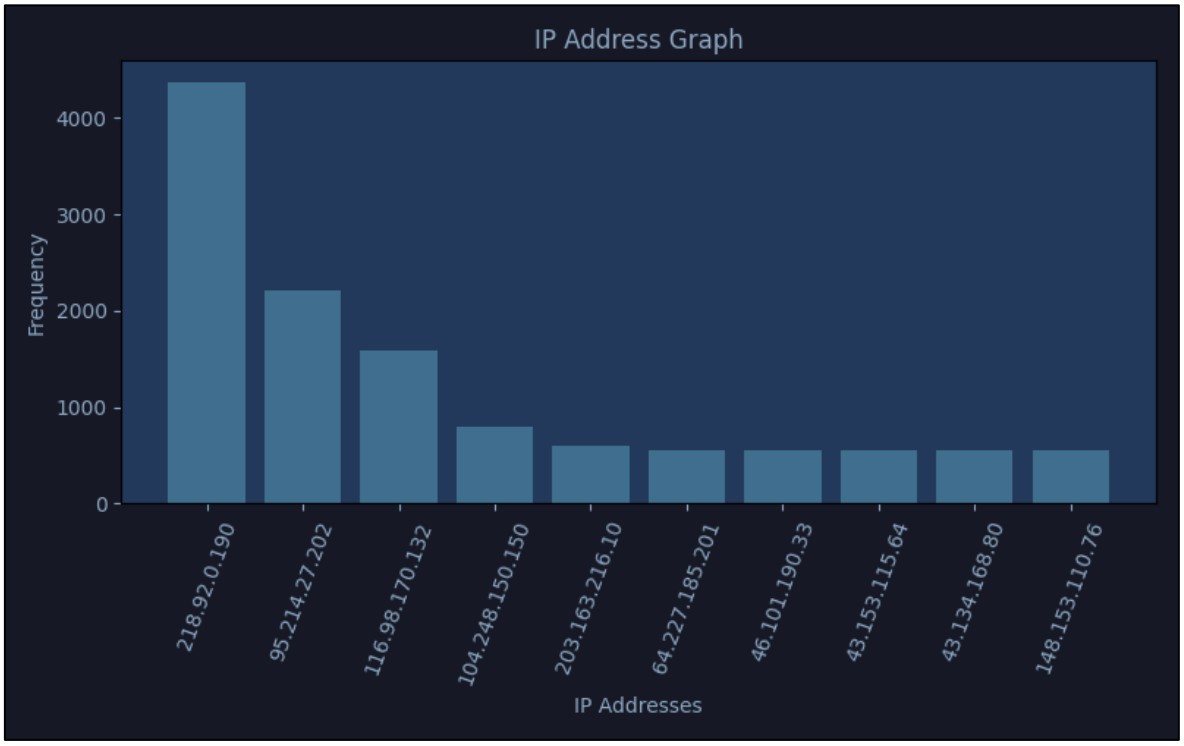

Figura 48. IPs más repetidas -Cowrie-

Las resumimos en una tabla:

Tabla IV. IPs más repetidas -Cowrie-

Remote Host	Recuento	Denunciada	País
218.92.0.190	4.300	30.114 veces	China
95.214.27.202	2.100	770 veces	Holanda
116.98.170.132	1.700	350 veces	Vietnam
104.248.150.150	900	16.417 veces	Singapur
203.163.216.10	800	89 veces	Taiwán
64.227.185.201	750	7.334 veces	India

46.101.190.33	700	2.036 veces	Alemania
43.153.115.64	650	1.000 veces	EEUU
43.134.168.80	650	3.568 veces	Singapur
148.153.110.76	600	15.254 veces	Alemania

En este caso, es mucho más obvio que las direcciones pertenecen a atacantes habituales. Lo más probable es que se trate de equipos automatizados que, periódicamente, barran rangos de direcciones IP en búsqueda de puertos abiertos y otras brechas de seguridad explotables. Por otra parte, no se observa ningún patrón a simple vista; los datos muestran una varianza razonable.

Otro dato que nos ofrece el *Log Analyzer* es el de los 10 países más comunes entre los atacantes. Los disponemos en un mapa, para estudiar si están relacionados o no:

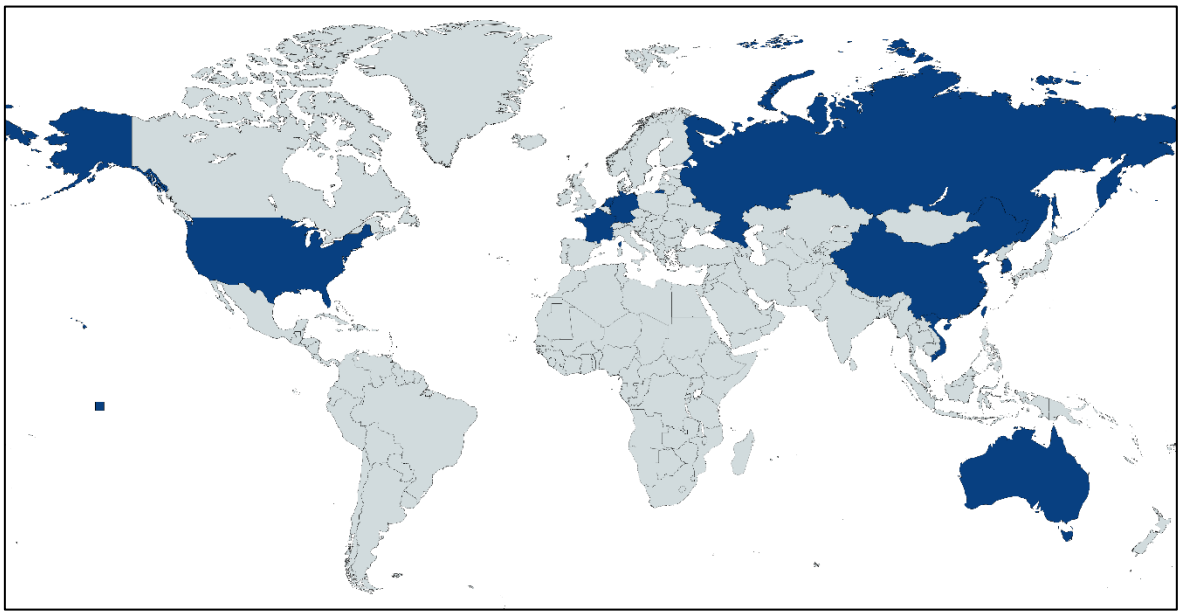

Figura 49. Mapa de los países más habituales -Cowrie-

Tampoco creemos que exista algún sesgo por proximidad, aunque nos llama la atención que ningún país sea de África o Sudamérica. Como en el caso previo, tan pocos países no nos valen para emitir un juicio definitivo.

Contamos con una lista de los 10 nombres y contraseñas más habituales, y también otra que condensa las parejas nombre-contraseña más repetidas.

Los ***usernames***:

1. root	6. pi
2. 345gs5662d34	7. test
3. admin	8. postgres
4. user	9. oracle
5. ubnt	10. ubuntu

Los ***passwords***:

1. 3245gs5662d34	6. 1234
2. 345gs5662d34	7. 123
3. 123456	8. 1
4. admin	9. 12345
5. password	10. pass

Y el **valor combinatorio** de ambos:

1. 345gs5662d34: 345gs5662d34	6. user: 1
2. root: 3245gs5662d34	7. user: user
3. admin: admin	8. root: root
4. root: 123456	9. uucp: uucp
5. root: admin	10. ubnt: password

Se parecen mucho a los de Dionaea, aunque aquí aparecen algunas diferencias, como la mayor frecuencia de “root” o el nombre “pi”, característico de la Raspberry. Lo importante es ser consciente de que una de las vulnerabilidades de las que más se aprovechan los ciberdelincuentes es la de las credenciales por defecto, que se suelen repetir y son muy fáciles de adivinar tras unos pocos intentos.

Capítulo 5. Conclusiones

5.1 Resumen e implicaciones de los resultados

La implementación de honeypots en una Raspberry Pi para simular un entorno IoT ha demostrado ser un método efectivo para analizar ciberataques. Los datos recopilados, analizados y visualizados han proporcionado un enfoque muy interesante sobre las tácticas utilizadas por los ciberdelincuentes, incluyendo la explotación de protocolos comunes, puertos inseguros y credenciales predeterminadas. Estos hallazgos subrayan la creciente importancia de la seguridad en el sector del IoT. El proyecto ha demostrado la efectividad de un enfoque proactivo, utilizando honeypots para anticipar amenazas emergentes, mejorar las defensas existentes y desarrollar estrategias de respuesta más efectivas. Las implicaciones de estos resultados son de gran alcance, destacando la necesidad de medidas de seguridad robustas en el paisaje del IoT, hoy en rápida expansión.

5.2 Línea de trabajo futuro

Basándonos en los hallazgos de este proyecto, el trabajo futuro podría centrarse en varias áreas. En primer lugar, el sistema de honeypots podría ampliarse para incluir una gama más amplia de dispositivos y protocolos IoT, proporcionando una imagen más completa de las amenazas que enfrenta el sector. En segundo lugar, se podrían emplear técnicas de análisis de datos más avanzadas para refinar aún más la comprensión de las tácticas de los ciberdelincuentes. Por ejemplo, se podrían utilizar algoritmos de aprendizaje automático para identificar patrones en los datos que pueden no ser inmediatamente aparentes.

Además, el proyecto podría ampliarse para incluir un sistema de respuesta en tiempo real, permitiendo tomar medidas inmediatas cuando se detecta una amenaza. Esto podría implicar bloquear automáticamente las direcciones IP asociadas con actividad sospechosa, o alertar a los administradores de la red sobre posibles amenazas. También se podría incorporar la información recopilada a una base de datos específicamente preparada para Big Data, con posibilidad de crecimiento horizontal si este se necesita.

Bibliografía

[1] Rose, K., Eldridge, S., & Chapin, L. (2015). The internet of things: An overview. The internet society (ISOC), 80, 1-50. Retrieved from https://www.internetsociety.org/wp-content/uploads/2021/01/ISOC-IoT-Overview-20151014_0.pdf

[2] Vailshery, L. S. (2022). *Number of IoT connected devices worldwide 2019-2021, with forecasts to 2030*. Retrieved from Statista: https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/

[3] Pawar, K., Ambhika, C., & Murukesh, C. (2021). IoT Hacking: Cyber Security Point of View. *Asian Journal of Basic Science & Research, 3*, 01-09. Retrieved from http://doi.org/10.38177/AJBSR.2021.3201

[4] Fariña Fernández-Portillo, A. (2022). Honeypot para dispositivos IoT. *Trabajo de Fin de Máster*. ICAI.

[5] RAYUELA. (2022). Obtenido de RAYUELA: https://www.rayuela-h2020.eu/

[6] Tabari, A. Z., Ou, X., & Singhal, A. (2021). What are Attackers after on IoT Devices? *Cornell University*. Retrieved from https://arxiv.org/pdf/2112.10974.pdf

[7] Microsoft. (2023). *About Us - Azure*. Obtenido de https://azure.microsoft.com/es-es

[8] Castro Astroz, D. A. (2018). Honeypot de dispositivos IoT usando una raspberry Pi 3. *Tesis de maestría*. Universitat Oberta de Catalunya. Obtenido de http://hdl.handle.net/10609/89885

[9] Armiñana Gorriz, J. (2018). Seguridad en internet de las cosas. *Trabajo Fin de Máster*. Universidad Oberta de Catalunya. Obtenido de http://hdl.handle.net/10609/89027

[10] Razali, M. F., Muruti, G., Razali, M. N., Jamil, N., & Mansor, F. Z. (2018). IoT Honeypot: A Review from Researcher's Perspective. *2018 IEEE Conference on Applications, Information and Network Security (AINS)*, 93-98. Retrieved from https://ieeexplore.ieee.org/document/8631494

[11] Kaspersky. (2023). ¿Qué es un honeypot? Obtenido de https://latam.kaspersky.com/resource-center/threats/what-is-a-honeypot

[12] Fortinet. (2023). *What are honeypots?* Obtenido de https://www.fortinet.com/resources/cyberglossary/what-is-honeypot

[13] Gallego, E., & E. López de Vergara, J. (s.f.). *Honeynets: Aprendiendo del atacante*. Obtenido de Universidad Politécnica de Madrid: https://web.dit.upm.es/~jlopez/publicaciones/mundointernet04.pdf

[14] Oosterhof, M. (2023). *Cowrie - Github*. Obtenido de https://github.com/cowrie/cowrie

[15] phibos. (2021). *Dionaea, DinoTools - Github*. Obtenido de https://github.com/DinoTools/dionaea

[16] desaster. (2016). *Kippo - Github*. Obtenido de https://github.com/desaster/kippo

[17] Mushorg. (2022). *Conpot - Github*. Obtenido de https://github.com/mushorg/conpot

[18] Dahua Security. (s.f.). *Dahua Vandal Proof Wi-Fi Dome*. Obtenido de https://www.dahuasecurity.com/asset/upload/download/DH-IPC-HDBW2201RP-ZSVFS1.pdf

[19] Aroganam, G., Manivannan, N., & Harrison, D. (2019). *Review on Wearable Technology Sensors Used in Consumer Sport Applications*. Obtenido de MDPI: https://www.mdpi.com/1424-8220/19/9/1983

[20] Sengupta, A. (2018). *Smartphone Age: The New Era of Human Evolution*. Obtenido de L&T: https://www.ltts.com/blog/smartphone-age-new-era-human-evolution

[21] Kaspersky. (s.f.). *¿Qué son los ataques DDoS?* Obtenido de https://latam.kaspersky.com/resource-center/threats/ddos-attacks

[22] Wikipedia. (2023). *dnsmasq*. Obtenido de https://es.wikipedia.org/wiki/Dnsmasq

[23] Hacks4Geeks. (s.f.). *HostAPD*. Obtenido de https://hacks4geeks.com/hostapd/

[24] mneumann. (2017). *Using the Raspi as a WIFI Hotspot*. Obtenido de PTC Community: https://community.ptc.com/t5/IoT-Tips/Using-the-Raspberry-Pi-as-a-WIFI-hotspot/ta-p/820359

[25] riupie, & ahezza. (2019). *How to Stop Dionaea on Ubuntu?* Obtenido de GitHub: https://github.com/DinoTools/dionaea/issues/232#issuecomment-471510405

[26] tarn, & joeforker. (2018). *Bind to ports less than 1024 without root access*. Obtenido de Serverfault: https://serverfault.com/questions/268099/bind-to-ports-less-than-1024-without-root-access

[27] SQLite. (2023). *SQLite - Index*. Obtenido de https://sqlite.org/index.html

[28] Salesforce. (s.f.). *Tableau*. Obtenido de https://www.tableau.com/

[29] Tableau. (s.f.). *Customize an ODBC Connection*. Obtenido de Tableau Help: https://help.tableau.com/current/pro/desktop/en-us/odbc_customize.htm

[30] Piacentini, M. (2023). *DBBrowser for SQLite*. Obtenido de Tabuleiro: https://sqlitebrowser.org/

[31] Hacker Target. (s.f.). *Cowrie Honeypot Analysis*. Obtenido de https://hackertarget.com/cowrie-honeypot-analysis-24hrs/

[32] MaxMind. (2023). *GeoLite2 - CSV downloads*. Obtenido de https://www.maxmind.com/en/accounts/884388/geoip/downloads

[33] jasonmpittman. (2020). *Cowrie Log Analyzer*. Obtenido de GitHub: https://github.com/jasonmpittman/cowrie-log-analyzer

[34] ONU. (2022). *Objetivo 9. Industria, Innovación e Infraestructuras*. Obtenido de Objetivos de Desarrollo Sostenible: https://www.un.org/sustainabledevelopment/es/infrastructure/

[35] ONU. (2022). *Objetivo 16. Paz, justicia e instituciones sólidas*. Obtenido de Objetivos de Desarrollo Sostenible: https://www.un.org/sustainabledevelopment/es/peace-justice/

Anexo I. Alineación del proyecto con los ODS

El autor muestra su firme compromiso con los ODS y enfoca su proyecto desde un prisma de ética y responsabilidad en la seguridad de los sistemas de la red.

El trabajo comparte los ODS de su antecesor [4]. En primer lugar, el ODS número 9, relativo a construir infraestructuras resilientes, promover la industrialización sostenible y fomentar la innovación [34], se garantiza a través de un trabajo enfocado en la investigación y el desarrollo en el sector del IoT.

Por otra parte, también se alinea con el ODS número 16, que busca promover sociedades justas, pacíficas e inclusivas [35]. La gran mayoría de los conflictos actuales se dan en el espacio ciber, al que han migrado las ofensivas entre naciones. Un entorno de red seguro y robusto que garantice un tráfico sin vulnerabilidades es sinónimo de un mundo en paz, o al menos un vector a este.