UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Domótica segura: Desarrollo de un sistema de control y monitoreo de dispositivos del hogar con Raspberry Pi

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Avis Garrido, Guillermo

Tutor/a: Pons Terol, Julio

CURSO ACADÉMICO: 2023/2024

Dedicado a mis abuelos. Por cuidarme y ayudarme a crecer como persona.

Resum

El projecte proposa abordar la necessitat creixent de millorar la seguretat i la gestió de dispositius en entorns residencials, a causa de l'augment del nombre d'aquests dispositius a les llars. Es proposa oferir una solució a aquest problema mitjançant l'ús d'eines executades a una Raspberry Pi.

El sistema proposat permet a l'usuari supervisar l'ús dels diferents dispositius de la llar, des de sistemes d'il·luminació fins a electrodomèstics amb connexió a la xarxa. A més, el sistema compta amb funcions de protecció contra possibles atacs a la placa i contra l'obtenció de dades dels dispositius per afegir seguretat a la nostra casa.

El procés de desenvolupament del projecte inclou des de la configuració de la Raspberry Pi, la descàrrega, configuració i implementació de les eines i la implementació de la seguretat. També s'inclouen proves per avaluar l'eficàcia i l'ús del sistema.

El projecte de Domòtica Segura pretén contribuir a un ús segur d'una llar intel·ligent, brindant als usuaris un entorn residencial més segur, eficient i connectat mitjançant la implementació d'un sistema integral basat en Raspberry Pi.

Paraules clau: Domòtica, seguretat, iot, llar intel·ligent, Raspberry Pi, Monitorització, Internet de les coses

Resumen

El proyecto propone abordar la creciente necesidad de mejorar la seguridad y gestión de dispositivos en entornos residenciales, debido al aumento del número de estos dispositivos en los hogares. Se propone ofrecer una solución a este problema mediante el uso de herramientas ejecutadas en una Raspberry Pi.

El sistema propuesto permite al usuario supervisar el uso de los distintos dispositivos del hogar, desde sistemas de iluminación hasta electrodomésticos con conexión a la red. Además, el sistema cuenta con funciones de protección contra posibles ataques a la placa y contra la obtención de datos de los dispositivos para agregar seguridad a nuestro hogar.

El proceso de desarrollo del proyecto abarca desde la configuración de la Raspberry Pi, la descarga, configuración e implementación de las herramientas y la implementación de la seguridad. También se incluyen pruebas para evaluar la eficacia y uso del sistema.

El proyecto de "Domótica Segura" pretende contribuir a un uso seguro de un hogar inteligente, brindando a los usuarios un entorno residencial más seguro, eficiente y conectado mediante la implementación de un sistema integral basado en Raspberry Pi.

Palabras clave: Domótica, seguridad, iot, hogar inteligente, Raspberry Pi, Monitorización, Internet de las cosas

Abstract

The project proposes to address the growing need to improve device security and management in residential environments, due to the increasing number of these devices in homes. It is proposed to offer a solution to this problem through the use of tools executed on a Raspberry Pi.

The proposed system allows the user to monitor the use of different devices in the home, from lighting systems to network-connected appliances. In addition, the system has protection functions against possible attacks on the board and against obtaining data from devices to add security to our home.

The project development process ranges from configuring the Raspberry Pi, downloading, configuring and implementing the tools and implementing security. Tests are also included to evaluate the effectiveness and use of the system.

The "Secure Home Automation" project aims to contribute to the safe use of a smart home, providing users with a safer, more efficient and connected residential environment through the implementation of a comprehensive system based on Raspberry Pi.

Key words: Home automation, security, iot, smart home, Raspberry Pi, Monitoring, Internet of Things

Índice general

Índice general VII
Índice de figuras IX
Índice de tablas X

1 Introducción 1
1.1 Motivación . . . 2
1.2 Objetivos . . . 2
1.3 Estructura de la memoria . . . 4
2 Estado del arte 7
2.1 Crítica al estado del arte . . . 8
2.2 Propuesta . . . 9
3 Análisis del problema 11
3.1 Análisis de la seguridad . . . 12
3.2 Análisis del marco legal y ético . . . 12
3.3 Identificación y análisis de soluciones posibles . . . 13
3.3.1 Elección de la placa . . . 13
3.3.2 Elección del software de control de domótica . . . 13
3.3.3 Elección del sistema operativo . . . 14
3.3.4 Sistema de contenedores . . . 14
3.3.5 Protección del acceso a la placa . . . 15
3.3.6 Certificado SSL . . . 16
3.3.7 Elección de broker y colas de mensajería . . . 16
3.4 Solución propuesta . . . 16
3.4.1 Elección del sistema . . . 17
3.4.2 Protección de la placa . . . 17
3.4.3 Broker de mensajería . . . 17
3.4.4 Plan de Desarrollo . . . 17
3.4.5 Plan de implantación . . . 18
3.4.6 Pruebas de funcionamiento . . . 18
4 Diseño de la solución 21
4.1 Arquitectura del Sistema . . . 21
4.1.1 Arquitectura general . . . 21
4.2 Diseño Detallado . . . 22
4.3 Tecnología Utilizada . . . 24
4.3.1 Hardware . . . 24
4.3.2 Software . . . 24
5 Contexto y antecedentes 25
5.1 Dispositivos y aplicaciones utilizadas . . . 25
5.2 Análisis del tráfico . . . 26
6 Desarrollo de la solución 29
6.1 Instalación del sistema operativo . . . 29
6.2 Protección del acceso ssh . . . 31

6.2.1 Configuración de un nombre para el servicio . . . 31
6.2.2 Uso de claves para ssh . . . 32
6.3 Configuración de una VPN . . . 34
6.3.1 Instalación de Wireguard y configuración inicial . . . 34
6.3.2 Modificaciones para el acceso con la VPN desde fuera de la red . . . 36
6.3.3 Instalación del software en el cliente . . . 37
6.4 Instalación y uso de Home Assistant . . . 39
6.4.1 Instalación y configuración de Docker . . . 39
6.4.2 Preparación del contenedor de Home Assitant . . . 40
6.4.3 Añadiendo seguridad mediante SSL . . . 41
6.4.4 Configuración de Home Assistant . . . 43
6.4.5 Agregar dispositivos . . . 44
6.5 Problemas con MQTT y uso de una alternativa para la comunicación directa con el dispositivo . . . 46
6.5.1 Problemas encontrados con mosquitto MQTT . . . 46
6.5.2 Configuración avanzada para Tuya . . . 46
6.5.3 Desarrollo del código . . . 49
6.6 Integración con home assistant . . . 50
6.7 Firewall . . . 52
7 Implantación **55**
7.1 Disposición del sistema domótico . . . 55
7.2 Agregados al sistema . . . 55
7.2.1 Visualizado de logs . . . 56
7.2.2 Creación de un cron de actualización . . . 56
8 Pruebas **59**
8.1 Funcionamiento del sistema . . . 59
8.2 Comprobación de la seguridad de ssh . . . 59
8.2.1 Prueba y errores encontrados . . . 59
8.2.2 Solución de fallos . . . 59
8.2.3 Intento de conexión sin acceso a la VPN . . . 60
8.3 Comprobación de la configuración de la *VPN* . . . 61
8.4 Análisis de los paquetes tras la introducción de *openssl* . . . 61
8.5 Análisis del tráfico home assistant . . . 62
8.6 Análisis del tráfico con el uso del script . . . 62
9 Conclusiones **65**
9.1 Relación del trabajo desarrollado con los estudios cursados . . . 66
Bibliografía **67**

Apéndices
A Glosario **69**
B ODS - Objetivos de Desarrollo Sostenible **71**

Índice de figuras

4.1 Esquema del sistema 21
4.2 Esquema capas 22
4.3 Diagrama del sistema 23

5.1 Enchufe usado para el proyecto. 25
5.2 Tráfico de red Tuya con la app. 27
5.3 Captura de la página who is 27

6.1 Menú *Raspberry Pi imager* 29
6.2 Selección de la placa 30
6.3 Selección del SO 30
6.4 Fichero host linux 32
6.5 Añadir entrada al DNS 32
6.6 Generación de clave con keygen 33
6.7 Generación clave privada en el cliente 33
6.8 Desactivación contraseña *ssh* 34
6.9 Asignar IP fija a la raspberry 36
6.10 Port fordwarding a raspberry Pi 37
6.11 Inicio wireguard 38
6.12 Activación VPN 38
6.13 Grupos agregados 39
6.14 Hello World Docker 39
6.15 Fichero yml de configuración 40
6.16 Fichero descargados por Home Assitant 41
6.17 Generar certificado 42
6.18 Página con certificado 43
6.19 Pestaña de inicio Home Assitant 43
6.20 Menú inicio *Home assistant* 44
6.21 Añadir dispositivo con *Home Assistant* 44
6.22 Buscar *firmware* 45
6.23 Introducir código aplicación Tuya 45
6.24 Página de desarrollo de Tuya 47
6.25 Campos proyecto 47
6.26 Vincular dispositivos desde la web 48
6.27 Dispositivos vinculados 48
6.28 Obtención claves secretas dispositivo 49
6.29 Prueba funcionamiento script 50
6.30 Seleccionar scripts 51
6.31 Ventana overview de home assistant 52

8.1 PasswordAuhentication en sshd 60
8.2 Conexión ssh rechazada por la raspberry 60
8.3 Intento de entrar a *home assistant* desde fuera de la *VPN* con *IP* subred . . 61
8.4 Captura *wireshark* de *wireguard* 61

8.5 Captura *wireshark* tras realizar la configuración de *openssl* 62
8.6 Captura de las capas de uno de los paquetes de *Home Assistant* 62
8.7 Tráfico capturado desde la Raspberry 62
8.8 Tráfico capturado cuando se utiliza el *script* 63

Índice de tablas

3.1 Tabla comparativa Raspberry Pi 13

CAPÍTULO 1
Introducción

El mundo ha entrado en una era en la que la tecnología está cada vez más presente en nuestras vidas. El uso de los dispositivos inteligentes está creciendo exponencialmente, cambiando la forma en la que interactuamos con nuestro entorno. Este avance ha permitido la creación de hogares inteligentes donde los dispositivos se conectan entre sí para brindar comodidad, eficiencia y ahorro energético. En este contexto, la Raspberry Pi, una placa que actúa como un pequeño computador, es una herramienta de bajo coste que puede ser clave para el desarrollo de sistemas de automatización del hogar.

Aunque la domótica y el uso de dispositivos inteligentes prometen hacer más sencillas las vidas de los usuarios, también plantean un desafío significativo de seguridad y privacidad. La interconexión constante y el envío de información del hogar a servidores remotos abren la puerta a múltiples vulnerabilidades que podrían comprometer la privacidad y la integridad de la información de los usuarios.

El presente Trabajo Fin de Grado (TFG) profundiza en la creación de un sistema de domótica con medidas de seguridad utilizando la Raspberry Pi como pieza central. Este proyecto aborda por una parte el uso de la placa para el control de los dispositivos del hogar, y por otra, las preocupaciones de seguridad asociadas a la domótica mediante el uso de protocolos de comunicación seguros y otras estrategias para proteger la privacidad.

A lo largo de estas páginas, se examinará la arquitectura del sistema propuesto, destacando las medidas de seguridad tomadas. Estas medidas se basarán en:

- Protección de la red doméstica: Implementación de *firewalls*, redes Wi-Fi seguras y aislamiento de dispositivos para minimizar el riesgo de intrusiones.
- Cifrado de datos: Utilización de protocolos de comunicación seguros como *HTTPS* y *SSH* para proteger la confidencialidad de la información transmitida.
- Autenticación y control de acceso: Implementación de mecanismos de autenticación robustos para garantizar que solo usuarios autorizados puedan acceder al sistema domótico.
- Actualizaciones de software: Mantenimiento de los dispositivos actualizados con los últimos parches de seguridad para eliminar vulnerabilidades conocidas.

Además de estos aspectos, también se aborda el cifrado de las comunicaciones a los dispositivos vinculados, para así aumentar la privacidad de los datos compartidos.

En conclusión, este TFG pretende contribuir en el uso seguro de la domótica, haciendo una propuesta no solo centrada en la centralización del uso de los dispositivos del hogar, sino que también en la protección contra posibles "ciberamenazas". La conectividad entre

dispositivos es cada vez mayor, por lo que es fundamental que los avances en tecnología y seguridad avancen a la par. Este trabajo pretende contribuir a ambos aspectos.

1.1 Motivación

La elección de la temática del TFG es para muchos alumnos un elemento de preocupación y discordia. A menudo el problema no es la falta inspiración sino la falta de motivación que generan las ideas pensadas, lo cual lleva al descarte de muchas de las mismas para la realización del proyecto. En mi caso, tras el descarte de varias ideas, encontré un elemento sobre el cual quería centrar la temática de mi TFG, la Raspberry Pi. Sentí una curiosidad por aprender sobre el uso de Raspberry Pi, una herramienta que puede facilitar la realización de infinidad de proyectos y que, a su vez, abarca todos los elementos necesarios para aplicar y ampliar los conocimientos adquiridos durante la carrera.

Una vez encontrado el que sería el eje de mi proyecto llegue al segundo aspecto en el que quería centrar mi proyecto, la domótica. La domótica y el Internet de las Cosas (IoT) son aspectos que se han ido introduciendo poco a poco en mi hogar, y que han generado en mi inquietudes sobre su funcionamiento y uso. Quería llevar a cabo un proyecto que abarcará estos dos elementos y tras investigar proyectos relacionados con ambos aspectos llegué a la idea de este proyecto.

Otro de los puntos importantes por los cuales tuve interés por la temática, fue la privacidad. La privacidad es un aspecto fundamental que tanto en el aspecto legal como en el aspecto informático se pretende defender, ya que día a día estamos más expuestos a que otras personas puedan tener acceso a nuestra información. Este es el riesgo fundamental que conlleva la digitalización que el mundo esta llevando a cabo y uno de los elementos más sensibles para la privacidad de las personas son los dispositivos del hogar. Son múltiples los datos que se pueden extraer de una personas mediante sus dispositivos. Desde lo más evidente, como cámaras o micrófonos integrados en dispositivos como alexa, hasta información como el tiempo de uso de nuestros dispositivos, el tiempo que pasamos en nuestro hogar o incluso información sobre nuestros hábitos.

Por otra parte, otros aspectos tratados son el uso y monitorización de la domótica. Estos son puntos en los que también tengo interés en profundizar ya que el ahorro energético y el control de la privacidad que pueden suponer son muy relevantes para usuarios de un hogar digital como es mi caso. Estos aspectos son fundamentales para el correcto uso y funcionamiento de un hogar conectado. El ser consciente de que están haciendo tus dispositivos y saber que información recopilan desde un lugar unificado concede un mayor control y propician un mejor uso de los dispositivos domésticos. Además en el aspecto de consumo, estar al tanto de lo que realizan tus dispositivos y poder programar sus rutinas puede suponer un ahorro tanto económico como ecológico.

La oportunidad de investigar y contribuir en este campo es el principal motivo de la elección de esta temática. Poder contar con un hogar conectado y seguro debería ser accesible para todas las personas que tengan la voluntad de hacerlo. Por ello, quiero llevar a cabo este proyecto para que, con el desarrollo de este trabajo, otras personas puedan implementar un sistema similar y disfrutar de una domótica controlada y segura.

1.2 Objetivos

El objetivo principal del proyecto es el desarrollo de un sistema accesible y con medidas de seguridad para la integración de dispositivos domóticos. Además de este objetivo, también se desarrollará como objetivo secundario el análisis y desarrollo de medidas

para aumentar la privacidad del usuario, evitando que empresas de terceros tengan control sobre los propios dispositivos y sobre la información que manejan los mismos. Para la consecución de dichos objetivos principales propongo la realización de los siguientes subobjetivos más concretos:

- Identificar que Raspberry Pi escoger para el proyecto.
  - Profundizar en las especificaciones técnicas de cada placa, incluyendo modelos y diferencias.
  - Explorar distintos programas compatibles con el sistema de Raspberry.
  - Analizar casos de uso previos de Raspberry Pi en domótica.
- Explicar paso a paso la configuración de la Raspberry Pi.
  - Describir paso a paso el proceso de configuración inicial de la Raspberry Pi, desde la instalación del sistema operativo hasta la configuración de red, usuario y acceso remoto.
  - Explicar como conectar y configurar periféricos relevantes para el proyecto.
- Selección e instalación de programas de domótica del hogar.
  - Investigar y evaluar distintas soluciones de software para la monitorización y control de dispositivos domésticos compatibles con la Raspberry Pi, considerando aspectos como funcionalidad, facilidad de uso y soporte.
  - Detallar el proceso de instalación y configuración de los programas elegidos.
- Implementación de las medidas de seguridad para la protección de los dispositivos.
  - Identificar y analizar posibles amenazas de seguridad en el sistema domótico, como vulnerabilidades y accesos no autorizados.
  - Diseñar e implementar estrategias de seguridad adecuadas como el uso de cortafuegos, el uso de cifrado de comunicaciones, el uso de certificados y la gestión de permisos.
  - Explorar opciones de conexión a los dispositivos en local para aumentar la privacidad del usuario.
- Realización de pruebas de funcionamiento y de seguridad.
  - Definir una serie de pruebas que incluyan pruebas de funcionalidad, rendimiento y resistencia a ataques.
  - Ejecución de pruebas planificadas analizando los resultados obtenidos.
  - Identificar y corregir posibles fallos y vulnerabilidades descubiertas durante las pruebas y documentar los fallos, justificando las decisiones tomadas.
- Evaluación y conclusiones
  - Evaluar el rendimiento y la efectividad del sistema domótico implementado.
  - Extraer conclusiones sobre los logros alcanzados, los desafios enfrentados y lo aprendido del desarrollo del trabajo.
  - Proponer posibles mejoras o lineas de investigación futuras para continuar con el desarrollo en el campo de la domótica.

La consecución de estos objetivos puede servir de guía para que un usuario medio pueda lograr tener un sistema domótico seguro y privado para su hogar.

1.3 Estructura de la memoria

La estructura de la memoria es la siguiente:

1. Introducción:
   - Apartado en el que se ha presentado la importancia de la domótica y la relevancia de la Raspberry Pi como plataforma de desarrollo del proyecto.
   - Exposición de los objetivos del trabajo, destacando la configuración de la placa, la instalación de programas, la implementación de medidas de seguridad y las pruebas correspondientes.
2. Estado del arte:
   - Se realiza una revisión de los trabajos previamente realizados de domótica y uso de Raspberry Pi tanto en TFGs como proyectos externos.
   - Se realiza una crítica a los proyectos ya realizados y a la seguridad actual.
   - Se realiza una propuesta de proyecto en el que se plantea como se abordarán los problemas mencionados.
3. Análisis del problema:
   - Análisis de la seguridad y del marco legal del proyecto.
   - Análisis de las distintas opciones a considerar tanto de software como de hardware para la consecución del proyecto.
   - Soluciones elegidas tras el análisis realizado previamente.
4. Diseño de la solución:
   - Descripción del diseño de la solución propuesta.
   - Justificación de las decisiones de diseño tomadas incluyendo arquitectura, componentes y tecnologías usadas.
5. Contexto y antecedentes:
   - Descripción de los elementos que se usarán en el proyecto.
   - Análisis del tráfico de red de los dispositivos previo al proyecto.
6. Desarrollo de la solución propuesta:
   - Se detalla el proceso de desarrollo de la solución.
   - Se proporcionan detalles sobre la programación y configuración utilizados.
7. Implantación:
   - Se explica como se lleva a cabo la implementación práctica de la solución en un entorno real.
   - Se describen los pasos necesarios para instalar y configurar los dispositivos domóticos para el hogar del usuario final.
8. Pruebas:
   - Se presenta un plan de pruebas para verificar el correcto funcionamiento del sistema.

- Se exponen los resultados obtenidos y se discuten las posibles mejoras.

9. Conclusiones:
   - Se resumen los resultados más importantes del trabajo realizado y se extraén las conclusiones.
   - Se discuten las limitaciones y se proponen mejoras y recomendaciones para el propio trabajo o para trabajos futuros.

- Apéndices:
  - Apéndice A: Definición de algunos términos utilizados.
  - Apéndice B: *ODS* - Objetivos de desarrollo sostenible

CAPÍTULO 2
Estado del arte

El auge de la domótica ha propiciado el desarrollo de múltiples sistemas de control y automatización de los dispositivos del hogar. A continuación, se exponen algunas de las tecnologías ya existentes que ofrecen funcionalidades similares o relacionadas con el sistema propuesto en este TFG.

Haciendo foco en el apartado de plataformas y aplicaciones que realizan funciones de control de dispositivos inteligentes tenemos:

1. Sistemas domóticos comerciales:
    - Google Nest[1]: Plataforma creada por google para el control del hogar inteligente. Integra camáras de seguridad, sistemas de iluminación, termostatos y otros dispositivos inteligentes.
    - SmartThings[2]: Aplicación para la gestión de dispositivos inteligentes compatibles con la app.
    - HomeKit[3]: Aplicación diseñada por Apple para la gestión de hogares inteligentes.
2. Sistemas compatibles con Raspberry Pi:
    - OpenHAB[4]: Plataforma de código abierto que permite la automatización de los dispositivos del hogar.
    - Domoticz[5]: Sistema domótico de código abierto con documentación sobre Raspberry Pi que ofrece soporte para una gran variedad de dispositios.
    - Jeedom[6]: Plataforma domótica con una comunidad activa y de código abierto.
    - Home Assitant [7]: Es una de las plataformas más populares y con comunidad más activa en domótica. También es un software de código abierto y posee fácilidad de integración con múltiples soluciones y dispositivos.

Las plataformas comerciales son plataformas muy poco flexibles y que tienen muchas limitaciones a la hora de configurar dispositivos ya que solo los compatibles podrán ser utilizados. En cuanto a las de código abierto y compatibles con Raspberry Pi, algunas serán utilizadas como parte del proyecto, explicando sus ventajas y funcionalidades. Estas dependen mucho de la comunidad, ya que al no tener el soporte de una gran empresa detrás se requiere de la comunidad de usuarios para mejorar y se arreglar los posibles fallos encontrados.

Fijándonos en proyectos concretos de control de dispositivos IoT, encontramos el libro "Internet of things programming projects"[8]. En dicho libro podemos encontrar una

explicación detallada de como realizar un proyecto en el cual se utiliza Raspberry Pi como plataforma y python como lenguaje de programación. Se explica paso por paso como realizar la instalación del sistema operativo en la Raspberry Pi y como dar los primeros pasos tanto de control de los dispositivos como de su protección. Este proyecto se asemeja al proyecto que quiero llevar a cabo pero se basa en el establecimiento de un servidor web y la creación de pequeños proyectos para aprender a controlar dispositivos sin profundizar en la creación de un sistema completo.

Finalmente, en cuanto al apartado de mejoras en la seguridad de sistemas domóticos existen una serie de proyectos que buscan introducir mejoras de seguridad en los dispositivos del hogar. Un ejemplo de esto es el proyecto "Artificial Intelligence Based Domotics Using Multimodal Security",[9] que busca mediante el uso de inteligencia artificial mejorar la seguridad de la domótica implementando datos biométricos. Con la utilización de datos biométricos busca cerciorarse de que solo usuarios autorizados tienen acceso al control de los dispositivos. Este proyecto también incluye protocolos de alertas por correo, bloqueo biométrico, control móvil, control de electrodomésticos y acceso por PIN. Todo esto son medidas que buscan la mejora de la seguridad al poner trabas al acceso no autorizado, consiguiendo también que en caso de vulneración del sistema el usuario pueda ser informado. Además de todo esto se ofrece una interfaz gráfica para el facilitar al usuario las tareas de control y seguridad. Todo esto tiene unas finalidades similares a las de mi proyecto, proporcionar una solución para el control y la mejora de la seguridad de los dispositivos del hogar.

2.1 Crítica al estado del arte

En este apartado realizaré un análisis de los proyectos realizados por otros alumnos de la universidad. Existen una gran variedad de proyectos que usan Raspberry Pi como plataforma de desarrollo y, que además, se centran en la domótica y en el control de dispositivos del hogar.

Analizando los proyectos que más se asemejan a mi trabajo de final de grado,[10] [11] [12] [13] podemos observar que son muy similares entre sí ya que hacen el foco en la integración de los distintos dispositivos IoT y en la interacción con los mismos mediante el uso de Raspberry Pi.

Analizando uno por uno los trabajos ya citados encontramos en primer lugar el trabajo de "Casa domótica con arduino"[10]. Este trabajo, a parte de la notable diferencia de usar Arduino en vez de Raspberry Pi, se centra más en los aspectos de diseño de la interfaz web y en el control de dispositivos concretos. El aspecto que se podría expandir es el control de un mayor número de dispositivos pero Arduino tiene mayores limitaciones que la Raspberry Pi para llevar a cabo integraciones más amplias.

En segundo lugar, tenemos el proyecto de [11] "Integración de sistemas domóticos mediante una plataforma de código abierto". En este proyecto se hace un trabajo similar al que realizaré en mi TFG, pero el proyecto analizado se centra en la adaptación del sistema al protocolo X10. Mi proyecto ampliará al mismo centrándose en el control del sistema domótico, la protección del mismo y el control de las interacciones con los dispositivos domóticos. Todo esto usando la Raspberry Pi como plataforma para el sistema.

En tercer lugar tenemos [12] "Instalación y configuración de un sistema integral de hogar inteligente basado en plataformas de software libre". Este proyecto utiliza OpenHAB y Mosquitto, dos opciones de software libre que se barajarán como software a utilizar para mi proyecto. La diferencia fundamental de mi proyecto con respecto a este y el valor diferenciador serán las medidas de seguridad añadidas al sistema.

Por último, tenemos el proyecto de [13] "Diseño e implementación de proyecto e infraestructura IoT". Este proyecto se centra en ideas similares a las que quiero tratar y usa Raspberry Pi como plataforma central, pero se centra más en la utilización de un asistente de voz para el control de los dispositivos. Mi aportación diferencial con respecto a este proyecto sigue la linea de la mencionada para los anteriores: el control y la protección del sistema y los dispositivos.

Algunos aspectos los cuales busco compartir con algunos de dichos proyectos (Como [11] o [12]) son el uso de aplicaciones "open source" ya que siguen una filosofía basada en compartir y mejorar con la comunidad. Muestran una mayor transparencia con los usuarios ya que pueden ver como funciona el programa, evita engaños y funcionalidades ocultas.

En resumen, el aspecto en el que mi trabajo busca profundizar y aportar un valor son la integración y el control de los dispositivos, temas comunes a los otros trabajos, y diferenciarse con el desarrollo de medidas de seguridad para hacer que el control de nuestro hogar y los dispositivos del mismo sean más seguros.

2.2 Propuesta

El sistema va a estar basado en el uso de Raspberry Pi por lo que uno de los aspectos principales será la protección del acceso a la placa. Para mejorar la seguridad del sistema se usarán distintas técnicas para limitar el acceso a la placa únicamente a usuarios correctamente autenticados. Se deberán aplicar técnicas que se centren en minimizar el riesgo de que un usuario atacante gane acceso al sistema averiguando o forzando la contraseña de un usuario con privilegios. Este proceso pasará por proteger el acceso *ssh* y la implantación de una *VPN* y reglas de *firewall* para proteger la entrada de la placa. Las posibles medidas concretas serán investigadas y analizadas en un capítulo posterior.

En lo que respecta al control de la domótica se utilizará un software o plataforma ya existente para el control de los dispositivos. Esta plataforma tendrá una interfaz web la cual podrá ser accedida al conectarse a través de la *VPN*. Se cifrará el acceso a la interfaz web para añadir una capa extra de seguridad. A través de la interfaz se tendrá acceso al control de los dispositivos y a la visualización del estado general del sistema. Se podrán conectar los distintos dispositivos mediante el uso del *firmware* de los mismos pero además se explicará como usar un *broker* de colas de mensajería para así poder controlar los mensajes que se envían a los dispositivos.

Una vez realizada toda la configuración de seguridad se ampliarán detalles sobre la implantación y algunas medidas extra que se podrán realizar para implantar el sistema y configurar nuevos dispositivos. Por último, se realizarán pruebas de funcionamiento analizando el tráfico de la red y observando como el contenido va cifrado y nadie puede acceder a él, además se tratarán de hacer accesos no autorizados a la placa y a la plataforma de control de domótica.

CAPÍTULO 3
Análisis del problema

En la actualidad son muchas las alternativas existentes para la centralización del control de sistemas domóticos y cada vez es mayor su crecimiento tanto en número de seguidores como en comunidad de usuarios. Pero los dispositivos domóticos contienen información muy sensible, información que de ser obtenida por otros vulneraría la privacidad o incluso la intimidad del usuario. Para el usuario promedio, en la mayoría de casos, su seguridad pasaría por la confianza depositada en las plataformas comerciales ofrecidas por empresas como Amazon, Samsung, Apple o Google. A pesar de que estas empresas se han ganado la confianza de los usuarios para comprar y usar sus servicios, siguen teniendo el recelo de una parte del público que ha visto como grandes empresas han sido acusadas de la venta de datos para su beneficio. Un ejemplo de esto es el caso de Meta[14] (Empresa dueña de redes como Facebook, Whatsapp, Instagram...) que tuvo que pagar 725 millones por filtrar los datos de 87 millones de usuarios y vendérselos a la empresa de *Cambridge Analytica*.

Es por ello que darle al usuario la posibilidad de tener una plataforma más segura con la que usar sus dispositivos y, que además, le aporte un mayor control de la domótica del hogar es clave para la lucha por una mayor privacidad. Cada vez es mayor el número de usuarios que contribuye a alternativas *open source* para la domótica del hogar. *Home assistant* es una de las plataforma más populares y extendidas entre los usuarios que usan alternativas a las plataformas comerciales.

Por otra parte, a pesar de usar software *open source*, que nos brinda un mayor control de las acciones que realizamos y la posibilidad de ver y contribuir con el funcionamiento del código, también existen otras brechas de seguridad hacia nuestra privacidad. Por ejemplo, en una red en la que hay más usuarios, bien sea porque sea una empresa o porque se conviva o comparta el internet con otras personas del hogar, los dispositivos y el control de los mismos están expuestos al resto de usuarios. Por ello, es conveniente aislar dentro de la red los propios dispositivos y limitar el acceso al control de los mismos.

Por último, también existe el problema de las propias conexiones que los dispositivos realizan para enviar la información ya que no tenemos el control de las mismas. Esto puede derivar en varios problemas, como que las comunicaciones no estén fuertemente cifradas, lo cual vulneraria la integridad y confidencialidad de nuestra información y por otra parte la falta de control de donde se envía nuestra información. Esto último es importante, ya que nuestros dispositivos se comunican con un servidor externo el cual hace de intermediario entre el usuario y el dispositivo. Esta situación puede ser un gran riesgo para nuestra privacidad debido a que todos los datos y ordenes enviadas a los dispositivos pasan por dicho servidor.

3.1 Análisis de la seguridad

La seguridad es crucial en los sistemas domóticos dado que los datos que contienen son sensibles.

A continuación se analizarán las principales vulnerabilidades:

- Conexión a internet: la conexión a internet es un punto de acceso sensible en gran parte de los sistemas pero en la mayoría de casos es imprescindible el acceso a internet para el funcionamiento del *software* y de los dispositivos.
- Autenticación débil: muchos de los dispositivos de domótica carecen de un sistema de autenticación. Reforzar los sistemas de autenticado o añadirlos en caso de que no existan es fundamental para evitar accesos no autorizados.
- Actualización de software: Mantener el software actualizado es un factor que aunque parezca evidente o no tan relevante es crucial para mantener la seguridad de los dispositivos. Cada vez que aparece una actualización significa que existen vulnerabilidades descubiertas y reparadas por el fabricante o desarrollador para las cuales no estamos protegidos.

También es relevante analizar los ataques potenciales que pueden sufrir nuestros dispositivos:

- Ataques de Denegación de Servicio (DoS): En nuestro ámbito de red doméstica no serán tan comunes pero es un posible ataque. Este consiste en la realización de acciones contra los dispositivos, las cuales provocan que los mismos dejen de poder proporcionar las funcionalidades habituales. Esto puede hacerse de varias maneras como sobrecargando de peticiones los dispositivos.
- Intrusiones en la red: Los atacantes pueden intentar infiltrarse en la red doméstica a través de posibles brechas de seguridad y acceder a datos sensibles.
- Suplantación de identidad: Si el atacante accede a nuestros datos y obtiene alguna credencial o identificación podría emplearlos para suplantar nuestra identidad.

3.2 Análisis del marco legal y ético

Cuando se realiza un proyecto sea de la índole que sea un aspecto a tener en cuenta es el marco legal y ético, sobre todo en los proyectos tecnológicos y que involucran información sensible. Es por ello que a continuación analizaré los aspectos legales y éticos:

- La ley de protección de datos. La ley de protección de datos a la cual estamos sujetos en España es a la LOPD (Ley orgánica de protección de datos) y esta establece la normativa por la cual se deben regir las instituciones o empresas las cuales almacenan o tratan información sensible de los usuarios. En el caso del proyecto llevado a cabo, se van a tratar únicamente nuestros propios datos y no datos de terceros, por lo cual es recomendable que tratemos de salvaguardarlos pero no tenemos gran parte de las obligaciones que exige esta ley.
- Ley de propiedad intelectual. En este aspecto el software que vamos a utilizar va a ser *open source* por lo que no requerimos de ningún tipo de licencia. En caso de requerirse deberá ser adquirida, respetando así los derechos de propiedad intelectual del autor del producto.

- Ética: Garantizar la privacidad y la seguridad de los usuarios, tanto miembros del hogar como visitas, es una obligación moral en la consecución del proyecto. Se debe considerar también el tratamiento y almacenamiento de los datos como un aspecto sensible del proyecto. Por último, es un aspecto relevante en el ámbito de la ética el respeto hacia los derechos de propiedad intelectual del software o hardware utilizado.

3.3 Identificación y análisis de soluciones posibles

3.3.1. Elección de la placa

Se han investigado distintas placas de Raspberry PI realizando una tabla comparativa de sus características (tabla 3.1). Se han analizado las distintas tareas para las cuales se va a utilizar la placa y no se requieren de demasiados recursos para llevar a cabo la tarea. La Raspberry Pi 4 es un modelo estable y para el cual hay múltiples proyectos desarrollados *online*, además de dar un amplio abanico de posibilidades en cuanto a la *RAM*. Un aspecto a tener en cuenta es la conectividad de la placa ya que nos puede dar más opciones para la conexión con distintos dispositivos. A pesar de esto, la elección dependerá sobre todo del presupuesto que el usuario esté dispuesto a gastar y de las labores extras que se requiera realizar con la placa a la par que el sistema domótico.

Modelo Pi	Procesador	RAM	Conectividad	Precio
Raspberry Pi 4	4 núcleos	2-8 GB	Gigabit, Wi-Fi ac, Bluetooth 4.2	90€+
Raspberry Pi 400	4 núcleos	4 GB	Gigabit, Wi-Fi ac, Bluetooth 4.2	90€+
Raspberry Pi Zero WH	1 núcleo	512 MB	Wi-Fi n, Bluetooth 4.1	30€+
Raspberry Pi 3 B+	4 núcleos	1 GB	Gigabit, Wi-Fi n, Bluetooth 4.1	80€+

Tabla 3.1: Tabla comparativa Raspberry Pi

3.3.2. Elección del software de control de domótica

Un punto clave es la elección del sistema de control y monitorización de la domótica para ello se han explorado las opciones de código abierto más compatibles con Raspberry Pi. Las distintas opciones son:

- Home Assistant: es una plataforma de domótica de código abierto de gran popularidad. Ofrece compatibilidad con una amplia gama de dispositivos, además de una interfaz de usuario cómoda para la gestión. Tiene una comunidad amplia y una buena documentación. Como contra, requiere de ciertos conocimientos técnicos para su instalación, uso y mantenimiento.

- OpenHAB: es una plataforma de código abierto que ofrece gran flexibilidad y personalización. Da soporte a un gran número de dispositivos y posee una comunidad activa. En contra, requiere también de ciertos conocimientos técnicos para su uso.

- Domoticz: es una plataforma que a diferencia de las dos anteriores tiene una mayor facilidad de uso y una interfaz simple. Como contra, carece de las características avanzadas que poseen las dos opciones anteriores.

- IoTivity: es una plataforma que permite la comunicación de dispositivos IoT y protege las conexiones. Como contra, requiere configuración extra para ciertos dispositivos.

El criterio para elegir el software de control de domótica será que la plataforma ofrezca la posibilidad de configurar el mayor número posible de elementos del sistema, que tenga soporte tanto de usuarios como de documentación, que sea de código abierto y que sirva para configurar el mayor número de dispositivos posibles. Además se valorarán que los pasos a seguir para configurar el dispositivo sean accesibles y estén bien documentados.

3.3.3. Elección del sistema operativo

En la Raspberry Pi, como en todo computador, se pueden instalar múltiples sistemas operativos compatibles. La elección del mismo dependerá de las necesidades y los objetivos que se pretendan alcanzar con el uso de la placa. Las opciones son las siguientes:

- Rasberry Pi OS (Anteriormente denominado *Raspbian*): es el sistema operativo oficial de Raspberry Pi. Por tanto, está optimizado para aprovechar al máximo los recursos del hardware y viene preinstalado con herramientas útiles para el uso de la placa. Es fácil de usar e instalar. Como contra, carece de algunas características de otros sistemas operativos así como de compatibilidades con nuevas funcionalidades.

- Ubuntu: es una gran opción dentro de los sistemas operativos basados en linux. Posee un soporte sobresaliente y una gran comunidad, además de tener compatibilidad con nuevas aplicaciones y funcionalidades. Como contra es más pesado y puede ralentizar un equipo limitado como es la Raspberry Pi.

- Arch Linux: es una distribución de linux el cual ofrece un sistema optimizado, eficiente y actualizado. Como contra ofrece una interfaz que requiere a un usuario más avanzado y requiere de más mantenimiento.

- Sistema operativo *Home Assistant*: en el caso de la elección de *Home Assistant* como plataforma, esta ofrece la opción de instalar el sistema operativo directamente sobre la placa lo cual facilita la tarea y elimina complejidad de instalación. Como contra limita más el uso de la placa ya que se depende de las posibilidades que te ofrezca el sistema operativo y del soporte del mismo.

La elección del sistema operativo dependerá del uso óptimo del sistema, el soporte, el aprovechamiento de la placa, la versatilidad y la posibilidad de configurar un mayor número de elementos.

3.3.4. Sistema de contenedores

En el caso en el que no sé utilice el sistema operativo de la plataforma de domótica, se debe tener un sistema para desplegar la imagen de ese sistema operativo en un contenedor. Para realizar esto la opción más extendida, utilizada y con mayor soporte es *Docker* y será la opción que se utilice en este caso.

3.3.5. Protección del acceso a la placa

Uno de los aspectos clave para este TFG es la protección y la limitación del acceso a la Raspberry Pi ya que es el eje central del proyecto. Para ello, se han valorado distintas alternativas para la consecución de este objetivo. En este punto las alternativas no son novedosas, pero se han valorado una combinación de alternativas existentes para llevar a cabo el proceso de protección.

El acceso mediante *ssh* será lo más utilizado, ya que es más cómodo para el acceso directo a la placa que la conexión de los distintos periféricos cada vez que se requiera hacer un cambio de configuración. Por ello, se han explorado distintas alternativas:

- Uso de políticas más estrictas. Algunas de estas políticas pueden ser: establecer contraseñas más complejas para dificultar el forzado de las mismas, limitar el número de intentos fallidos y/o el tiempo entre intentos de acceso. Estas medidas pueden resultar insuficientes, aunque aumentan la seguridad.

- Uso de *fingerprints*. El uso de *fingerprints* consiste en la generación de claves públicas y privadas tanto en el cliente como en el servidor para así conseguir un acceso seguro y cifrado. Solo los usuarios que posean las claves podrán acceder. Esta medida puede ser más compleja de gestionar si múltiples dispositivos quieren acceder a la placa, pero si son un menor número de dispositivos como es el caso, puede ser una buena solución.

- Uso de kerberos. Kerberos es un protocolo de autenticación muy utilizado en la actualidad. Consiste en la negociación para la obtención de *tickets* lo cual permite un acceso temporal al dispositivo o dispositivos que pertenecen al dominio del *ticket* de kerberos. Es un buen sistema de obtención de acceso pero para un sistema pequeño y doméstico puede ser innecesariamente complejo.

Las medidas a tomar se elegirán para tratar de proteger el acceso sin configurar elementos innecesarios.

Otra de las medidas tomadas para un acceso más seguro a la placa será el uso de *VPN*, se han explorado las opciones más extendidas de uso de *VPN* las cuales son:

- OpenVPN: es una de las opciones más usadas para el establecimiento de *VPNs* y que además tiene un buen funcionamiento en Raspberry Pi. Es de código abierto y además posee multiples guias para la configuración en Raspberry Pi. Tiene un gran número de opciones de configuración lo cual es positivo pero aumenta la complejidad de su configuración. Esto puede suponer una mayor barrera para usuarios no tan familiarizados con la administración. Además es una opción que usa más recursos que otras opciones.

- WireGuard: Es una *VPN* que está creciendo en popularidad debido a su simplicidad y eficiencia. Tiene mejor rendimiento que *OpenVPN* y mayor simplicidad, pero menos guías en comparación.

- PiVPN: Es una alternativa especifica para Raspberry Pi la cual facilita la instalación y configuración de *OpenVPN* en el dispositivo mediante *scripts*.

La *VPN* se elegirá centrándose en una configuración sencilla para que un usuario no tan avanzado sea capaz de configurarlo, que además exista una amplia documentación y que tenga un buen rendimiento.

3.3.6. Certificado SSL

Una vez configurado el sistema de la placa e instalada la plataforma de control de domótica se podrá añadir una medida extra de protección de la conexión con la interfaz de *home assistant* mediante el uso de certificados *SSL*. Para ello, se usará un certificado autogenerado mediante el uso de *openSSL* ya que es la opción más extendida y al ser para uso propio no se requiere la adquisición de un certificado más aceptado en plataformas como *let's encrypt*.

Otro de los motivos por los cuales no se ha elegido el certificado de *lets encrypt* es porque este tiene una validez temporal máxima de 90 días, lo cual fuerza a su renovación cada mucho menos tiempo, comparado con *openssl* cuyos certificados pueden durar hasta 10 años. Además al no ser el sistema de domótica un servidor accesible desde la red o que vaya a estar expuesto al público como una página web, se excluye completamente el uso de *Let's Encrypt*. Esto es debido a que uno de los requisitos para la obtención de un certificado de esta página es la validación del servidor y, para esta validación, se requiere superar unos "challenges" o desafíos de validación que implican que el servidor responda a solicitudes específicas desde la red. Estos desafíos se realizan para evitar otorgar certificados a dominios sobre los que no se tiene control.

En resumen, la elección de *openssl* simplifica el proceso de mantenimiento y reduce la carga al eliminar la necesidad de renovar certificados con regularidad. Esto permite la obtención del cifrado que conlleva el certificado y con ello la mejora en la seguridad.

3.3.7. Elección de broker y colas de mensajería

Una opción para la comunicación segura con los dispositivos es el uso de colas de mensajería. Con ese objetivo se deberá investigar el uso de un *broker* de mensajería o de una tecnología de colas de mensajería. Algunas de estas opciones son:

- Mosquitto MQTT broker: es un *broker* de colas ligero, con una fácil integración con plataformas domóticas como *Home Assistant* y un gran soporte con la comunidad IoT.
- Apache Kafka: Altamente escalable y tolerante a fallos. Como contra requiere de un mayor volumen de recursos y es más compleja.
- RabbitMQ: soporta distintos protocolos de mensajería y también posee una alta disponibilidad y escalabilidad. Como contra es más compleja y requiere de más recursos.

El criterio en este caso será el menor uso de los recursos ya que la Raspberry tiene unos recursos límitados, además de la compatibilidad y capacidad de integración con plataformas de integración de domótica. Por otro lado, se tendrá en cuenta la documentación que haya para trabajos de este tipo.

3.4 Solución propuesta

Tras la investigación y el análisis de los distintos recursos y tecnologías necesarias para llevar a cabo el trabajo se han seleccionado las más óptimas para la realización del trabajo.

3.4.1. Elección del sistema

Respecto a la elección de placa, como se ha mencionado en el apartado anterior, la mayoría tienen los recursos necesarios para poder desarrollar el sistema de este proyecto. Por ende, la elección de la placa dependerá del presupuesto o de las labores que se quieran realizar con la placa al margen del proyecto. En caso del presente TFG, se ha elegido una Raspberry Pi 4 Modelo B (4GB) debido a que es una de las más potentes, con mayor conectividad (aspecto importante para la conexión con múltiples dispositivos domóticos), de más número de puertos y que posee una amplia comunidad de usuarios activos.

Como plataforma de control de domótica se ha elegido *Home Assitant* principalmente por su gran comunidad de usuarios, su software de código abierto, su compatibilidad con otros dispositivos, sus opciones de configuración y su compatibilidad con otros programas.

Con relación al sistema operativo se ha escogido el Raspberry Pi OS puesto que es uno de los sistemas operativos que más aprovecha los recursos de la placa y ofrece amplias opciones de configuración y soporte. Se ha valorado el uso del sistema operativo de *Home Assitant*, pero si se quiere dar otros usos a la Raspberry Pi puede ser muy limitante y por tanto se utilizará un contenedor para lanzar la imagen del SO de *Home Assitant*. El contenedor como se ha mencionado en la sección anterior será lanzado con *Docker* ya que es la alternativa más extendida y utilizada.

3.4.2. Protección de la placa

Tras la valoración de las alternativas se ha optado por el uso de *fingerprints*, debido a que el acceso se va a hacer desde uno o dos dispositivos por lo que la configuración de las mismas no requerirá de muchos cambios en la configuración y aumentará la seguridad mejorando la autenticación. Una política de configuración que se llevará a cabo es la desactivación de la contraseña para evitar una posible intrusión mediante fuerza bruta.

En el aspecto de la *VPN*, la elección ha sido *Wireguard* debido a la facilidad de su configuración, el menor uso de recursos con respecto a *OpenVPN* y la sencillez en la realización de cambios.

Como se ha mencionado también, el certificado se realizará empleando *openssl*.

3.4.3. Broker de mensajería

Mosquitto *MQTT* ha sido la elección en cuanto al *broker* de mensajería ya que es muy fácilmente integrable con *Home Assistant*, se utiliza mucho en domótica, tiene una amplia documentación y es de código abierto.

3.4.4. Plan de Desarrollo

El desarrollo consistirá en la instalación del sistema operativo en la Raspberry Pi y la preconfiguración de una serie de parámetros como el usuario, la contraseña, la red, el *ssh*, etc. Una vez configurados estos parámetros se pondrá en marcha la Raspberry Pi y se comenzará con la configuración.

Para la configuración de la protección del acceso a la Raspberry Pi se crearán los ficheros y carpetas necesarias para la configuración de las claves *ssh*. Se generarán las claves publicas y privadas, tanto de cliente como servidor y se intercambiarán entre los equi-

pos las claves necesarias. Habrá que asegurarse de que a los ficheros de configuración solo tienen acceso usuarios autorizados, ya que si un usuario no autorizado pudiera leer esos ficheros podría copiar las claves y acceder con permisos privilegiados. Finalmente, se cambiará la configuración de *ssh* para desactivar el acceso por contraseña.

A continuación, se instalará *Wireguard* como *VPN* creando las carpetas y ficheros necesarios para realizar la configuración. Siguiendo un proceso parecido al de *ssh* se generarán las claves publicas y privadas tanto de cliente como de servidor y se crearán los ficheros de configuración de ambos extremos. En el servidor de *VPN* ubicado en la Raspberry se añadirán las claves publicas de los clientes que se vayan a conectar a la *VPN* y la clave privada del servidor. En el lado de los clientes, el fichero de configuración contendrá la clave privada del cliente y la clave publica del servidor para así realizar el intercambio de mensajes. También se configurarán otros aspectos en el servidor como el rango de la subred de la *VPN*.Para limitar el acceso por *ssh* se añadirán reglas de *firewall* para limitar el acceso de la conexión *ssh* únicamente para usuarios conectados a través de la *VPN*.

Una vez tomadas las medidas de seguridad de acceso a la placa habrá que conectarse a la misma para descargar la plataforma de integración de *Home Assistant*. Para realizar esto, se deberá instalar primero *Docker*, ya que se requiere del lanzamiento de un contenedor que contenga el *Home Assistant*. Se seguirá el proceso de descarga de *Docker* y se comprobará que la descarga sea correcta mediante la ejecución de un "Hello World". Una vez descargado *Docker* correctamente se creará el fichero de configuración del contenedor de *Home Assistant* en el que se indicará la imagen a descargar, donde se almacenarán los ficheros de configuración y como se configurará la red. También se generarán los certificados *ssl* para la protección de la conexión con el servidor de *Home Assistant* y se añadirá al fichero de configuración, aunque este apartado se podrá realizar con posterioridad.

Una vez finalizada la configuración se lanzará el *Home Assitant* mediante un *Docker compose up* y se comenzará la configuración del mismo. Se añadirá un dispositivo para realizar las posteriores configuraciones de envió de mensajes y cifrado. Para limitar el acceso a la interfaz web se establecerán reglas de *firewall* para que únicamente los usuarios conectados a través de la *VPN* puedan visualizarla.

Se usará *Mosquitto MQTT* para realizar envíos de datos a los dispositivos domóticos y se cifrarán los envíos. Se explorarán distintas reglas de *firewall* y configuración para aislar las comunicaciones y evitar ataques.

3.4.5. Plan de implantación

Dado que en la configuración han sido descritas muchas de las medidas a tomar en la implantación del sistema, esta sección servirá como complemento de la sección previamente realizada. Se describirán las recomendaciones en la implantación del sistema del hogar, se realizará una ampliación de como añadir nuevos dispositivos y se explicarán alternativas al control web, como el uso de la aplicación móvil de *Home Assistant*.

Además de todo lo anterior se propondrán mejoras y agregados, como agregar procesadores de voz como alexa al sistema para utilizar comandos de voz pero sin perder control total del sistema, exponiendo solamente los dispositivos que se quieran exponer.

3.4.6. Pruebas de funcionamiento

Por último, se realizarán pruebas de funcionamiento, tanto del sistema domótico para el control del dispositivo, como en el cifrado y en la autenticación. Para estas pruebas se utilizará el programa *wireshark*, utilizado durante la carrera, para analizar los paquetes

de la red dirigidos a la Raspberry y comprobar que la información no es visible en plano y que, por tanto, esta cifrada por las distintas capas. Además se tratará de acceder por *ssh* desde un dispositivo no autorizado a la Raspberry para comprobar que el acceso es seguro. También se analizará el tráfico desde la Raspberry a los dispositivos de domótica para verificar el cifrado.

CAPÍTULO 4
Diseño de la solución

Tras identificar los requisitos del sistema, se procederá al análisis de los mismos para acercarnos a la solución final. Con esto se pretende pasar de la idea conceptual a la aplicación práctica del proyecto y el desarrollo del diseño del sistema final. Este capítulo se dividirá en dos secciones. La primera sección será un visión más general del proyecto en el que se ilustrarán como encajan las distintas piezas en el sistema. En la segunda se pasará a un diseño más detallado del sistema explicando parte por parte los elementos que lo constituirán.

4.1 Arquitectura del Sistema

4.1.1. Arquitectura general

El sistema domótico propuesto está basado en una arquitectura cliente-servidor distribuida, donde la Raspberry Pi actúa como servidor central y los dispositivos domóticos como clientes. La comunicación entre estos elementos se realiza a través de una red local y una *VPN* para el acceso remoto seguro a la interfaz del controlador de dispositivos. El envió cifrado de mensajes a los dispositivos se hace usando el *broker* de mensajería *MQTT*. Se puede observar la representación del sistema en la figura 4.1.

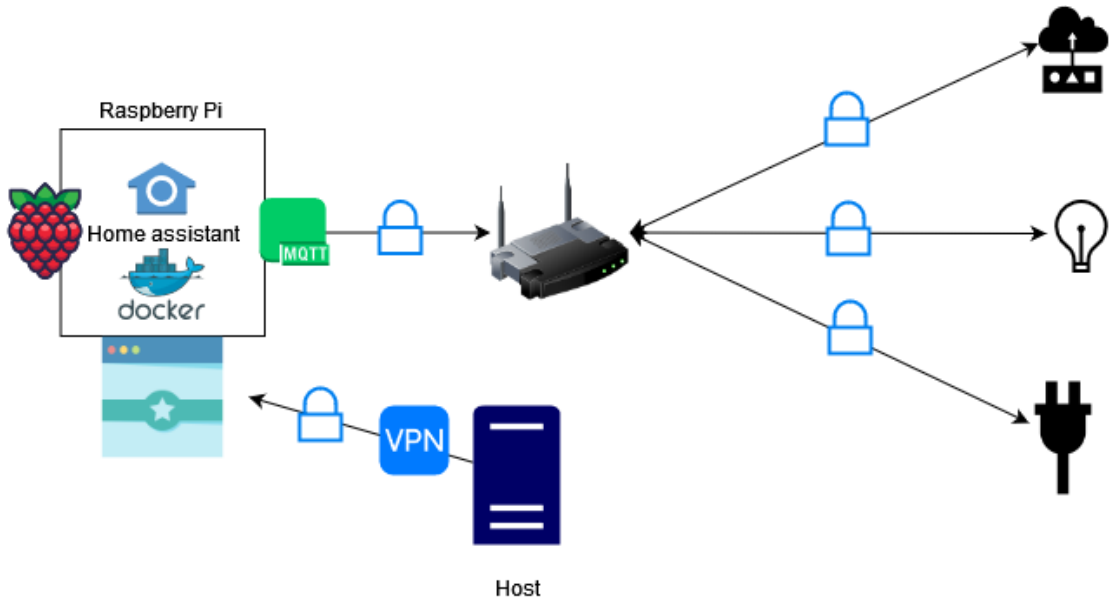

Figura 4.1: Esquema del sistema

Para desarrollar más en profundidad la arquitectura general se van a explicar dos apartados sobre la misma: las capas funcionales en las que se divide el sistema y el funcionamiento del flujo de datos.

Capas funcionales

Para realizar la subdivisión en capas funcionales se ha elegido adaptar a las necesidades del sistema el modelo de tres capas común en ingeniería informática (capa de presentación, capa de lógica de negocio y capa de datos) . En el caso de este sistema, al utilizar *Home Assistant* se desdibujan las lineas que diferencian la gestión de los datos y la gestión de la lógica por tanto a nivel del sistema ambas capas o niveles estarán representados como uno solo. Por tanto, la subdivisión será la siguiente:

- Presentación y acceso seguro: Esta capa la constituyen tanto los protocolos y medidas de acceso a la placa, como la interfaz gráfica de *Home Assistant*. Este nivel es el nivel con el que comúnmente interaccionará el usuario una vez finalizado el proyecto y establecidas todas las configuraciones. El objetivo de este nivel es el de conseguir un acceso seguro a la placa de usuarios autenticados, para que así estos puedan gestionar y realizar cambios en la configuración del sistema domótico.
- Lógica de negocio y datos: En esta capa se constituye la lógica detrás de los envíos y recepción de información cifrada a los distintos dispositivos. Además de esto *Home Assitant* también puede almacenar ciertos datos, ya sean de dispositivos de toma de medidas, como de encendidos o apagados.

Podemos observar un esquema de las diferentes capas en la figura 4.2.

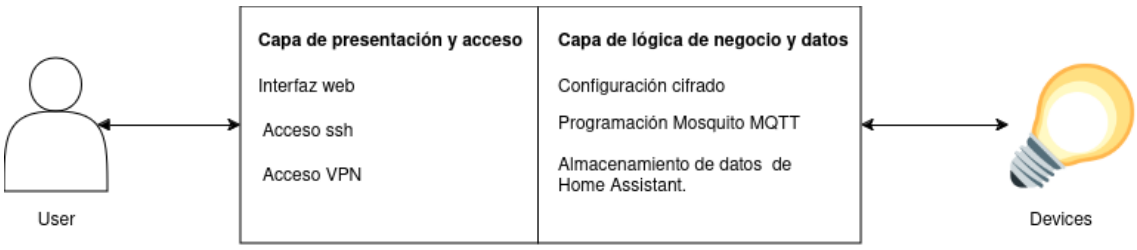

Figura 4.2: Esquema capas

Flujo de datos

El flujo que seguirán los datos en el sistema será el siguiente dependiendo del caso:

Caso de control de los dispositivos: En caso del control de dispositivos el usuario ingresará a la interfaz web o mediante la aplicación de *Home Assistant*, habiendo ingresado previamente en la *vpn*, y enviará la señal de control la cual irá cifrada a través *MQTT*.

Caso de recepción de datos: En caso de que se pretendan recibir los datos, estos serán enviados desde los dispositivos hacia la Raspberry y solo podrán ser accedidos por el usuario autenticado.

4.2 Diseño Detallado

En este segundo nivel de diseño se explicarán los distintos componentes, el flujo de trabajo y las consideraciones de seguridad.

Componentes del sistema (Ver figura 4.3):

- Raspberry Pi: Es el dispositivo hardware que actua como pieza central del sistema, hace posible las conexiones con los dispositivos y en él se ejecutan la mayor parte de los componentes del sistema.
- Sensores/Dispositivos: Son el elementos pasivos en el sistema, los dispositivos reciben las comunicaciones de la placa y son configurados para ser parte del sistema.
- Home Assistant (UI): Es la plataforma que actuará como interfaz de usuario para el control de los dispositivos, permite al usuario controlar los dispositivos y además permite tener una visión del sistema y monitorizar los dispositivos.
- Mosquitto MQTT: Es el *broker* de mensajería que se utilizará para controlar ciertos dispositivos los cuales se quiera añadir un cifrado extra y tener un control mayor sobre ellos.

Consideraciones de seguridad:

- Acceso SSH: se habilita el acceso *ssh* mediante el uso de *fingerprint* para reducir el riesgo de accesos no autorizados.
- VPN: se utiliza una *VPN* para el acceso de forma segura al *Home Assistant* y al *Mosquitto MQTT*, creándose la posibilidad de acceder de forma externa (desde fuera del hogar) al sistema y añadiendo un cifrado extra.
- Control de acceso: se tendrán en cuenta los permisos que se asignan a las carpetas y usuarios.

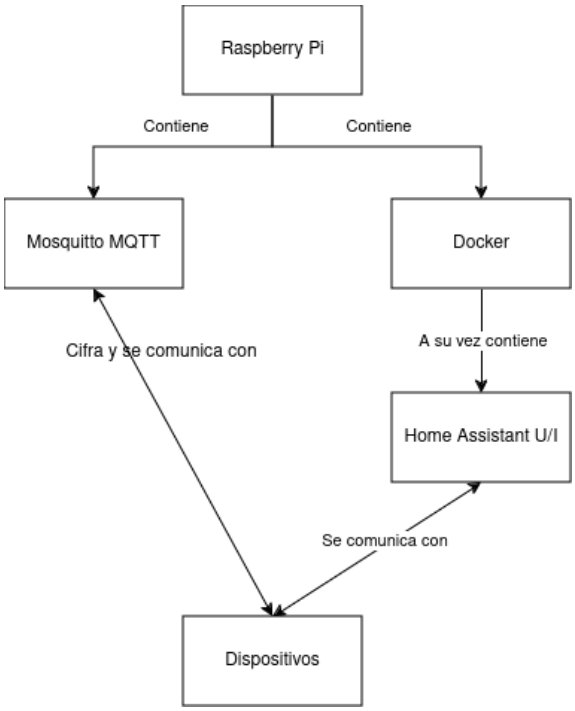

Figura 4.3: Diagrama del sistema

4.3 Tecnología Utilizada

Este apartado se enfocará en describir las tecnologías clave para el desarrollo del sistema.

4.3.1. Hardware

- Raspberry Pi: La placa base en la cual se ejecuta el software y en la que se conectan el resto de elementos del sistema.
- Router: Será una pieza necesaria para la interconexión de los dispositivos del hogar aunque no se requerirán configuraciones complejas en el dispositivo.
- Dispositivos domóticos: No tendrán unas características concretas ya que la tecnología y los protocolos pueden variar, pero se deberá investigar como compatibilizarlos con el sistema.

4.3.2. Software

- Raspbian: Es el sistema operativo más optimizado para la Raspberry Pi, lo cual genera un entorno estable y eficiente para la ejecución de aplicaciones.
- MQTT: Es un *broker* de mensajería que ofrece comunicación entre dispositivos y la Raspberry Pi.
- Home Assistant: Plataforma domótica de código abierto que gestiona los dispositivos y ofrece una interfaz de usuario.
- Ssh: Protocolo de "secure shell" para acceder de forma remota a la Raspberry Pi y, con ellos, poder admiistrar el sistema.
- WireGuard VPN: Es una red privada virtual que sirve para establecer una conexión segura y cifrada entre dispositivos remotos y la red doméstica.
- Open SSL: Herramienta de código abierto para la criptografía y seguridad de la información.

CAPÍTULO 5
Contexto y antecedentes

En este capítulo se explicará y analizarán los elementos ya existentes en el hogar, para dar contexto y así poder ver las diferencias tanto a nivel de tráfico como a nivel de integración.

5.1 Dispositivos y aplicaciones utilizadas

La elección de los dispositivos del hogar es algo que se debe hacer de forma meticulosa, ya que elegir dispositivos de marcas muy dispares puede provocar que requieras del uso de múltiples aplicaciones. Este mi caso, por lo que requiero de distintas aplicaciones para controlar los distintos dispositivos presentes en el hogar, lo cual puede ser muy tedioso sobre todo cuando hay algún problema en la red, lo cual provoca que haya que vincular los dispositivos de formas distintas dependiendo de la aplicación que controle el mismo.

Para los análisis presentes en el capítulo se limitará el análisis al enchufe *teckin* de la marca Tuya (Ver figura 5.1). Este será el dispositivo que se utilice como ejemplo de configuración una vez construido el sistema domótico.

Figura 5.1: Enchufe usado para el proyecto.

En cuanto a las aplicaciones, como ya se ha mencionado, son muchas las requeridas para configurar los dispositivos. Estas aplicaciones son: "Xiaomi Home", "Gosund", "Govee Home" y, la que se utiliza para controlar el enchufe, "Tuya".

5.2 Análisis del tráfico

En esta sección se analizará el tráfico para posteriormente tener un contraste del tráfico entre la conexión mediante el uso de la aplicación de Tuya, la conexión mediante *home assistant* y la conexión mediante el uso de colas de mensajería.

Para capturar el tráfico de una mejor forma se ha utilizado la web de *developers* de Tuya para poder capturar el envió de paquetes de encendido y apagado. La captura se realizará con el envió de un mensaje de "apagado" al enchufe. Capturando la petición, mediante el uso del analizador de red del navegador, se puede observar que se realiza un *post* al servidor web, el cual es el que se comunicará con el dispositivo. La petición es la siguiente.

```
POST /v1.0/devices/bf15f54ba65a95d1f1kpx9/commands HTTP/2
Host: openapi.tuyaeu.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:127.0) Gecko/20100101
    Firefox/127.0
Accept: application/json, text/plain, */*
Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate, br, zstd
Content-Type: application/json; charset=utf-8
Content-Length: 81
Referer: https://eu.platform.tuya.com/
```

De la petición también se puede analizar que la conexión se realiza al *host openapi.tuyaeu.com*. El contenido que se envía es un *json*, como pone en la linea de *content-type*. Con el analizador también se puede extraer el objeto *json* que se envía, el cual es el siguiente:

```
{
    "commands":[
        {
            "code":"countdown_1",
            "value":0
        },
        {
            "code":"switch_1",
            "value":false
        }
    ]
}
```

Este *json* le indica al servidor que tiene que poner a *false* el *switch*, es decir, apagar el interruptor.

En cuanto a un análisis del tráfico capturado con la herramienta *wireshark* se puede ver el tráfico de la figura 5.2. En esta captura se puede ver que el tráfico se intercambia en todo momento con el servidor de Tuya (*IP 52.81.66.10*) y el PC (*IP 192.168.1.38*) pero en ningún momento se intercambian mensajes directamente con el dispositivo.

61	1.219702	192.168.1.1	192.168.1.38	DNS	198 Standard query response 0x0129 AAAA out-kong-89f3d345e4401982.elb.eu-central-1.amazonaws.com S
62	1.219702	18.156.61.1	192.168.1.38	TLSv1.2	573 Application Data
63	1.263993	192.168.1.38	18.156.61.1	TCP	54 51470 → 443 [ACK] Seq=385 Ack=1062 Win=508 Len=0
64	1.524442	192.168.1.38	52.84.66.10	TLSv1.2	311 Application Data
65	1.545295	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=1 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
66	1.545295	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=1441 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
67	1.545295	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [PSH, ACK] Seq=2881 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
68	1.545295	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=4321 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
69	1.545295	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=5761 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
70	1.545295	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [PSH, ACK] Seq=7201 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
71	1.545426	192.168.1.38	52.84.66.10	TCP	54 49982 → 443 [ACK] Seq=258 Ack=8641 Win=4140 Len=0
72	1.548134	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=8641 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
73	1.548134	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=10081 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
74	1.548134	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [PSH, ACK] Seq=11521 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
75	1.548134	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=12961 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
76	1.548134	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=14401 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
77	1.548134	52.84.66.10	192.168.1.38	TLSv1.2	1494 Application Data
78	1.548134	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=17281 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
79	1.548134	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=18721 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
80	1.548134	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [PSH, ACK] Seq=20161 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
81	1.548276	192.168.1.38	52.84.66.10	TCP	54 49982 → 443 [ACK] Seq=258 Ack=21601 Win=4140 Len=0
82	1.552421	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=21601 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]
83	1.552421	52.84.66.10	192.168.1.38	TCP	1494 443 → 49982 [ACK] Seq=23041 Ack=258 Win=265 Len=1440 [TCP segment of a reassembled PDU]

Figura 5.2: Tráfico de red Tuya con la app.

Buscando la *IP* para ver a quien pertenece utilizando la página *who is* se puede ver que la *IP* pertenece a *Amazon*, (Ver figura 5.3) lo cual es coherente, ya que Tuya utiliza los servidores de *Amazon*. Esto último se puede ver analizando más a profundidad los paquetes capturados, concretamente los paquetes de respuesta DNS donde se puede ver más detalles del servidor con el que se comunica el pc y, efectivamente, se observa que utiliza *amazonaws*:

```
1 openapi.tuyaeu.com: type CNAME, class IN, cname out-kong-89f3d345e4401982.elb.
    eu-central-1.amazonaws.com
```

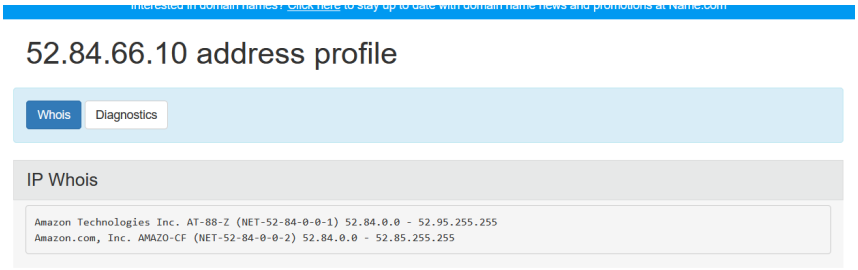

Figura 5.3: Captura de la página who is

Se puede deducir también de la captura que los paquetes que utilizan el protocolo *TLS* son los que contienen el envió de apagado y el resto son intercambios de información con el servidor para ver el resto de elementos de la web de Tuya. Los paquetes *TLS* tienen como dirección destino/origen la dirección *18.156.61.1* la cual también pertenece a *Amazon*.

En conclusión, la comunicación con el dispositivo se hace mediante el uso de un tercero, el servidor de Tuya, y nunca directamente con el dispositivo. En el proyecto, mediante el uso de colas de mensajería, se intentará evitar el uso de este servidor.

CAPÍTULO 6
Desarrollo de la solución

Se comenzará el desarrollo de la solución instalando el sistema operativo de la Raspberry Pi, en caso del proyecto se ha utilizado la Raspberry Pi 4 Modelo B (4GB), pero se podría llevar a cabo con otros modelos de menor capacidad. El primer elemento a tener en cuenta a la hora de descargar el sistema operativo es decidir si se quiere usar un sistema operativo dedicado por completo a la domótica del hogar, como es el *home assistant*[7], o trabajar sobre el sistema operativo de Raspberry Pi OS. Tras valorar ambas alternativas, la opción elegida ha sido la de instalar el sistema operativo de Raspberry Pi ya que no limita el uso de la Raspberry a la domótica. Al elegir como sistema operativo Raspberry Pi OS se debe escoger la forma en la que se lanzará el *Home Assistant* y la opción que se usará en el proyecto será la configurar el *Home assistant* mediante el uso de contenedores, como será detallado más adelante.

6.1 Instalación del sistema operativo

Para la instalación del sistema operativo habrá que ir al apartado de sistemas operativos dentro del sitio web oficial de Raspberry Pi [15]. Entre las diversas opciones de descarga se usará el *Raspberry Pi Imager* ya que facilitará la tarea de instalar la imagen del sistema operativo. Una vez descargado, se ejecutará el archivo y aparecerá un menú como el de la figura 6.1.

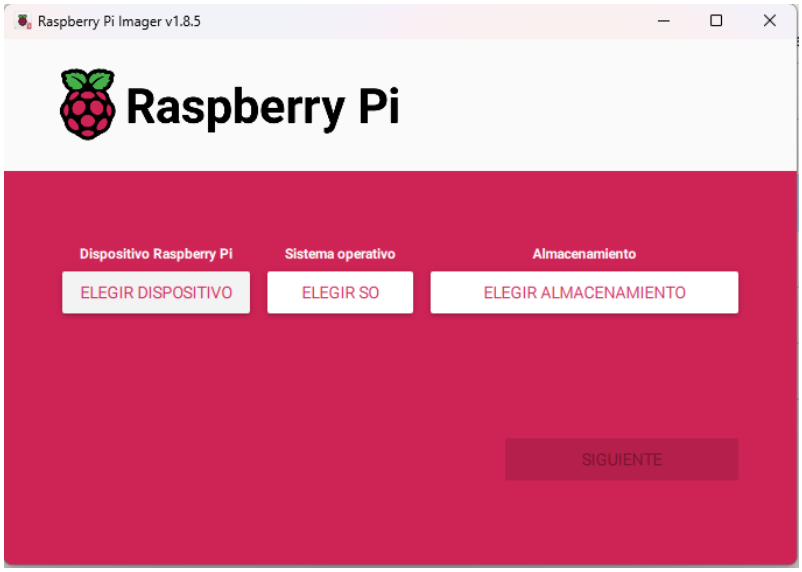

Figura 6.1: Menú *Raspberry Pi imager*

En este menú, se escogerá la opción de elegir dispositivos y se buscará el modelo de la placa, en caso de este proyecto la placa seleccionada será la Raspberry Pi 4 Modelo B (4GB) como en la figura 6.2.

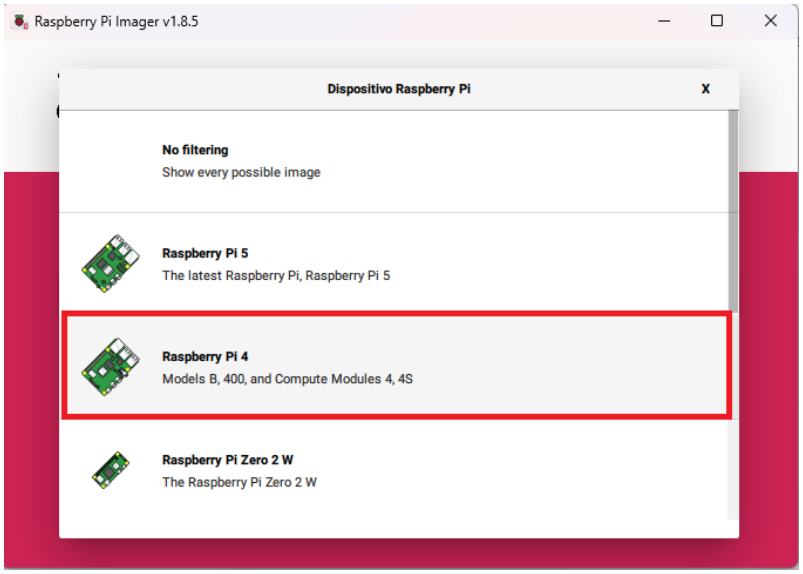

Figura 6.2: Selección de la placa

En cuanto a la elección del sistema operativo, se escogerá la opción de 64 bits (figura 6.3) ya que tiene significativas ventajas frente a la opción de 32 bits. Algunas de estas ventajas son una mejor gestión del espacio de la ram y una mayor seguridad debido a las técnicas que se usan con 64 bits.

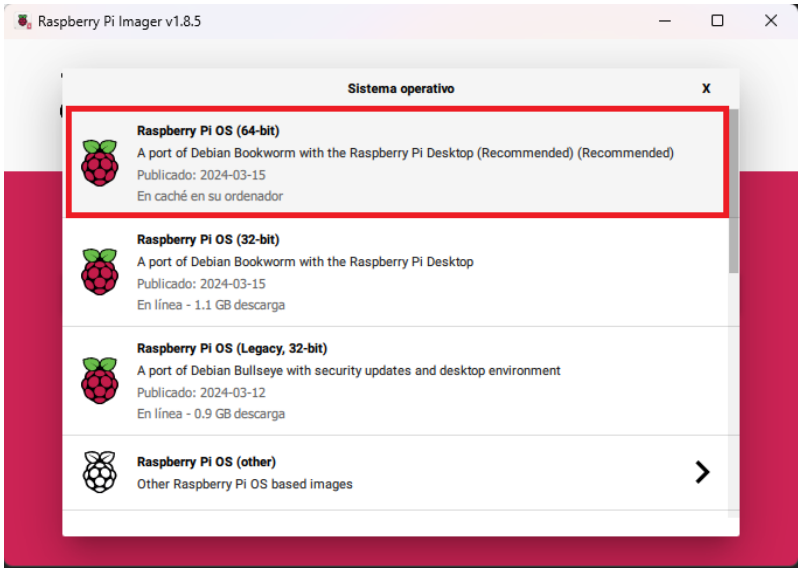

Figura 6.3: Selección del SO

Finalmente se escogerá como almacenamiento la tarjeta sd, que será la memoria secundaria de la Raspberry Pi. También es posible preconfigurar una serie de parámetros como el nombre de usuario, la contraseña o la conexión wifi. El menú de preconfiguración puede ser accedido mediante el atajo de comandos control + shift + x. Para ahorrar tiempo se preconfigurarán una serie de parámetros, como son el nombre de usuario, contraseña, conexión a internet y el acceso mediante *ssh*. Una vez introducidos estos pa-

rámetros, se pulsará el botón de "siguiente" y se esperará a que el *imager* prepare la sd. Cuando el proceso finalice se introducirá la tarjeta sd en la Raspberry.

A partir de este punto, se realizará la configuración de la placa mediante *ssh* aunque se podría hacer desde la consola del SO de la Raspberry.

6.2 Protección del acceso ssh

En primer lugar, se requiere averiguar la dirección IP de la Raspberry para conectar con la placa en remoto. Para ello, existen distintas alternativas:

1. Usar la interfaz gráfica de Raspberry Pi y usar "*ifconfig*" o "*ip a*" para averiguarla mediante la visualización de las interfaces de red. También ser podría averigurar lo mismo desde la configuración de la red de la interfaz gráfica.

2. Conectar con el *router* mediante la interfaz web y ver los dispositivos conectados.

3. Usar herramientas de terceros como "*IP Scanner*".

Una vez averiguada la IP se podrá acceder a la placa por *ssh* usando el usuario y la contraseña que han sido preconfiguradas al instalar la imagen. Una vez establecida la conexión se comenzará el proceso de protección del inicio de sesión mediante el uso de *fingerprint*.

6.2.1. Configuración de un nombre para el servicio

Como punto adicional es posible agregar un nombre de host a la IP bien sea en el fichero host del equipo local o en el *dns* del *router* (aunque esto conlleva más riesgos). Esto es para una mayor comodidad y no tener que memorizar la IP del dispositivo. Para realizar esto tendremos que:

1. En windows: abrir el bloc de notas con permisos de administrador y añadir al fichero de host en la ruta `C:\Windows\System32\drivers\etc` la linea: "IP de la raspberry (espacio) nombre". p. ej 192.168.1.21 rasp.

2. En linux: modificar el fichero `/etc/host` con permisos de administrador y añadir una linea con el siguiente formato: "IP de la raspberry (espacio) nombre". Como en la figura 6.4.

```
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host
# localhost name resolution is handled within DNS itself.
#       127.0.0.1       localhost
#       ::1             localhost
127.0.0.1 checkhost.local
192.168.1.21 rasp
10.0.0.1 raspvpn
```

Figura 6.4: Fichero host linux

3. En el router: conectarse a través del navegador con la ip del *router*, normalmente 192.168.1.1. Una vez conectado ir al apartado *DNS* y añadir una entrada como en la figura 6.5.

Figura 6.5: Añadir entrada al DNS

6.2.2. Uso de claves para ssh

El uso del protocolo *ssh* requiere de la exposición de un puerto de entrada a la máquina, el cual puede ser explotado por atacantes. Para prevenir que puedan entrar forzando el usuario y la contraseña de entrada, se dará uso de claves tanto en el servidor *ssh* establecido en la Raspberry, como en el cliente *ssh*, el cual se usa para la conexión con a la placa. Esto actuará como una especie de llave que poseerá el usuario para que el servidor lo identifique como usuario autorizado para acceder.

Se comenzará la configuración [16] con la creación de una carpeta ".ssh" en la ruta "/home/nombreUsuario" y el cambio de los permisos de la carpeta. Este cambio de permisos es muy relevante ya que hace que solo el usuario que la creó y el *root* puedan agregar o modificar claves.

```
1 # mkdir /home/gavigar/.ssh
2 # chmod 700 ~/.ssh
```

Se generarán las claves *ssh* con un cifrado *rsa* mediante el uso del comando *ssh-keygen*. Cuando se introduce el comando el sistema solicita una *passphrase* con la cual se generará la clave privada, la *passphrase* introducida deberá ser recordada. (Resultado de la ejecución en la figura 6.6).

```
gavigar@raspberrypi:~ $ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/gavigar/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/gavigar/.ssh/id_rsa
Your public key has been saved in /home/gavigar/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:mN22AEmJZ7zIN4CekxThJ+xVqwXkWa3OTprAW8h624k gavigar@raspberrypi
The key's randomart image is:
+---[RSA 3072]----+
|  o=oo+o         |
| oo.o*=o.        |
| o++===o         |
| .=+oo== .       |
| o.o.+o.S o      |
|  = . +  o .     |
| . + =    .      |
|. ooo..          |
| .E.o            |
+----[SHA256]-----+
```

Figura 6.6: Generación de clave con keygen

Es posible comprobar que las claves se han creado correctamente listando el contenido de la ruta `~/.ssh`.

Habrá que generar también una clave privada que usará el cliente para conectarse al servidor *ssh*. Para ello, se usa el comando *ssh-keygen -t rsa* y al igual que en caso anterior pedirá una *passphrase* para la generación de la clave. La *passphrase* introducida también deberá ser recordada, ya que podrá ser requerida para conectar por *ssh*. Aparecerá un resultado como el de la figura .6.7.

```
29/03/2024 11:35.09 /home/mobaxterm ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/mobaxterm/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/mobaxterm/.ssh/id_rsa
Your public key has been saved in /home/mobaxterm/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:3wrh7oDh4I6s/8y2VMhbueH0lYTSdGck4RPaPn7s0v8 guill@Kratos
The key's randomart image is:
+---[RSA 3072]----+
|         . =o+   |
|       o * =     |
|      . + =      |
|   . . o o o     |
|  . + * S =      |
| . o 0 = = +     |
|  . = + + o +    |
|.o +.  o ..+     |
|+ooo=. .o o+o.E  |
+----[SHA256]-----+
```

Figura 6.7: Generación clave privada en el cliente

Una vez generadas las claves, se deberá crear un fichero llamado "authorized_keys" para indicarle al servidor qué claves están autorizadas para acceder al servidor *ssh*. Al igual que con la carpeta *ssh*, habrá que cambiar los permisos para que solo el autor pueda editar y leer el fichero. El cambio de permisos aunque parezca trivial es fundamental para la seguridad del sistema, ya que si un usuario sin permisos pudiese leer o escribir en los ficheros de configuración podría ganar acceso a la placa con permisos privilegiados.

```
1 # touch ~/.ssh/authorized_keys
2 # chmod 600 ~/.ssh/authorized_keys
```

Desde el cliente se utilizará el comando *ssh-copy-id user@ip* para copiar la clave generada al servidor *ssh* de la Raspberry Pi. Este comando incluye la clave privada del cliente en el fichero de claves autorizadas. Una vez el cliente está autorizado habrá que desactivar la autenticación por contraseña editando el fichero *sshconfig.d*.

```
# sudo vi /etc/ssh/ssh_config.d
```

Dentro del fichero hay que cambiar el valor de la variable "PasswordAuthentication" de "yes" a "no", como en la figura 6.8.

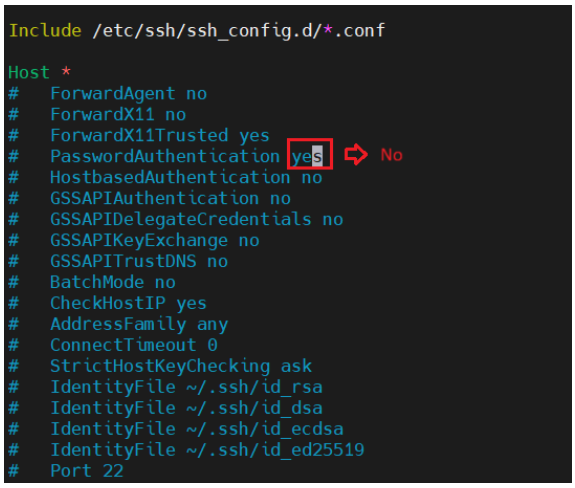

Figura 6.8: Desactivación contraseña *ssh*

Para que aplicar los cambios habrá que reiniciar el servicio:

```
# sudo systemctl restart ssh
```

Se pueden comprobar los resultados conectando desde el cliente configurado por *ssh* e introduciendo la *passphrase* (esto solo lo pedirá la primera vez y cada cierto tiempo).

Existe un fallo encontrado en las pruebas de funcionamiento, ver el capítulo de "Pruebas de funcionamiento" (8.2.1) para ver el fallo y su solución.

6.3 Configuración de una VPN

Otra medida para el acceso seguro a la placa, que es el centro de control del sistema domótico, es la instalación de la *VPN*. Algunas de las *VPN* más comunes son *OpenVPN* y *Wireguard*. *OpenVPN* ofrece más opciones de configuración de la *VPN* pero dado que no es el centro de este trabajo, se ha escogido *Wireguard* que es más sencilla de configurar y utilizar.

6.3.1. Instalación de Wireguard y configuración inicial

Para instalar *wireguard*, habrá que conectarse por *ssh*, actualizar los repositorios e instalar *Wireguard* de la siguiente forma:

```
# sudo apt update
# sudo apt install wireguard
```

En caso de que la instalación de *wireguard* no cree automáticamente la carpeta de "wireguard" habrá que ejecutar:

```
# sudo mkdir -p /etc/wireguard
```

Ahora generaremos claves siendo este proceso similar a lo realizado para autenticar *ssh*, pero en este caso servirá para cifrar las comunicaciones que pasen por el túnel. Para ello, habrá que generar las claves, publica y privada, en el servidor *VPN* y moverlas a la carpeta correspondiente de *wireguard* usando la orden:

```
# wg genkey | sudo tee /etc/wireguard/privatekey_server | wg pubkey | sudo tee
    /etc/wireguard/publickey_server.
```

En el lado del cliente habrá que hacer también la generación de claves para que el servidor autorice la conexión. La autorización se realiza mediante el guardado de la clave pública del cliente en el servidor *VPN* situado en la placa. De este modo el servidor podrá desencriptar los mensajes encriptados con la clave privada del cliente, y con ello, verificará que él es el que envía los mensajes. Se debe generar la clave siguiendo el modelo de los siguientes comandos, indicando la ruta donde se quieran guardar las claves:

```
# wg genkey | sudo tee rutadondeguardarlo/privatekey_client
# sudo wg pubkey < /rutadondequeramosguardarlaclave | sudo tee
    rutadondeguardarlo/publickey_client
```

En la carpeta de *wireguard* previamente creada crearemos la configuración de la interfaz de red de la *VPN*. Para ello, crearemos el fichero *wg0.conf* y le pondremos la configuración siguiendo la siguiente plantilla:

```
[Interface]
Address = 10.0.0.1/24      # Dirección IP del servidor
PrivateKey = <clave privada del servidor>   # Clave privada del servidor
ListenPort = 51820         # Puerto en el que escucha el servidor

# Configuración del primer cliente
[Peer]
PublicKey = <clave pública del cliente 1>   # Clave pública del cliente 1
AllowedIPs = 10.0.0.2/32  # Dirección IP del cliente 1

# Configuración del segundo cliente
[Peer]
PublicKey = <clave pública del cliente 2>   # Clave pública del cliente 2
AllowedIPs = 10.0.0.3/32  # Dirección IP del cliente 2

# Y así sucesivamente para cada cliente adicional
```

En el lado del cliente también se debe crear un fichero de configuración que se llamará *wg0-client.conf* en el que se incluirán los datos de la clave privada del cliente y la clave pública del servidor. Se seguirá el modelo de la siguiente plantilla:

```
[Interface]
Address = 10.0.0.2/24
PrivateKey = <clave privada del cliente>

[Peer]
PublicKey = <clave pública del servidor>
Endpoint = <IP pública del servidor>:51820
AllowedIPs = 0.0.0.0/0
```

En este caso si solo se quisiera la *VPN* para el cifrado desde la red interna únicamente se requeriría poner en IP pública del servidor la IP local de la Raspberry Pi. En el siguiente apartado se pondrán los pasos y configuraciones necesarias para poder acceder a la VPN desde fuera de la red.

Una vez finalizada la configuración se lanzará el servicio de la *VPN* con el comando:

```
sudo systemctl start wg-quick@wg0
```

6.3.2. Modificaciones para el acceso con la VPN desde fuera de la red

Para poder acceder a la *VPN* desde fuera de la red lo primero será entrar en la configuración del *router* y hacer una serie de modificaciones. Para poder acceder a la configuración del *router* se utilizará su *IP* por defecto que suele ser *192.168.1.1*, *192.168.0.1* o *10.0.0.1*.

Una vez accedido al *router* desde el navegador (estos apartados pueden variar en función del fabricante) se deberá ir al apartado configuración avanzada y posteriormente al apartado *DHCP*. En este apartado se añadirá una *IP* fija a la raspberry para que cuando se apague y encienda no varíe y por tanto pueda fallar el *port forwarding* que se realizará a continuación. Para añadir esta *IP* fija en la parte inferior se seleccionará el dispositivo, y se pondrá la *IP* que se le quiera asignar para que quede fija. Esto lo podemos ver en la figura 6.9

Figura 6.9: Asignar IP fija a la raspberry

A continuación, habrá que ir al apartado de NAS para realizar el *port forwarding*, es decir, el establecimiento de una correlación entre un puerto de la *IP* pública y un par *IP* local-puerto de la red local. Una vez en este apartado, en la parte inferior habrá una serie de campos a rellenar donde habrá que poner el protocolo, el puerto que se expondrá en el *router* y el puerto e *IP* del dispositivo de la red local al que se quiera redirigir el tráfico de ese puerto. En este caso el protocolo será "VPN", nombre introducido a mano ya que no se corresponde con ninguno de los valores por defecto, el puerto será en ambos casos el mismo el "51820" y la *IP* del dispositivo será la que se le haya asignado como fija. Se puede observar el resultado en la figura 6.10.

Figura 6.10: Port fordwarding a raspberry Pi

Por último, habrá que realizar una serie de configuraciones extra en la Raspberry Pi. En cuanto a los ficheros de configuración habrá que poner la IP pública del *router* como *IP* del servidor, esto también se puede hacer si te conectas desde la red local pero puede ocurrir un *hairpinning* que es que el *router* no redirija bien los paquetes al usar la *IP* pública en vez de la local.

En cuanto a otras configuraciones se deberá introducir o descomentar la siguiente linea: *"net.ipv4.ip_forward = 1"*en el fichero *sysctl.conf* para habilitar el reenvió. Y aplicar los cambios con:

```
# sudo sysctl -p
```

Además de esto habrá que asegurarse de que el puerto está permitido en el *firewall* mediante:

```
# sudo ufw allow 51820/udp
```

NOTA: Habrá que tener en cuenta que con cada reinicio del *router* la *IP* puede cambiar, por lo que habría que editar el fichero de configuración cada vez que cambie la misma. Esto podría solventarse pagando una *IP* fija para el router o estableciendo un DNS.

6.3.3. Instalación del software en el cliente

Desde el lado del cliente para conectar a la *vpn* se deberá instalar el software de *Wireguard* en el dispositivo y sistema operativo desde el cual se quiere establecer la conexión. Para la configuración en windows se deberá instalar el software de *wireguard* de su página web oficial.

Una vez descargado el fichero de instalación, se debe ejecutar y seguir el "wizard" de instalación. Al abrir el programa se puede observar la interfaz de la figura 6.11. Para agregar la *VPN*, hay que pulsar el botón "import" y seleccionar el fichero "wg0-client.conf".

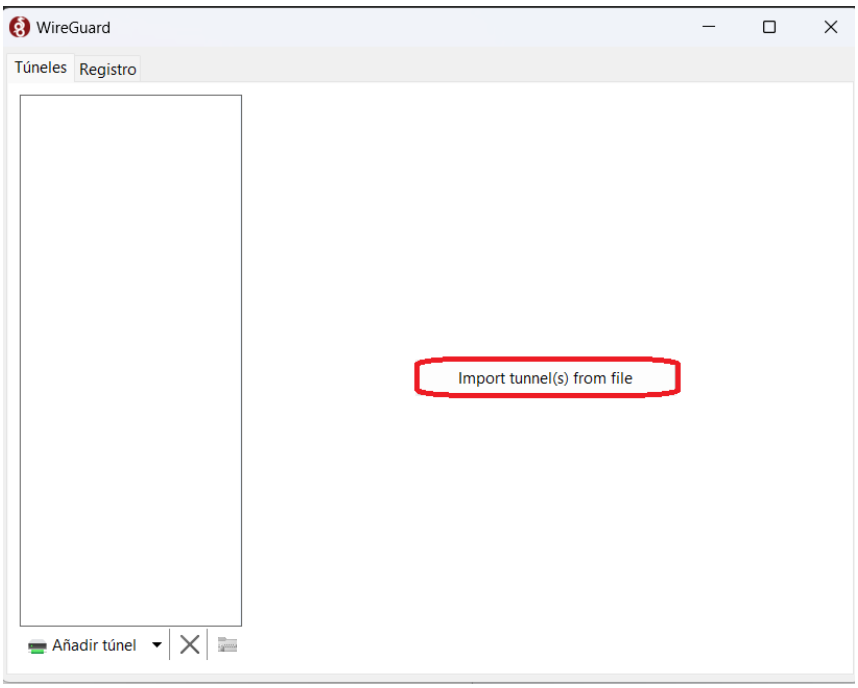

Figura 6.11: Inicio wireguard

Una vez importada la configuración de la *VPN* para activarla se debe seleccionar la configuración añadida y pulsar el botón "activar" como en la figura 6.12.

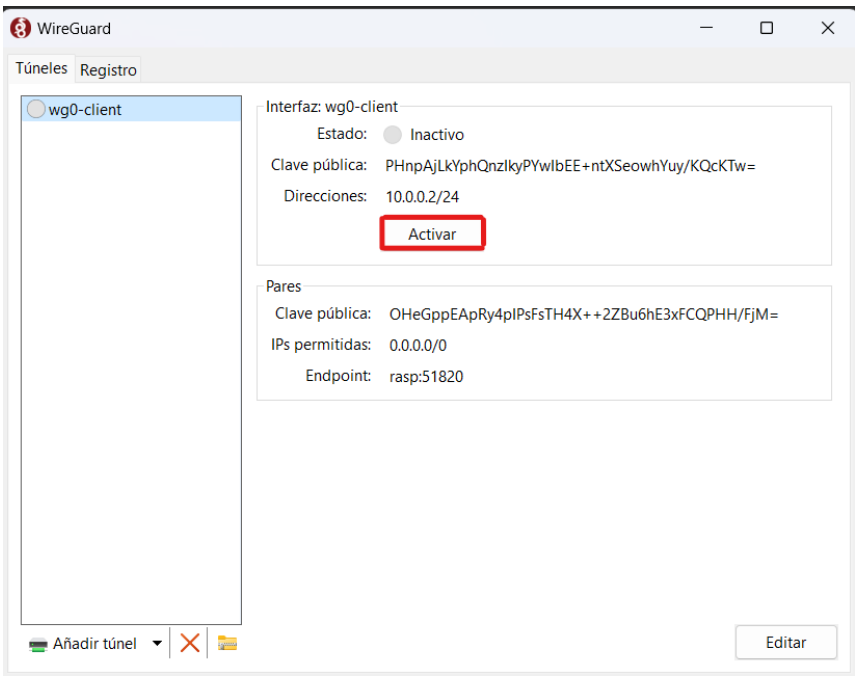

Figura 6.12: Activación VPN

A partir de este punto, la IP del dispositivo será la que haya sido asignada en el fichero de configuración. Para el acceso por *ssh* a la placa la IP también será distinta y esta será la configurada en la subred de la *VPN* (en este caso *10.0.0.1*). Si se quiere seguir resolviendo el nombre (DNS) o se quiere tener un nombre nuevo para esta nueva IP habrá que editar otra vez el fichero de host (Figura 6.4) o modificar el DNS.

6.4 Instalación y uso de Home Assistant

Como se ha mencionado al comienzo del desarrollo se ha elegido el uso del sistema operativo de Raspberry Pi. Por lo cual, para el uso de *Home Assistant* se requiere del uso de contenedores y concretamente del uso de *Docker*. Para la instalación y configuración de *Home Assistant* en *Docker* seguiremos la guía de *Home Assistant*.

6.4.1. Instalación y configuración de Docker

Tras realizar varias pruebas con la instalación de la Raspberry pi y, siguiendo la guía de *Docker* sobre la instalación del software en la Raspberry Pi, han surgido varios fallos. Para solucionarlos, se ha recurrido a una alternativa la cual consiste en la descarga de *docker* mediante un script de la página de *docker* oficial. Por tanto, se comenzará la instalación mediante una petición curl a la página de get.docker.com que devolverá un script de instalación el cual se le pasará directamente al *shell* mediante el uso de una pipe (|).

```
1 # curl -sSL https://get.docker.com | sh
```

Una vez ejecutado el *script* se deberá crear un grupo *docker* para el uso de los contenedores y se deberá reiniciar la Raspberry para que se apliquen los cambios:

```
1 # sudo usermod -aG docker $USER
2 # reboot
```

Una vez reiniciado se ejecutará el comando groups y con ello, se verificará que el grupo haya sido añadido correctamente (Ver figura 6.13).

```
gavigar@raspberrypi:~ $ groups
gavigar adm dialout cdrom sudo audio video plugdev games users input render netdev lpadmin docker gpio i2c spi
```

Figura 6.13: Grupos agregados

Para comprobar la correcta instalación de *docker* hay que ejecutar *docker run hello-world*. Que es un ejemplo de lanzamiento de un *docker* de inicio ya creado. El resultado de la salida será como el de la figura 6.14.

```
gavigar@raspberrypi:~ $ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (arm64v8)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Figura 6.14: Hello World Docker

6.4.2. Preparación del contenedor de Home Assitant

Para preparar el contenedor de *Home assístant* hay que que crear un fichero de configuración yml que se usará para establecer los parámetros de lanzamiento de *docker*. Para ello, se deberá crear un directorio *docker* y dentro de este un fichero yml y un subdirectorio *docker_config*.

```
# mkdir docker
# touch docker/homeassitant.yml
# mkdir docker/config
```

Seguidamente, habrá que abrir el fichero de configuración con cualquier editor de texto y escribir la configuración (Ver figura 6.15). En la configuración se comenzará poniendo la versión en caso de que queramos llevar un seguimiento de los distintos cambios y versiones que se vayan realizando sobre el contenedor. Se continuará poniendo el nombre del contenedor, que en este caso se llamará "homeassistant"; una imagen la cual será la del link proporcionado por la guía de *Home assistant*; se establecerán unos volumenes y se pondrán las políticas de no reinicio a no ser que se detenga el servicio y que tenga permisos privilegiados dentro del contenedor (root); y, por último, se establecerá que use la red del host en vez de una propia interna. Los puntos a destacar de esta configuración son que los volumenes dan persistencia a la configuración, lo que hará que cuando se reinicie el contenedor no se pierdan los datos, y que se use la red del host, lo que permite que se pueda cargar la web de Home Assitant desde fuera de la Raspberry Pi.

```yaml
version: '3'
services:
  homeassistant:
    container_name: homeassistant
    image: "ghcr.io/home-assistant/home-assistant:stable"
    volumes:
      - /home/gavigar/docker/config:/config
      - /etc/localtime:/etc/localtime:ro
      - /run/dbus:/run/dbus:ro
    restart: unless-stopped
    privileged: true
    network_mode: host
```

Figura 6.15: Fichero yml de configuración

Para lanzar el contenedor se debe ejecutar: *docker compose up -d*. La primera vez que sea ejecutado, *docker* deberá descargar una serie de ficheros ya que no los tiene en local (Ver figura 6.16).

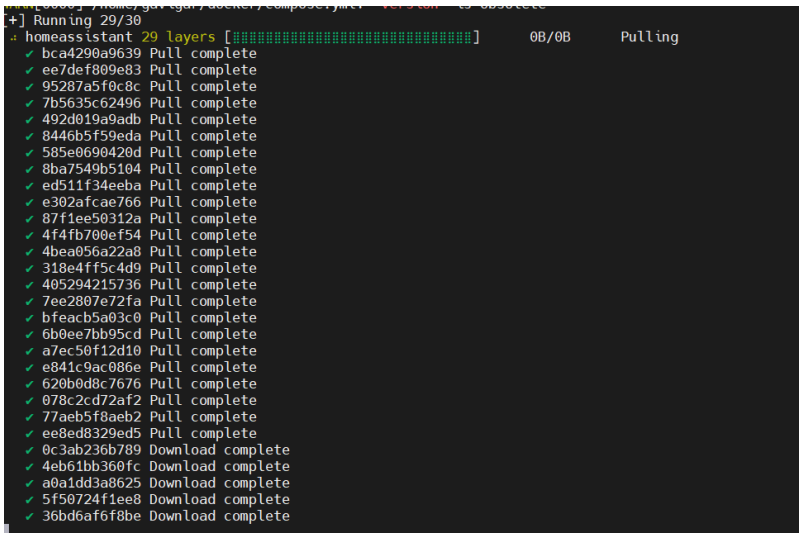

Figura 6.16: Fichero descargados por Home Assitant

6.4.3. Añadiendo seguridad mediante SSL

Problemas de la primera solución

En esta sección primero se explicará el primer intento de configuración el cual no conseguia que el uso de certificados funcionará y el proceso que llevo a la correcta configuración del mismo. Para la solución que inicialmente se planteaba, se realizará una modificación extra al *docker* para añadir un certificado ssl al servidor de *home assistant* y con ello tener la conexión entre la interfaz web y el host cifrada. Para realizar esto, se comenzará generando el certificado ssl y creando una carpeta (dentro de la carpeta *docker*) donde irán los certificados:

```
# mkdir homeassistant-certs
# cd homeassistant-certs
# openssl genrsa -out homeassistant.key 4096
# openssl req -new -key homeassistant.key -out homeassistant.csr
# openssl x509 -req -in homeassistant.csr -signkey homeassistant.key -out
    homeassistant.crt -days 3650
```

La explicación del proceso seguido es la siguiente: Se comienza generando una clave privada *.key* a partir de la cual se genera la solicitud del certificado *.crt*. Al lanzar este segundo comando se preguntarán ciertos parámetros para completar el certificado como podemos ver en la figura 6.17. Finalmente, ambos elementos se utilizarán para generar el certificado autofirmado, indicando como último parámetro el número de días de validez del certificado.

```
gavigar@raspberrypi:~/docker/homeassistant-certs $ openssl req -new -key homeassistant.key -out homeassistant.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
String too long, must be at most 2 bytes long
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:Valencia
Locality Name (eg, city) []:Valencia
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Home
Organizational Unit Name (eg, section) []:Home
Common Name (e.g. server FQDN or YOUR name) []:Home
Email Address []:home@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:raspberry
An optional company name []:home
```

Figura 6.17: Generar certificado

Solución final

Tras investigar distintas soluciones se ha llegado a la utilización de la consola del *docker*[17]. Para acceder a ella, realizamos un *docker ps* para averiguar el id de nuestro contenedor. Una vez realizado esto ejecutamos el siguiente comando para abrir un bash con permisos de root:

```
# docker exec -it --user=root {dockerID} bash
```

Una vez dentro se actualizarán los paquetes e instalarán *openssl*:

```
# apk update
# apk updgrade
# apk add openssl
```

Con *openssl* instalado habrá que ir a la carpeta raíz y lanzar las siguientes ordenes:

```
# mkdir ssl
# openssl req -new -x509 -days 3650 -nodes -keyout ssl/hass.key -out ssl/hass.
    crt
```

Con esto ya se tendrán los certificados en la ruta y ahora habrá que modificar el *yaml* de configuración, haciendo un *backup* por si hay error:

```
# cp config/configuration.yaml config/configuration.bak
# openssl req -new -x509 -days 3650 -nodes -keyout ssl/hass.key -out ssl/hass.
    crt
# vi config/configuration.yaml
```

Y se añaden las siguientes lineas al final del fichero *configuration.yaml*:

```
http:
    ssl_certificate: /ssl/hass.crt
    ssl_key: /ssl/hass.key
```

Finalmente, habrá que reiniciar el contenedor mediante un *docker compose restart* y volver a la interfaz web para comprobar que se ha realizado correctamente. En la figura 6.18 veremos el mensaje que aparecerá al entrar en la página.

En las pruebas se podrá observar a través de los paquetes que el contenido va cifrado mendiante https. Para comprobar la persistencia del *docker*, se puede realizar un *docker compose up -d* lo cual relanzará el *docker* y se podrá observar que persiste la conexión segura.

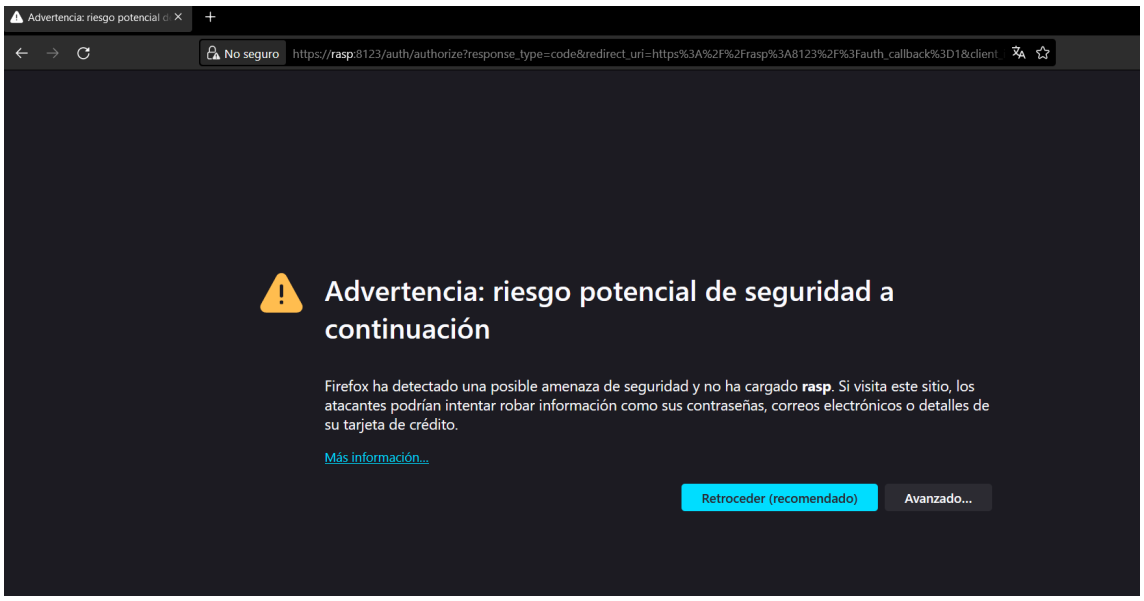

Figura 6.18: Página con certificado

6.4.4. Configuración de Home Assistant

Una vez lanzado el contenedor se podrá acceder al Home Assitant usando un navegador a través de la ip de la Raspberry (o bien el nombre de host) y el puerto por defecto de *Home Assistant*, que es el 8123. Aparecerá una interfaz como la de la figura 6.19.

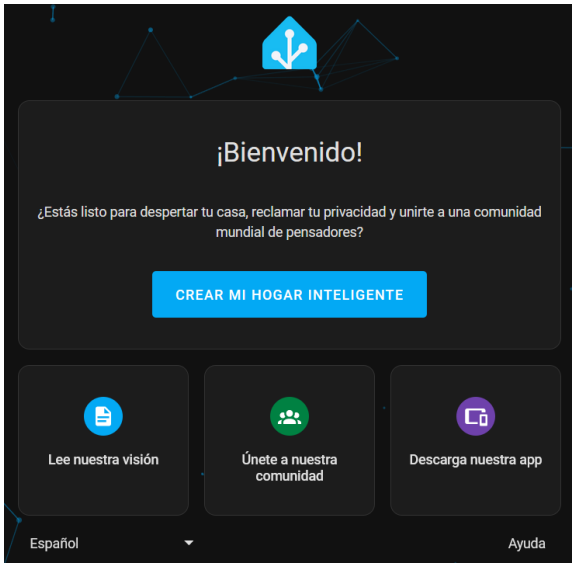

Figura 6.19: Pestaña de inicio Home Assitant

Para empezar la configuración se pulsará en crear mi hogar inteligente y se rellenarán los campos que se solicitan como son: nombre, nombre de usuario y contraseña. Se establecerá la ubicación del hogar donde estará establecido el sistema si se quiere recibir datos del clima local. Una vez completados todos estos campos se llegará al menú de inicio (Ver figura 6.20).

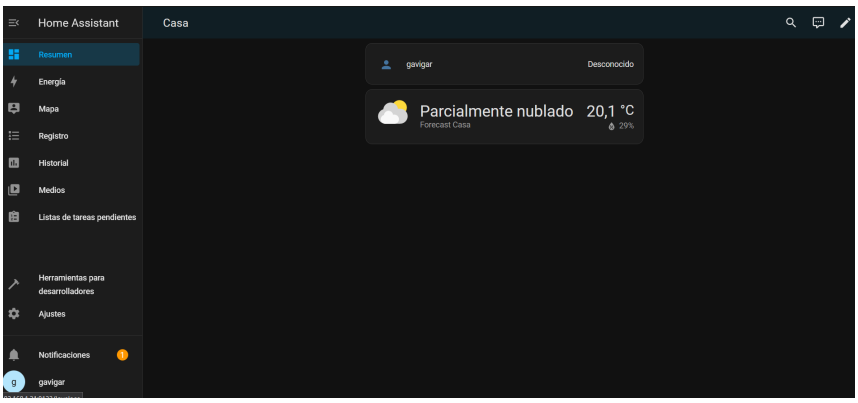

Figura 6.20: Menú inicio *Home assistant*

6.4.5. Agregar dispositivos

Una vez desplegado el *home assistant* se podrán agregar dispositivos para poder controlarlos y monitorizarlos. Para ello, se debe ir al apartado ajustes, pulsar en "dispositivos" y, finalmente, pulsar en el botón de "agregar dispositivo" de la parte inferior derecha. Estos pasos pueden ser observados en la figura 6.21.

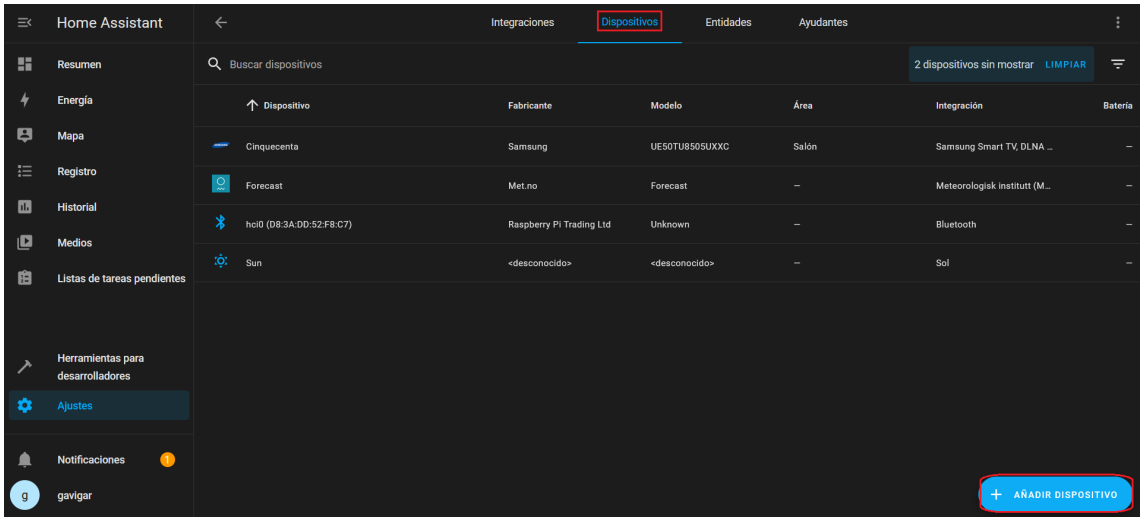

Figura 6.21: Añadir dispositivo con *Home Assistant*

Se despliega un menú en el cual se deberá seleccionar el *firmware* del dispositivo que se quiera agregar al sistema. Una dificultad que se puede encontrar es la de averiguar cuál es el *firmware* del dispositivo. *Home Assistant*, como se ha mencionado tiene una amplia comunidad de usuarios, y en sus foros puedes encontrar a miles de usuarios que indican como agregar la mayoría de dispositivos y ayudarán a resolver las dudas que se tengan. Este es uno de los principales motivos de la elección de *Home Assistant* como gestor de dispositivos.

Para ejemplificar el agregado de un dispositivo, se agregarán unos enchufes inteligentes que tienen el *firmware* Tuya. Para realizar esto, escribimos el nombre del *firmware* en el buscador o lo buscamos entre la lista de *firmwares* y pulsamos sobre él (Ver figura 6.22).

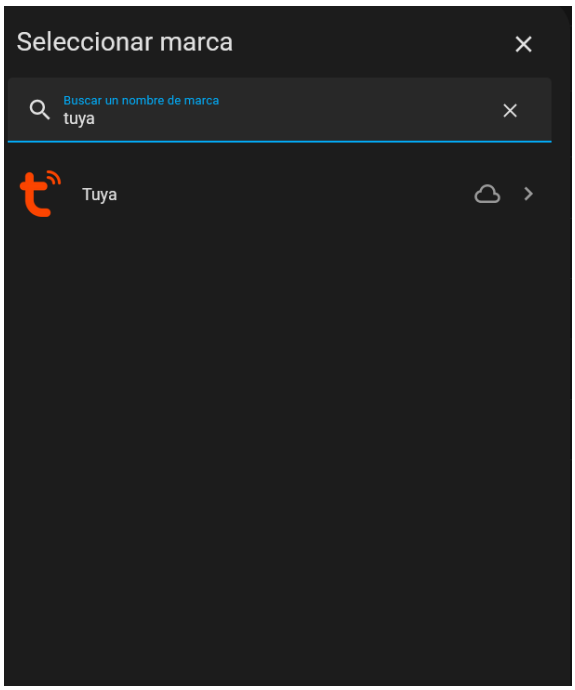

Figura 6.22: Buscar *firmware*

Dependiendo del *firmware* el proceso de agregar un dispositivo puede ser distinto. Las opciones más comunes suelen ser la utilización de algún id de usuario de la app por defecto del dispositivo y/o indicar la IP concreta del dispositivo dentro de la subred.

En caso del ejemplo, solicitará la id de usuario de la aplicación Tuya donde normalmente se agregaría el dispositivo para su control (Ver figura 6.23).

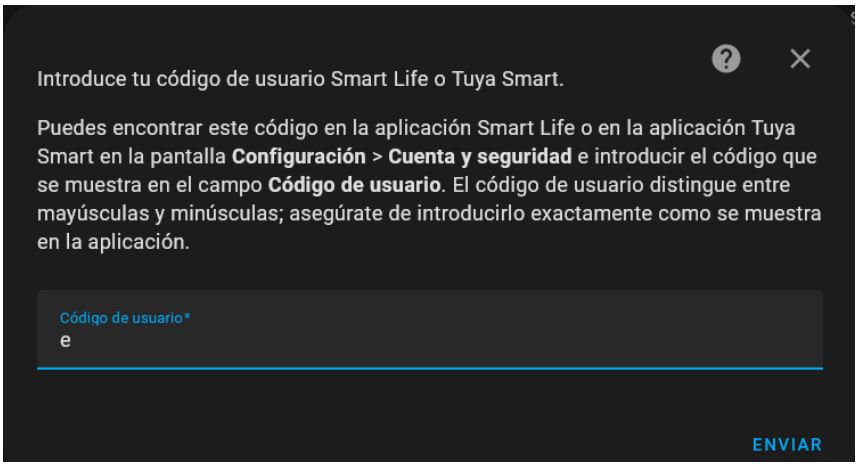

Figura 6.23: Introducir código aplicación Tuya

Una vez introducido el id de usuario todos los dispositivos que estaban previamente agregados a la app serán añadidos automáticamente a *Home Assistant*. En caso de que se quiera agregar un nuevo dispositivo o que se quiera registrar un dispositivo que no estaba en la app de Tuya se podrá hacer desde el apartado integraciones (Se puede observar en la anterior figura 6.21). En este apartado se pulsará en agregar integración y se volverá a seleccionar en la lista Tuya en un proceso muy similar al seguido en el apartado "dispositivos". Una vez seleccionado el *firmware* se seguirá los pasos que aparezcan en pantalla, los cuales suelen consistir en poner el dispositivo en modo detección manteniendo pulsado el botón de apagado y esperando a que el sistema encuentre al dispositivo.

6.5 Problemas con MQTT y uso de una alternativa para la comunicación directa con el dispositivo

6.5.1. Problemas encontrados con mosquitto MQTT

La idea inicial del proyecto era el uso de *mosquitto* ya que se podía integrar fácilmente con *home assistant* pero al realizar este proyecto usando *home assistant* en un contenedor *docker* el *add-on* de Mosquitto no estaba disponible por lo que se han tenido que explorar otras alternativas para poder realizar la misma función que se pretendía realizar con el brocker de mensajería.

El uso inicial que se le pretendía dar a las colas de mensajería era el uso de las mismas como alternativa para la conexión directa con el dispositivo domótico para así explorar una opción que respeta más la privacidad del uso de los dispositivos. Esta privacidad se gana al prescindir del servidor de terceros para el uso de los dispositivos. Además si se consigue el uso local de los dispositivos en caso de que la empresa que controla los servidores deje de dar servicio, los dispositivos podrían seguir siendo usados sin dependencia de la empresa.

Se barajaron varias alternativas a la integración de *MQTT* ya que una de sus principales ventajas ya no era posible, así que se consideró el uso del propio *Mosquitto* desde el terminal pero se acabo descartando por dificultades a la hora de conectar y porque perdía el componente de integración completa al sistema. Se pretende que la nueva solución sustituya por completo al componente de Mosquitto, respetando el modelo definido y desarrollando una alternativa integrable. Por lo que se acabo optando por la opción del desarrollo manual de este nuevo componente, como es la creación de un *script* en *python* para la conexión con el dispositivo, sustituyendo con este *script* al componente de la cola de mensajería en el esquema general del proyecto.

6.5.2. Configuración avanzada para Tuya

Para configurar un dispositivo concreto habrá que revisar la documentación que ofrezca el fabricante del dispositivo, en este caso se va a ejemplificar con Tuya al igual que se hizo en *Home Assistant*.

En este punto es necesario explicar que las ordenes que se envían a los dispositivos, ya sea un simple encendido o apagado, requieren de la comunicación con un servidor de la empresa fabricante del dispositivo. Por ello, se requerirá de la obtención de una serie de claves para realizar la comunicación.

Primero habrá que registrarse en la web de Tuya *developers*, ir al apartado de *cloud* y pulsar en "create cloud project"(ver figura 6.24). Las figuras de este apartado usan referencias a Mosquitto ya que la idea inicial era utilizarlas para configurar las colas de mensajería pero las mismas claves serán utilizadas para el script.

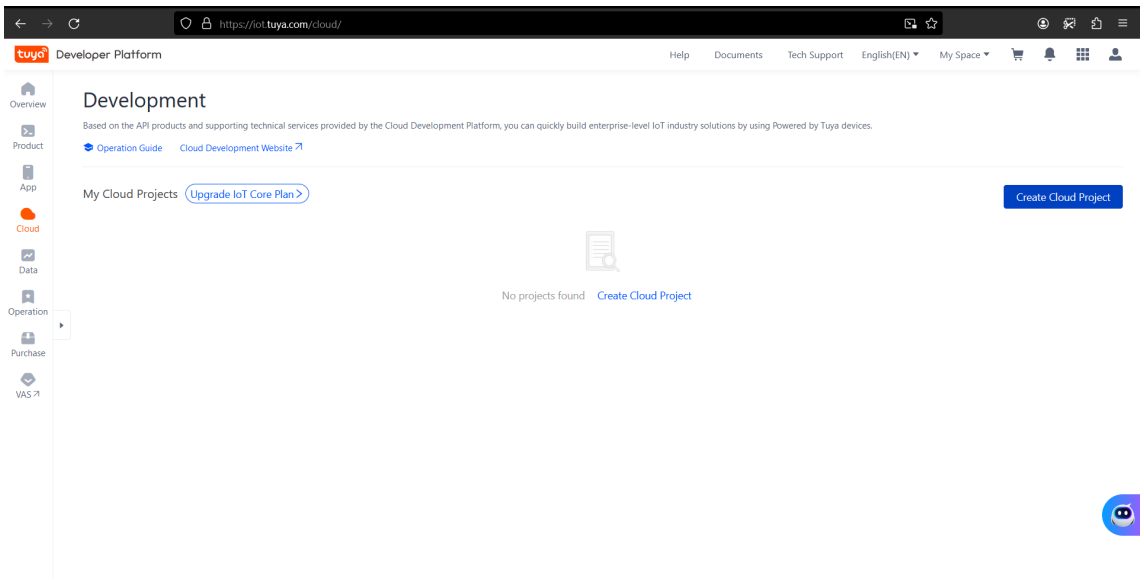

Figura 6.24: Página de desarrollo de Tuya

Saltará una ventana donde habrá que rellenar los campos del proyecto y elegir un *datacenter*, se elegirá el más próximo a la ubicación desde la que se desarrolle el proyecto, en este caso centro Europa. Se pueden ver los campos rellenos en la figura 6.25.

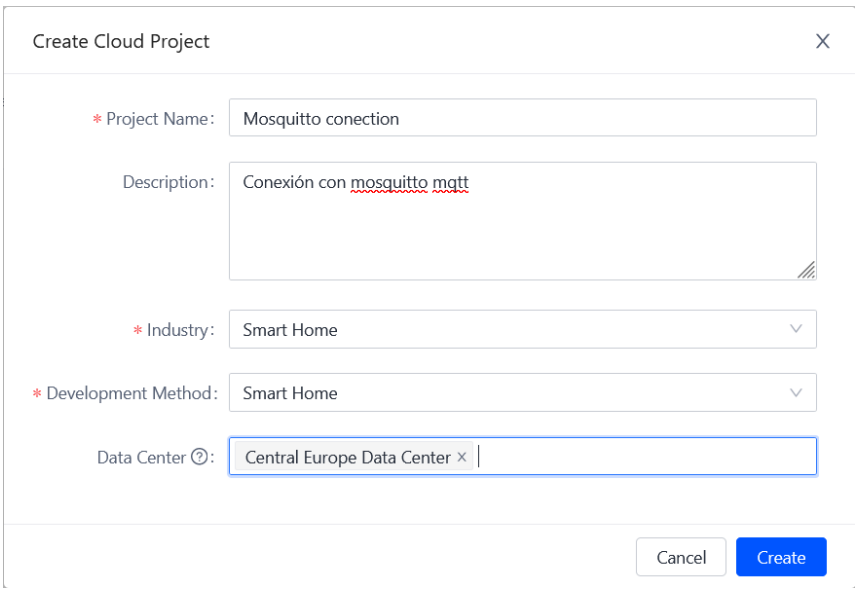

Figura 6.25: Campos proyecto

Después aparecerá una ventana en la que hay que aceptar el acceso a ciertas *APIs*. A partir de este punto el proyecto estará creado y habrá que añadir los dispositivos vinculados a la app móvil. En la página del proyecto habrá que ir a la ruta "Devices" >"Link Tuya App Account" >"Add App Account" (ver figura 6.26). Una vez realizado esto saldrá por pantalla un qr el cual se deberá escanear desde la aplicación móvil.

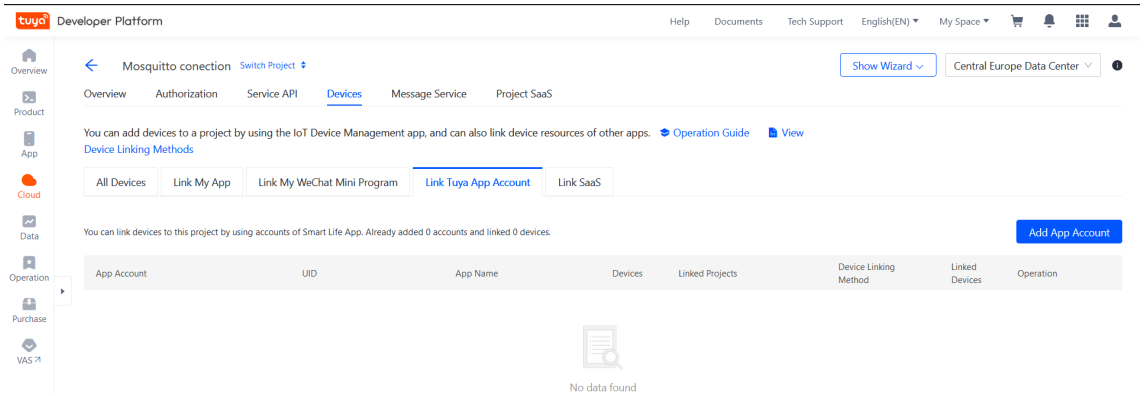

Figura 6.26: Vincular dispositivos desde la web

Ahora desde la app móvil que tiene los dispositivos conectados se deberá pulsar en la parte superior derecha y pulsar en escanear qr. Una vez escaneado el qr aparecerá une mensaje en la app móvil para autorizar la vinculación. Una vez autorizado, aparecerán los dispositivos vinculados (Ver figura 6.27). Una vez en la página habrá que copiar el ”device id” para utilizarlo para obtener las otras el código secreto y el id del producto.

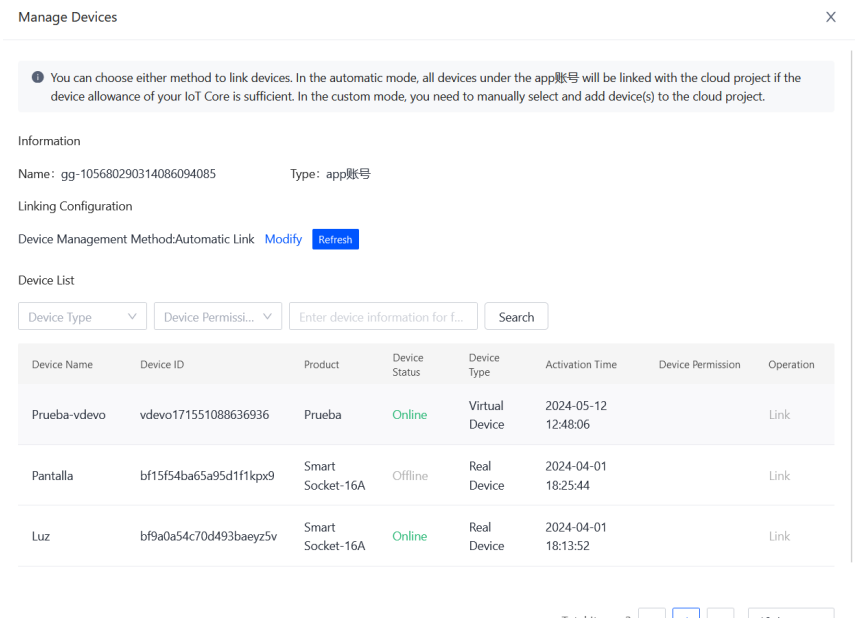

Figura 6.27: Dispositivos vinculados

Para obtener las claves necesarias habrá que ir al apartado *API explorer* en el menú del lateral izquierdo que aparece al desplegar *cloud*. Aparecerá una ventana nueva en la cual habrá que navegar en el panel de la izquierda ”Device Management” >”Query Device Details” y seleccionar arriba el *datacenter* de Europa. Una vez realizado todo esto, se deberá pegar el ”device id” y pulsar en el botón azul de ”Submit Request”. Esto enviará un *curl* el cual devolverá una serie de datos sobre el dispositivo entre ellos el ”local_key” y el ”product_id”. Todo esto se puede ver en la figura 6.28.

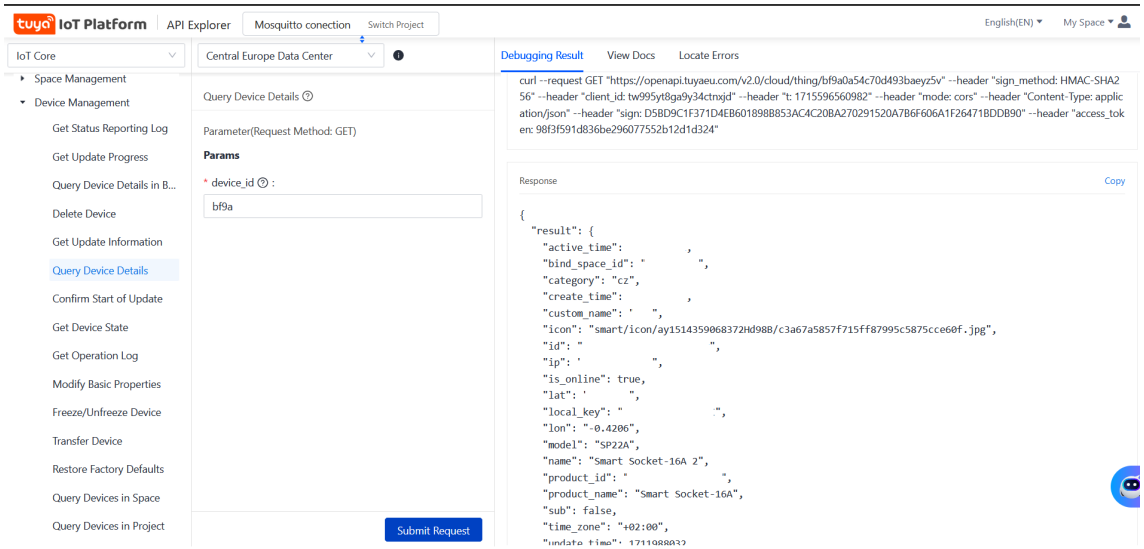

Figura 6.28: Obtención claves secretas dispositivo

6.5.3. Desarrollo del código

Una vez obtenidas las claves quedaba explorar las alternativas a aplicar en el código. Primero se valoró capturar los paquetes para así imitar el envío de mensajes al dispositivo y mediante el uso de las claves controlar el dispositivo, pero esta opción fue descartada ya que los paquetes como se vio en el capítulo de antecedentes van cifrados por *TLS*.

Posteriormente se valoró el uso de librerías y tras investigar las distintas alternativas se llegó a las dos que podrían ayudar a llevar a cabo la conexión en local a los dispositivos: *tinytuya* y *localtuya*. La librería *localtuya* tenía un aspecto interesante dado que también existe un *add-on* en *home assistant* que mediante la introducción del *device id*, el *local key* y la *IP* permite la conexión con el dispositivo en local evitando al servidor central. Pero para el desarrollo del *script* la librería con mejor documentación y más sencilla de aplicar acabo siendo *tinytuya*.

El código desarrollado para una el control y la consulta del estado del dispositivo es el siguiente:

```
import tinytuya

DEVICE_ID = '*****'
IP_ADDRESS = '******'
LOCAL_KEY = '*******'

# Crea una instancia del dispositivo
device = tinytuya.OutletDevice(
    dev_id=DEVICE_ID,
    address= IP_ADDRESS,
    local_key=LOCAL_KEY,
    version=3.3)

accion = input("Quieres encender (1) o apagar (0) el dispositivo? ")

if accion == '1':
    device.turn_on()
    print("Dispositivo encendido.")
elif accion == '0':
    device.turn_off()
    print("Dispositivo apagado.")
else:
```

```
    print("Opción no válida. Debes ingresar 1 para encender o 0 para apagar.")

# Obtén el estado actual del dispositivo
status = device.status()
print(f'Estado del dispositivo: {status}')
```

Con este código se logra la conexión con el dispositivo en local pudiendo encenderlo, apagarlo y consultar su estado sin necesidad de la intervención de ningún servidor de terceros. Se puede ver una prueba de su funcionamiento en la figura 6.29.

```
(tuyatest) gavigar@raspberrypi:~ $ python3 enciendeApaga.py
¿Quieres encender (1) o apagar (0) el dispositivo? 1
Dispositivo encendido.
Estado del dispositivo: {'dps': {'1': True, '9': 0, '18': 0, '19': 0, '20': 2297, '21':
1, '22': 615, '23': 30757, '24': 17734, '25': 1176, '26': 0}}
(tuyatest) gavigar@raspberrypi:~ $ python3 enciendeApaga.py
¿Quieres encender (1) o apagar (0) el dispositivo? 0
Dispositivo apagado.
Estado del dispositivo: {'dps': {'1': False, '9': 0, '18': 0, '19': 0, '20': 2297, '21':
 1, '22': 615, '23': 30757, '24': 17734, '25': 1176, '26': 0}}
```

Figura 6.29: Prueba funcionamiento script

6.6 Integración con home assistant

Dado que no se han podido utilizar los *add-ons* lo ideal es la integración del *script* en la plataforma ya creada de *home assistant* en el *docker* y con ello tener centralizado todo en un único lugar. Para ello, se instalar en el *docker* las dependencias de librería ya sea haciendo ”pip install tinytuya” y ”pip install homeassistant”en el fichero de configuración o accediendo a la consola del *docker* e instalando las dependencias de la misma forma.

Ahora habrá que realizar una serie de pasos en el fichero de configuración de *home assistant*. Lo primero será modificar el fichero configuration.yaml y añadir la linea ”python_script:”. Posteriormente, habrá que crear una carpeta llamada ”python_scripts” donde se situará el *script* de *python* antes de ser modificado. Una vez dentro de esta carpeta habrá que modificar el *script* para que exponga una función la cual sea la que controle el encendido/apagado del dispositivo. El código quedaría de la siguiente forma:

```
from tinytuya import OutletDevice
import homeassistant

DEVICE_ID = '****'
IP_ADDRESS = '****'
LOCAL_KEY = '****'

@service
def controla_enchufe(action):
    try:
        device = OutletDevice(DEVICE_ID, IP_ADDRESS, LOCAL_KEY, 3.3)

        if action == 'encender':
            # Enciende el dispositivo
            device.turn_on()
            print("Enchufe encendido.")
        elif action == 'apagar':
            # Apaga el dispositivo
            device.turn_off()
            print("Enchufe apagado.")
        else:
            print("Acción desconocida:", action)
```

```
    except Exception as e:
        print("Error al controlar el enchufe:", str(e))

controla_enchufe(data.get('action'))
```

Habrá que modificar también el *scripts.yaml* que será el encargado de pasar el parámetro action el cual definirá si el dispositivo se enciende o se apaga. EL *scripts.yaml* debe quedar de la siguiente forma:

```
encender_enchufe:
  alias: Encender Enchufe
  sequence:
    - service: python_script.controla_enchufe
      data:
        action: encender

apagar_enchufe:
  alias: Apagar Enchufe
  sequence:
    - service: python_script.controla_enchufe
      data:
        action: apagar
```

Por último, para que todo esto quede integrado en la interfaz habrá que ir al apartado *overview* dentro de la interfaz de *home assistant* y darle al botón de editar en la parte superior derecha. Tras hacer *click*, aparecerá un botón en la parte inferior derecha que pondrá ”add new card”. Al pulsar sobre el mismo, se podrá añadir una nueva entidad entre las que se podrán seleccionar los dos *scripts* como se ve en la figura 6.30. Con esto se podrá controlar el dispositivo desde la interfaz de *home assistant* de manera local y prescindiendo de los *add-ons* no disponibles con *docker*. Puede que sea necesario reiniciar para que todo funcione correctamente.

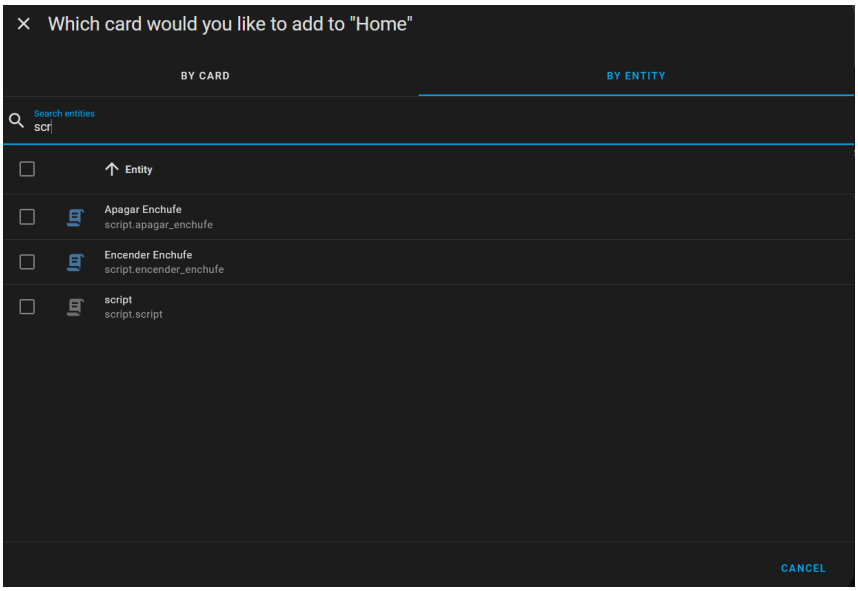

Figura 6.30: Seleccionar scripts

Una vez agregados los scripts, en la interfaz web el resultado quedaría como en la figura 6.31.

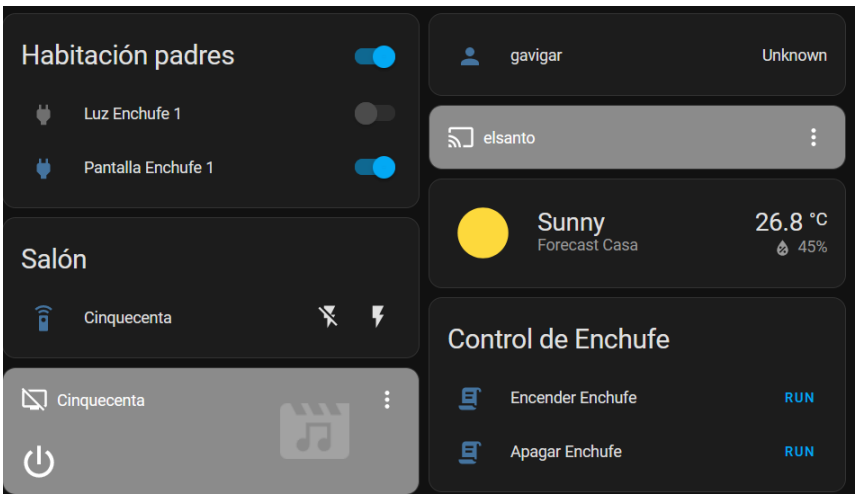

Figura 6.31: Ventana overview de home assistant

6.7 Firewall

Ahora está todo configurado vamos a restringir el uso del *Home Assistant* y *ssh* solo para la subred de la *VPN* y a permitir el uso de los puertos del *script* de python mediante el siguiente *script* de bash:

```
#!/bin/bash

# Limpiar reglas existentes y configurar políticas por defecto
iptables -F
iptables -X
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

# Aceptar tráfico de loopback
iptables -A INPUT -i lo -j ACCEPT

# Aceptar tráfico establecido y relacionado
iptables -A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT

# Aceptar SSH desde la interfaz VPN wg0 (ajustar según tu configuración de puerto)
iptables -A INPUT -i wg0 -p tcp --dport 22 -j ACCEPT

# Aceptar acceso a Home Assistant (puerto 8123) desde la interfaz VPN wg0
iptables -A INPUT -i wg0 -p tcp --dport 8123 -j ACCEPT

# Aceptar acceso del script de Python a través de los puertos 39954 y 39956
iptables -A INPUT -i eth0 -p tcp --dport 39954 -j ACCEPT
iptables -A INPUT -i eth0 -p tcp --dport 39956 -j ACCEPT

# Aceptar tráfico de salida
iptables -A OUTPUT -j ACCEPT

# Guardar las reglas de iptables
iptables-save > /etc/iptables/rules.v4

# Reiniciar servicio de iptables-persistent para aplicar reglas persistentemente
systemctl restart iptables-persistent

```

```
35 # Reiniciar interfaz VPN WireGuard (wg0) si es necesario
36 systemctl restart wg-quick@wg0
37
38 # Mostrar mensaje de éxito
39 echo "Reglas de firewall configuradas correctamente. Conexiones permitidas a
       través de VPN para SSH, Home Assistant y puertos específicos del script de
       Python."
```

Las reglas de firewall se basan por una parte en la prohibición del tráfico por el rango de IPs que no sea de la *VPN* ni de la interfaz y, por otra parte, en permitir el tráfico que venga o vaya por el puerto/s que utilice el *home assistant*. Con la orden de *established* y *related* se permite que una vez establecida la conexión por el *script* de python se pueda mantener un intercambio de mensajes sin que las reglas hagan *DROP* a los mensajes entrantes.

CAPÍTULO 7
Implantación

Tras el capítulo previo en el que aparte de la configuración del sistema se muestra como agregar y configurar dispositivos, se usará este apartado para dar más detalles sobre la puesta en marcha del sistema.

7.1 Disposición del sistema domótico

Para una correcta conexión de la placa a la red, se recomienda que la placa esté conectada directamente al router a través de ethernet. Con esto se reduce el riesgo de perdida de paquetes y, con ello, una mayor fiabilidad a la hora de usar nuestros dispositivos.

Una vez conectada la Raspberry Pi, otro de los aspectos fundamentales es determinar que tipo de dispositivo domótico se quiere unir al sistema. Si ya se poseen ciertos dispositivos habrá que investigar en los foros de *home assistant* su tipo de *firmware* y como configurarlos. Si se quiere adquirir nuevos dispositivos se recomienda que se trate de ser uniforme con la marca y *firmware* de los dispositivos, con ello, se ahorrará tiempo en la configuración ya que el *firmware* ya estará en el sistema y agregarlo será una cuestión de vinculación.

Otro punto que puede añadir comodidad al proyecto es el uso de la aplicación móvil de *Home Assistant*, la cual presenta una interfaz cómoda para el control de los dispositivos. Es importante destacar que el uso de la aplicación móvil parece ser incompatible con el uso de certificados de openssl.

7.2 Agregados al sistema

Como gran parte de la implementación ha sido desarrollada en el capitulo de "desarrollo de la solución", se aprovechará esta sección para explicar como implantar ciertas funcionalidades extras.

Una de las funcionalidades que un usuario de domótica comercial podría echar en falta es el uso de un asistente de voz como *alexa* o *google assistant*. Para integrar este tipo de sistemas hará falta un complemento de pago de home assistant de cloud que permite la integración de este tipo de elementos. Se agrega como un dispositivo normal, siguiendo los pasos vistos en la configuración. La ventaja que tiene la utilización de estos controladores por comando de voz a través de *Home Assistant* es la restricción de la visibilidad de los dispositivos. Es el usuario el que decide que dispositivos exponer para que los mismos sean controlados por voz a través del *google assistant* o de *alexa*.

7.2.1. Visualizado de logs

Una parte fundamental del uso de un sistema informático es el visualizado de los *logs*, ya que serán de gran ayuda para la detección de errores y la reparación de los mismos. Para ello, se explicará en esta sección los distintos *logs* que se podrán visualizar en el sistema.

Logs en Raspberry

En la Raspberry Pi podremos visualizar distintos *logs*:

Logs del sistema en el fichero *syslog*:

```
1 # cat /var/log/syslog
```

Logs de autenticación en el *auth.log*:

```
1 # cat /var/log/auth.log
```

Los logs del kernel:

```
1 # cat /var/log/kern.log
```

Para visualizar los fichero según se van escribiendo se puede utilizar el comando *tail* con la opción -f como por ejemplo:

```
1 # tail -f /var/log/syslog
```

Logs de docker

Para visualizar los logs de docker primero habrá que identificar el contenedor del que se quieren obtener los logs mediante un *docker ps*. Una vez obtenido el id o el nombre habrá que ejecutar:

```
1 # docker logs <container_id or container_name>
```

o en tiempo real mediante:

```
1 # docker logs -f <container_id or container_name>
```

Logs de Home Assistant

Los ficheros de log de *Home Assistant* dentro del contenedor *docker* se encontrarán en el fichero en el cual se guarde la persistencia en un fichero con el nombre *home-assistant.log* dentro de la carpeta */config*. En caso del presente proyecto, la ruta se puede observar en el fichero de despliegue del *docker* y esta ruta es: */home/gavigar/docker/config* (Ver figura 6.15).

7.2.2. Creación de un cron de actualización

Una vez implantado el sistema es fundamental que el sistema esté correctamente actualizado por lo que a continuación se va a explicar como establecer un cron, es decir, para que cada día (o intervalo de tiempo que se desee) realice automáticamente la acción de actualizar el sistema.

Para ello, se deberá crear el fichero cron en la ruta correspondiente de la siguiente forma:

```
# sudo vi /etc/cron.d/actualizar_aplicaciones.cron
```

Dentro del fichero creado se añadirá la siguiente linea, que establece que se realice un update y un upgrade a las 12 del medio día con permisos de root:

```
0 12 * * * root /usr/bin/apt update && /usr/bin/apt upgrade -y
```

Por último, habrá que cambiar los permisos del fichero para que todos los usuarios puedan leer y ejecutar pero que solo root pueda escribir:

```
# sudo chmod 644 /etc/cron.d/actualizar_aplicaciones.cron
```

CAPÍTULO 8
Pruebas

En este apartado se analizarán los resultados de las medidas de seguridad tomadas. Estas pruebas consistirán en el análisis del paquetes para comprobar que efectivamente las medidas están funcionando y activas. Las pruebas serán realizadas independientemente del resto de capas para encontrar posibles errores, es decir que cada apartado tratará individualmente el aspecto a testear, aislándolo de otros para que así los fallos queden más expuestos.

8.1 Funcionamiento del sistema

En la sección de configuración del dispositivo se ha demostrado como se pueden añadir dispositivos al sistema y como estos dispositivos se agregan correctamente al sistema. A lo largo de los días de desarrollo del TFG se ha estado probando el sistema y comprobando que el sistema funciona correctamente y puede controlar los dispositivos. Uno de los problemas surgido durante estas pruebas ha sido que el *token* de acceso de uno de los *firmwares* caduco como consecuencia de un mal reinicio de la raspberry y del *docker*, por lo que se tuvo que volver a agregar el *token ID* para su correcto funcionamiento. Este fallo fue puntual y el funcionamiento ha sido correcto a partir de este punto.

8.2 Comprobación de la seguridad de ssh

8.2.1. Prueba y errores encontrados

Para comprobar la seguridad del *ssh* se utilizará un equipo el cual no tiene ningún *fingerprint* configurado y se tratará de acceder a la placa primero solo con las medidas de seguridad *ssh* y posteriormente con las reglas de *firewall* y la *VPN*. Para ello, se utilizará la consola de *mobaxterm* y se hará el *ssh* a la IP y usuario de la raspberry (suponiendo que el atacante pueda conocer el usuario). En la primera prueba de acceso desde el cliente que no tenía huella el protocolo pidió la contraseña (cosa que no debería hacer) y se pudo acceder mediante la misma. Tras investigar las posibles causas del fallo se llegó a que se debe configurar también el fichero *sshd*.

8.2.2. Solución de fallos

Para solucionarlo habrá que seguir los siguientes pasos:

Configurar el fichero sshd_config.d para desactivar la ”PasswordAuthentication”:

```
# sudo vi /etc/ssh/sshd_config.d
```

Se buscará la linea que pone ”PasswordAuthentication” y habrá que quitar el # para que deje de estar comentada y pasar el valor de ”yes” a ”no” (Ver figura 8.1).

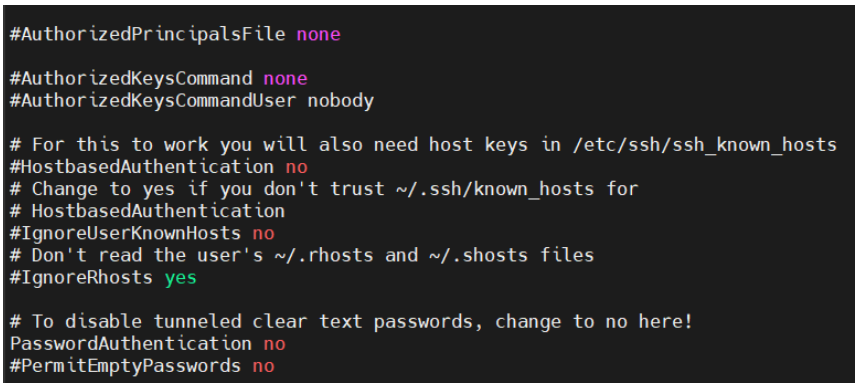

Figura 8.1: PasswordAuhentication en sshd

Ahora repitiendo la prueba y en la figura 8.2, podemos ver como la raspberry rechaza la conexión y que por tanto la configuración es correcta.

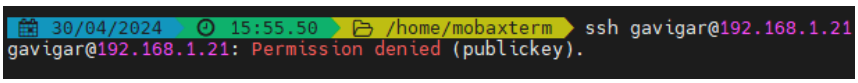

Figura 8.2: Conexión ssh rechazada por la raspberry

Cabe destacar que una vez introducidas las reglas de *firewall* el fallo hubiese quedado parcheado pero dado el análisis por capas realizado en estas pruebas se ha podido detectar el fallo.

8.2.3. Intento de conexión sin acceso a la VPN

Ahora se realizará una prueba de conexión al *home assistant* desde fuera de la *VPN*, es decir, se simulará el intento de acceso de un usuario que haya accedido a la red local e intenta acceder mediante al *home assistant*. Al intentar acceder tanto por la *IP* local asignada a la Raspberry Pi como por la *IP* de la subred de la *VPN* se encontrará el resultado de la figura 8.3, en la cual se puede ver que la placa no responde a la solicitud, ya sea porque la placa hace un *DROP* o porque la petición no consigue alcanzar la placa debido a no estar en la subred, y por ello da un *timeout*. Por otra parte, se ha realizado un intento desde fuera de la red con la IP pública y el puerto del port fordwarding y en la subred con la IP local de la raspberry, en ambos casos con resultado iguales al de la figura 8.3.

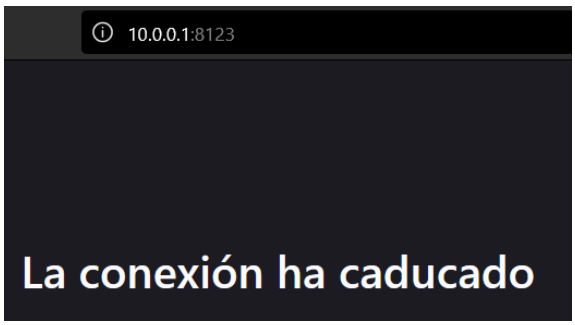

Figura 8.3: Intento de entrar a *home assistant* desde fuera de la *VPN* con *IP* subred

8.3 Comprobación de la configuración de la *VPN*

Para verificar si el cifrado de la *VPN* es correcto se utilizará *wireshark* como herramienta. *Wireshark* sirve para la captura y el análisis de paquetes. Para realizar la siguiente captura se deberá emplear el equipo con la *VPN* configurada para poder enviar tráfico a través del túnel y que en los paquetes se pueda visualizar. En la figura 8.4.

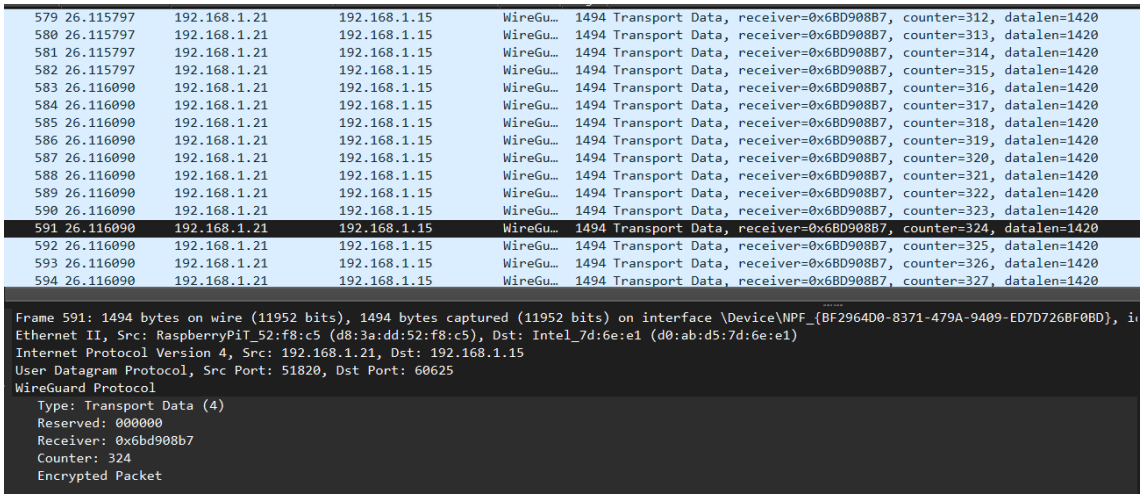

Figura 8.4: Captura *wireshark* de *wireguard*

Se puede observar que *Wireguard* usa su propio protocolo para encapsular por encima de de *UDP* los mensajes y enrutarlos hacia el destino.

Si se trata de analizar el tráfico desde fuera de la *VPN* es difícil trazar los paquetes y aunque podrían encontrarse filtrando por el puerto utilizado por *wireguard* el paquete al ir cifrado no se puede obtener información del mismo. Esto es muy útil si se quisiera acceder a la placa de forma remota desde fuera del hogar abriendo puertos.

8.4 Análisis de los paquetes tras la introducción de *openssl*

En este caso volveremos a emplear *wireshark* para la captura de paquetes. La captura se realizará escuchando el tráfico de la red y filtrando por el puerto 8123, puerto en el cual escucha el *Home Assistant*. En la captura realizada (Ver figuras 8.5 y 8.6) 1110se puede apreciar que existe un *handshake* para el establecimiento de la conexión y que además hay una capa extra en el paquete que *wireshark* reconoce como *TLS* (Transport Layer Security). Esta configuración permite que los datos vayan cifrados y no por *HTTP* plano, es decir,

legible para cualquier atacante que esté escuchando en la red. Realmente el sistema *ssl* que es menos seguro que *tls*, pero *wireshark* reconoce la cabecera como *tls*.

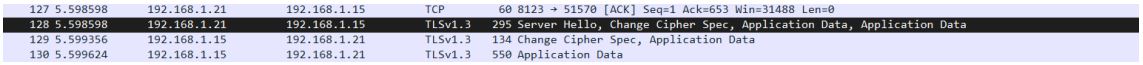

Figura 8.5: Captura *wireshark* tras realizar la configuración de *openssl*

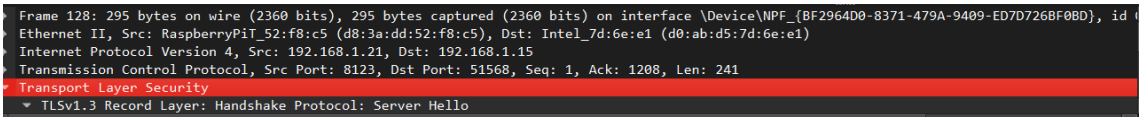

Figura 8.6: Captura de las capas de uno de los paquetes de *Home Assistant*

8.5 Análisis del tráfico home assistant

Al igual que en el capítulo de antecedentes en esta sección se analizará el tráfico del *home assistant* al realizar la acción de encendido/apagado del dispositivo.

En el tráfico visto en la figura 8.7 se puede observar el intercambio de mensajes, primero entre el pc y la raspberry (*home assistant*) para darle la orden de que quiere apagar el enchufe y, posteriormente, entre la raspberry y los servidores de *tuya*. El *home assistant* en esta ocasión hace de intermediario entre los servidores de las marcas de los distintos dispositivos para así unificar e integrar la comunicación con todos los dispositivos presentes en el hogar. También se podemos observar que al igual que antes del uso de *home assistant* el tráfico va cifrado con el protocolo *TLS* asegurando que por lo menos terceros no podrán tener acceso a la información. A pesar de que los atacantes capturen el tráfico no podrán ver el contenido, pero continua existiendo una dependencia y vulnerabilidad de privacidad por parte del servidor de Tuya el cual sí que tiene acceso a los datos ya que tienen acceso a la información descifrada. Esta captura a nivel de intercambio de mensajes entre pc y servidor de Tuya es muy similar tanto con el uso de la *VPN* como sin usarla.

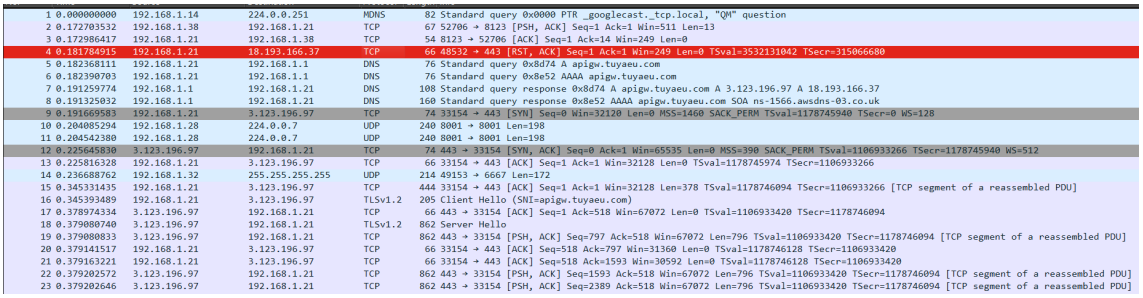

Figura 8.7: Tráfico capturado desde la Raspberry

8.6 Análisis del tráfico con el uso del script

Tras la captura del tráfico mediante el uso de *home assistant*, también se capturará el tráfico resultante del uso del *script* para confirmar si se prescinde realmente de terceros a la hora de comunicarse con el dispositivo.

En la figura 8.8 se puede observar como el tráfico es completamente realizado entre la raspberry (*IP 192.168.1.21*) y el dispositivo (*IP 192.168.1.32*). A pesar de que el tráfico ha

pasado a ser en plano (sin cifrar) también ha pasado a no salir de la red local haciendo que agentes externos ya no puedan capturar el tráfico. Con la captura se puede concluir que el uso del *script* es exitoso y que evitamos al servidor como intermediario para controlar el dispositivo. Para bloquear completamente el acceso del servidor a la información del dispositivo habría que bloquear las *IPs* de los servidores mediante el *firewall*, pero esto provocaría que tanto la aplicación de Tuya los dispositivos configurados como Tuya en *home assistant* dejasen de funcionar, siendo el *script* o el uso del *add-on localtuya*la única alternativa para la conexión a los dispositivos.

No.	Time	Source	Destination	Protocol	Length	Info
17	1.737833434	192.168.1.21	192.168.1.32	TCP	74	39954 → 6668 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=547239080 TSecr=0 WS=128
22	1.838190827	192.168.1.32	192.168.1.21	TCP	60	6668 → 39954 [SYN, ACK] Seq=0 Ack=1 Win=2920 Len=0 MSS=1460
23	1.838232789	192.168.1.21	192.168.1.32	TCP	54	39954 → 6668 [ACK] Seq=1 Ack=1 Win=32120 Len=0
24	1.922644971	192.168.1.21	192.168.1.32	TCP	205	39954 → 6668 [PSH, ACK] Seq=1 Ack=1 Win=32120 Len=151 [TCP segment of a reassembled PDU]
25	1.947718626	192.168.1.32	192.168.1.21	TCP	145	6668 → 39954 [PSH, ACK] Seq=1 Ack=152 Win=2769 Len=91 [TCP segment of a reassembled PDU]
26	1.947780291	192.168.1.21	192.168.1.32	TCP	54	39954 → 6668 [ACK] Seq=152 Ack=92 Win=32029 Len=0
27	1.948364988	192.168.1.21	192.168.1.32	TCP	54	39954 → 6668 [FIN, ACK] Seq=152 Ack=92 Win=32029 Len=0
29	1.948742520	192.168.1.21	192.168.1.32	TCP	74	39956 → 6668 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=547239291 TSecr=0 WS=128
30	1.949747415	192.168.1.32	192.168.1.21	TCP	82	6668 → 39954 [PSH, ACK] Seq=92 Ack=152 Win=2769 Len=28 [TCP segment of a reassembled PDU]
31	1.949801544	192.168.1.21	192.168.1.32	TCP	54	39954 → 6668 [RST] Seq=152 Win=0 Len=0
32	1.950717089	192.168.1.32	192.168.1.21	TCP	60	6668 → 39954 [ACK] Seq=120 Ack=153 Win=2768 Len=0
33	1.950737014	192.168.1.21	192.168.1.32	TCP	54	39954 → 6668 [RST] Seq=153 Win=0 Len=0
34	1.951676929	192.168.1.32	192.168.1.21	TCP	60	6668 → 39956 [SYN, ACK] Seq=0 Ack=1 Win=2920 Len=0 MSS=1460
35	1.951704965	192.168.1.21	192.168.1.32	TCP	54	39956 → 6668 [ACK] Seq=1 Ack=1 Win=32120 Len=0
36	1.952137867	192.168.1.21	192.168.1.32	TCP	206	39956 → 6668 [PSH, ACK] Seq=1 Ack=1 Win=32120 Len=152 [TCP segment of a reassembled PDU]
37	1.958288588	192.168.1.32	192.168.1.21	TCP	60	6668 → 39956 [ACK] Seq=1 Ack=153 Win=2768 Len=0
38	1.971572110	192.168.1.32	192.168.1.21	TCP	194	6668 → 39956 [PSH, ACK] Seq=1 Ack=153 Win=2768 Len=140 [TCP segment of a reassembled PDU]
39	1.971592276	192.168.1.21	192.168.1.32	TCP	54	39956 → 6668 [ACK] Seq=153 Ack=141 Win=31980 Len=0
40	1.972011289	192.168.1.21	192.168.1.32	TCP	54	39956 → 6668 [FIN, ACK] Seq=153 Ack=141 Win=31980 Len=0
43	1.974225077	192.168.1.32	192.168.1.21	TCP	60	6668 → 39956 [ACK] Seq=141 Ack=154 Win=2767 Len=0
44	1.988177016	192.168.1.32	192.168.1.21	TCP	60	6668 → 39956 [RST, ACK] Seq=141 Ack=154 Win=2920 Len=0

Figura 8.8: Tráfico capturado cuando se utiliza el *script*

CAPÍTULO 9
Conclusiones

A modo de cierre de este trabajo, podemos señalar que la consecución de la creación de un sistema de control y monitoreo de dispositivos del hogar con Raspberry Pi ha sido un éxito. Se ha logrado el desarrollo del sistema diseñado, se han establecido medidas de seguridad para proteger el acceso al sistema y se han llevado a cabo pruebas para comprobar el correcto funcionamiento del sistema.

Cada uno de los objetivos definidos al comienzo del trabajo han sido satisfactoriamente realizados. Se ha definido un análisis de las tecnologías y alternativas que se pueden llegar a utilizar para la realización de un sistema de control de dispositivos del hogar, esto con el fin de que se puedan tener en cuenta las distintas opciones y estudiar los posibles cambios a realizar. Además se ha descrito una configuración detallada del software seleccionado, así como de las medidas de seguridad implementadas en el sistema, exponiendo algunas de las dificultades encontradas y la solución a la que se ha llegado. También se han dispuesto pruebas de funcionamiento, en las cuales se ha comprobado el correcto uso del cifrado y como el *script* ha logrado hacer que el uso del dispositivo se realice en local. Con estas pruebas se han descubierto errores en la configuración de algunos protocolos, por lo que estás pruebas han servido para arreglar y mejorar el sistema.

Se han encontrado ciertas dificultades durante realización del TFG, principalmente causados por la decisión de no utilizar el sistema operativo de *Home Assistant*. Esto ha sido solventado mediante la realización de un *script* como alternativa. Pero en caso de dedicar la Raspberry Pi únicamente al control domótico, sería interesante explorar el uso de *add-ons* nativos en la plataforma de *Home Assistant*. Otro de los aspectos que sería interesante desarrollar es la integración de elementos *cloud* los cuales se pueden integrar mediante el pago de una cuota de *Home Assistant* pero que también existen mecanismos para la integración de los mismos de forma manual.

Algunos de los aspectos que sería interesante tratar en futuros TFGs serían por una parte la protección del acceso remoto a la placa, explorando como proteger la entrada a través del *router*, y por otra, el tratamiento de los datos obtenidos de los dispositivos para realizar estudios de consumo energético y de tiempo de uso.

En conclusión, acercar el uso libre de las tecnologías sin la dependencia y el riesgo de dar a terceros tu información es algo plausible y con este TFG queda demostrado. Cada día las personas deben ser más conscientes y capaces de utilizar las tecnologías a su gusto y crear los proyectos que quieran con herramientas accesibles como la Raspberry Pi.

9.1 Relación del trabajo desarrollado con los estudios cursados

El trabajo realizado tiene plena relación principalmente con la rama cursada ya que he requerido de todos los conocimientos adquiridos en la misma. El trabajo se ha centrado en la protección utilizando protocolos como *ssh* y *VPNs*, ambos elementos estudiados en la asignatura de "Redes corporativas". La integración de los dispositivos también ha sido una parte fundamental del desarrollo del trabajo, lo cual formaba parte de los conocimientos adquiridos en "Integración de aplicaciones".

También he requerido de conocimientos sobre seguridad y el funcionamiento de la red, que he podido adquirir gracias a asignaturas como "Seguridad en redes" o "Sistemas y servicios en red". Además de la utilización de *ssl* y *tls* para la protección de las comunicaciones que ha sido otro de los conocimientos adquiridos en "Redes corporativas".

Fuera de los conocimientos específicos sobre las tecnologías, los estudios cursados me han permitido adquirir otras competencias como la capacidad de analizar y resolver problemas, la capacidad de diseñar y crear proyectos, la capacidad de planificar proyectos y la concienciación con problemas contemporáneos.

La creación de un sistema completo y funcional en el que se utilizan varias tecnologías fuertemente relacionadas con las tecnologías de la información era uno de los objetivos que pretendía conseguir y he conseguido. Con este trabajo he podido aplicar los conocimientos adquiridos durante la carrera los cuales, más que permitirme configurar todo a la primera, me han permitido resolver las dificultades que se me presentaban en el camino haciendo que pueda llevar a cabo el proyecto.

Bibliografía

[1] Google. Google nest crea tu hogar conectado. https://store.google.com/category/connected_home?hl=es, 2024.

[2] Samsung. Conecta tu hogar con samsung. https://www.samsung.com/es/smartthings/, 2024.

[3] Apple. App casa - apple. https://www.apple.com/es/home-app/, 2024.

[4] OpenHAB. Documentación de openhab. https://www.openhab.org/docs/installation/openhabian.html, 2024.

[5] Domoticz. Raspberry pi - domoticz. https://www.domoticz.com/wiki/Raspberry_Pi, 2024.

[6] Jeedom. Documentation jeedom. https://doc.jeedom.com/fr_FR/installation/rpi, 2024.

[7] Home Assistant. Raspberry pi: Install home assistant operative system. https://www.home-assistant.io/installation/raspberrypi/#install-home-assistant-operating-system, 2024.

[8] Colin Dow. *Internet of things programming projects*. Packt Publishing, 2018.

[9] Sinnarkar O. Saurav S. Soni, C. Kumari P. Assistive domotic system: Survey on the development of home automation system via raspberry pi and voice assistive technology. https://acortar.link/EJmFes, 2024.

[10] Valentí Beso Ballester. Diseño y creación de una casa domótica mediante arduino y raspberry pi. https://riunet.upv.es/handle/10251/183775, 2022.

[11] Adrián Zengotitabengoa García. Integración de sistemas domóticos mediante una plataforma de código abierto. https://riunet.upv.es/handle/10251/151382, 2020.

[12] Antonio Camarena Ivars. Instalación y configuración de un sistema integral de hogar inteligente basado en plataformas de software libre. https://riunet.upv.es/handle/10251/120314, 2019.

[13] Javier Ferrero Cerdá. Diseño e implementación de proyecto e infraestructura iot. https://riunet.upv.es/handle/10251/127876, 2019.

[14] El Mundo. Meta pagará 725 millones de dólares para resolver la demanda colectiva por la filtración de facebook a cambridge analytica. https://www.elmundo.es/economia/empresas/2022/12/23/63a5909bfc6c835b3a8b4588.html, 2022.

[15] Raspberry Pi. Raspberry pi operative system. https://www.raspberrypi.com/software/operating-systems/, 2024.

[16] PhoenixNAP. How to use public key authentication with ssh. https://phoenixnap.com/kb/ssh-with-key, 2024.

[17] novamostra. Add ssl/tls certificate to home assistant. https://novamostra.com/2023/04/29/add-ssl-tls-certificate-to-homeassistant-web-interface/, 2024.

APÉNDICE A
Glosario

Clave privada: es el otro componente del par de claves del cifrado asimétrico. Se utiliza para cifrar datos, los cuales podrán ser descifrados con la clave pública, y para descifrar datos cifrados con la clave pública. Esta clave no se puede compartir ya que si otra persona las posee podría suplantar la identidad y vulnerar mensajes cifrados con la clave pública.

Clave pública: es un componente del par de claves del cifrado asimétrico. Se utiliza para cifrar datos, los cuales podrán ser descifrados con la clave privada, y para descifrar datos cifrados con la clave privada. Esta clave se puede compartir libremente ya que se usará para que el resto de usuarios sepan que se están comunicando con el servidor que posee la clave pública con la que cifran.

Clave secreta: Es una clave utilizada en criptografía simétrica para cifrar y descifrar datos.

Contenedor: en informática es una unidad ligera que encapsula una imagen de un sistema operativo o software y sus dependencias, permitiendo que se ejecute de manera aislada en cualquier entorno.

Criptografía asimétrica: se utiliza pares de claves públicas y privadas, la clave pública será compartida por ambos y la clave privada será conservada en secreto por cada extremo de la comunicación.

Criptografía simétrica: se usa una sola clave secreta compartida por ambos miembros de la comunicación y se utiliza para cifrar y descifrar datos.

Docker: Software muy extendido para la gestión de contenedores.

Fingerprint: Es una representación única y corta de la clave pública de un servidor *ssh* y se utiliza para representarlo de manera única y verificar su autenticidad.

Internet of Things (IoT): Se refiere a la conexión de dispositivos a través de internet compartiendo y recopilando datos. Estos dispositivos pueden ser electrodomésticos, sensores y otros dispositivos que permiten el monitoreo y recopilación de datos en tiempo real.

Open source o código abierto: Es un modelo de desarrollo de software en el cual el código fuente está accesible para cualquier persona de forma gratuita y se permite su modificación y redistribución. Las comunidades de programadores se juntan para la creación y mejora de estos programas.

Raspberry Pi: Pequeño ordenador el cual destaca por su bajo coste y su pequeño tamaño. Existen varias versiones de este hardware al igual que se pueden ejecutar en ella distintos sistemas operativos.

SSH: Su significado es ”secure shell” y es un protocolo criptográfico de red que proporciona una forma segura de acceder a un equipo remoto de forma segura.

VPN: Son las siglas de ”Virtual Private Network” y es una tecnología que establece una conexión segura y cifrada sobre una red, permitiendo transmitir datos de manera segura a través de redes inseguras.

APÉNDICE B
ODS - Objetivos de Desarrollo Sostenible

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenible	**Alto**	**Medio**	**Bajo**	**No procede**
ODS 1. **Fin de la pobreza.**			X	
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**		X		
ODS 4. **Educación de calidad.**				X
ODS 5. **Igualdad de género.**				X
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**		X		
ODS 8. **Trabajo decente y crecimiento económico.**				X
ODS 9. **Industria, innovación e infraestructuras.**	X			
ODS 10. **Reducción de las desigualdades.**				X
ODS 11. **Ciudades y comunidades sostenibles.**		X		
ODS 12. **Producción y consumo responsables.**	X			
ODS 13. **Acción por el clima.**			X	
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**				X
ODS 17. **Alianzas para lograr objetivos.**		X		

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

El buen uso de la domótica puede beneficiar en gran medida a los hogares de las personas que la usen. Esto es debido principalmente a la programación del encendido y apagado de dispositivos, pudiendo limitar así su uso únicamente a las franjas en las que se le de uso a los mismos y/o el consumo eléctrico sea más asequible económicamente.

Estas acciones pueden suponer una reducción considerable del uso de la electricidad, reduciendo así tanto el impacto económico del uso de la electricidad como el impacto ecológico de la generación de esa electricidad que a priori sería desperdiciada. Todo esto se puede relacionar con los ODS que estén vinculados con el ahorro o mejor uso de la energía, la reducción de la pobreza (dado el ahorro económico que supone el uso de la domótica) y la innovación. A continuación, explicaré el porqué de cada ODS marcado en la tabla de la parte superior y su relación con mi trabajo:

- ***ODS 1. Fin de la pobreza.*** *Es cierto que mi proyecto no es accesible para las personas con peor situación económica, pero para las clases bajas el ahorro en la factura eléctrica puede suponer un gran alivio en su economía.*
- ***ODS 3. Salud y bienestar.*** *La domótica supone una mejora en la calidad de vida de las personas ya que pueden automatizar tareas del día a día. Esto puede aliviar la carga mental del usuario pudiendo así reducir sus preocupaciones mejorando así su bienestar. También permite seguir y establecer rutinas lo cual es positivo para la salud de las personas.*
- ***ODS 7. Energía asequible y no contaminante.*** *Siguiendo con la línea de los argumentos ya expuestos, el buen uso de la energía es algo fundamental ya que supone un ahorro considerable tanto a nivel económico como en el impacto ecológico.*
- ***ODS 9. Industria, innovación e infraestructuras.*** *La domótica y la securización de la misma están fuertemente relacionados con la innovación y la industria ya que poco a poco la automatización de los procesos se está introduciendo tanto en la vida diaria como en los procesos industriales.*
- ***ODS 11. Ciudades y comunidades sostenibles.*** *La sostenibilidad de las ciudades pasa por el uso de dispositivos para controlar las emisiones y poder así tener datos para controlar las emisiones. La creación de espacios verdes puede involucrar la domótica para un buen cuidado de sus espacios, necesitando también una mejor seguridad en los mismos para evitar problemas.*
- ***ODS 12. Producción y consumo responsables.*** *Uno de los puntos principales del buen uso de la domótica es el consumo responsable de la electricidad reduciendo el desperdicio y el mal uso de la misma.*
- ***ODS 13. Acción por el clima.*** *Como ya he mencionado el ahorro energético supone una colaboración indirecta para la reducción de la huella de carbono gracias al consumo sostenible.*
- ***ODS 17. Alianzas para lograr los objetivos.*** *El hecho de realizar proyectos tecnológicos y compartirlos con el mundo implica la unión de las personas para la evolución, la innovación y la unión en la creación de proyectos tecnológicos. Este TFG no es excepción a esto ya que implica colaborar en el desarrollo e innovación tecnológicas y en el ahorro energético.*

En conclusión, este TFG que puede parecer que no tenga tanto impacto sobre los ODS puede suponer un cambio significativo. Cada acción cuenta cuando se trata del cambio y el desarrollo conjunto en colaboración para un bien común.