UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Grado en Ingeniería del Software

Blockchain para gestionar vulnerabilidades y exposiciones comunes en sistemas de Smart Grid federados - VECsg

Blockchain to manage common vulnerabilities and exposures in federated Smart Grid systems - VECsg

Realizado por
Álvaro Luna Luna

Tutorizado por
María Cristina Alcaraz Tello

Departamento
Lenguajes y Ciencias de la Computación
UNIVERSIDAD DE MÁLAGA

MÁLAGA, septiembre de 2022

UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA DEL SOFTWARE

Blockchain para gestionar vulnerabilidades y exposiciones comunes en sistemas de Smart Grid federados - VECsg

Blockchain to manage common vulnerabilities and exposures in federated Smart Grid systems - VECsg

Realizado por
Álvaro Luna Luna

Tutorizado por
María Cristina Alcaraz Tello

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, septiembre de 2022

Fecha defensa: octubre de 2022

Resumen

La manera en la que gestionamos la energía eléctrica debe evolucionar para que sea mucho más eficiente y óptima. Esto es a lo que aspira la *Smart Grid* o red eléctrica inteligente del futuro, un nuevo modelo de sistema eléctrico que pone el foco en el intercambio de datos entre todas las entidades de la red eléctrica que se pretende que cuenten con infinidad de dispositivos conectados (como, por ejemplo, sensores del *Internet of Things*). Es por ello que la Smart Grid trae consigo muchas ventajas, pero a su vez está expuesta a múltiples tipos de riesgos y amenazas de seguridad que deberían ser compartidas por la propia comunidad que conforma el consorcio de la Smart Grid para reducir los riesgos en entornos relacionados y garantizar la prevención proactiva, en parte, por la compartición de datos. Por esta razón, este Trabajo Fin de Grado aborda esta necesidad particular y plantea y desarrolla una plataforma web segura de gestión de Vulnerabilidades y Exposiciones Comunes basada en una red de blockchain permisionada, a la que se le ha dado el nombre de VECsg. Este sistema va a permitir a las entidades críticas y relacionadas dentro de una red de Smart Grid, compartir en un consorcio privado datos de vulnerabilidades, garantizando que estos datos sean de confianza e inalterables, además de proporcionar trazabilidad completa y conocimiento real de autoría (conocido en inglés como *accountability*), gracias a la tecnología blockchain. VECsg ofrece amplias y numerosas funcionalidades, tales como la gestión de vulnerabilidades con un formato propio, sistemas de alerta, gestión de permisos, etc. Todo centrado en que los miembros del consorcio de la Smart Grid puedan gozar de un repositorio de vulnerabilidades cómodo e intuitivo a la vez que seguro y controlado, que les permita mantener una base de conocimiento de vulnerabilidades compartida y útil.

Palabras clave: smart grid, blockchain, ciberseguridad, vulnerabilidades, CVE

Abstract

The way in which we manage electrical energy must evolve to be much more efficient and optimal. This is what the Smart Grid or intelligent electrical network of the future aspires to, a new model of electrical system that focuses on the exchange of data between all the entities of the electrical network that are intended to have infinity of connected devices (such as sensors of the *Internet of Things*). That is why the Smart Grid brings many advantages with it, but at the same time it is exposed to multiple types of risks and security threats that should be shared by the community that makes up the Smart Grid consortium to reduce risks in related environments and ensure proactive prevention, in part, by data sharing. For this reason, this Final Degree Project addresses this particular need and proposes and develops a secure web platform for the management of Common Vulnerabilities and Exposures based on a permissioned blockchain network, which has been given the name of VECsg. This system will allow critical and related entities within a Smart Grid network to share vulnerability data in a private consortium, guaranteeing that this data is trustworthy and tamper-proof, in addition to providing complete traceability and accountability, thanks to blockchain technology. VECsg offers a large and rich set of functionalities, such as vulnerability management with a particular format, alert systems, permission management, etc. All designed so that the members of the smart grid consortium can enjoy a friendly and intuitive repository of vulnerabilities, as well as safe and controlled, which allows them to maintain a shared and useful knowledge base of vulnerabilities.

Keywords: smart grid, blockchain, cybersecurity, vulnerabilities, CVE

Índice

1 Introducción

1.1. Motivación

La Smart Grid es la red eléctrica con la capacidad de sincronizar de forma inteligente y dinámica las acciones de todos y cada uno de los miembros con los diferentes perfiles que se conectan a ella —los que generan energía, los que la consumen, los que la transportan— con el fin de suministrar electricidad de manera segura, eficiente y sostenible [1]. Esto es posible gracias a la digitalización y la hiperconexión de los dispositivos IoT (*Internet of Things* [2]), permitiendo así, manejar gran cantidad de datos en tiempo real e incluso controlar a distancia estos dispositivos eléctricos, por ejemplo, en caso de averías que requieran intervención o ante aumentos del consumo que requieran abrir o cerrar unos circuitos para guiar el destino de la energía, acciones que se podrían llegar a automatizar y optimizar gracias al nuevo planteamiento que ofrece la Smart Grid.

Esta nueva forma de concebir la red eléctrica, mucho más conectada, supone un nuevo reto para los especialistas en ciberseguridad. Proteger un sistema como el eléctrico, clave para el funcionamiento de las sociedades modernas, se hace aún más complicado con la aparición y el desarrollo de la Smart Grid, que aunque abre muchas posibilidades, también habilita numerosos vectores de ataque explotables. Los atacantes malintencionados van a poner su foco en los numerosos dispositivos que permiten que la red eléctrica inteligente funcione con normalidad (contadores inteligentes, dispositivos de control industrial, sistemas software de operación de la red, etc.).

Tal y como profundizaremos más adelante (véase el capítulo 2 donde se aborda el estado del arte), el escenario que se presenta con la Smart Grid, es muy favorable para los ciberdelicuentes: multitud de dispositivos y aparatos conectados distribuidos a lo largo de muchas entidades y organizaciones del consorcio de la red eléctrica, complicándose enormemente la gestión de la seguridad de cada uno de ellos y de todo el sistema en general. Por eso, surge una plataforma como VECsg, denominada así por la combinación de Vulnerabilidades y Exposiciones Comunes ("VEC") y Smart Grid ("sg"), para ofrecer un lugar donde las entidades de la Smart Grid puedan guardar de forma segura las vulnerabilidades más relevantes.

Además, la idea no es que estas vulnerabilidades se queden almacenadas solo para un organización, sino que, se compartan entre todas los miembros del consorcio privado de la Smart Grid, que se entiende que funciona unido, colaborando para conseguir el correcto funcionamiento de la red eléctrica. El potencial de compartir conocimiento de vulnerabilidades entre entidades, reside en la capacidad de tomar mayor consciencia de lo robusto o débil que es el sistema Smart Grid en su conjunto y en que una organización puede tomar ventaja de las vulnerabilidades con las que otras, dentro de su mismo consorcio, han tenido que lidiar, por ejemplo, aprovechando que a través de VECsg, se pueda compartir cómo actuar y darle solución a estos problemas en diferentes dispositivos y sistemas software.

En definitiva, se busca proteger esta información crítica sobre vulnerabilidades, por lo que tienen que premiar la confidencialidad, la integridad y el control de acceso, para que no haya intrusiones de fuera del consorcio. Un consorcio dentro del cual, los usuarios, también tienen que estar bien identificados y sus roles bien definidos. Para poder satisfacer todos estos requisitos se decide utilizar, en VECsg, la tecnología blockchain permisionada, que permite crear un repositorio de gestión y almacenamiento de vulnerabilidades, seguro, inalterable y confiable.

1.2. Objetivos

El Trabajo Fin de Grado consistirá en desarrollar una red de Blockchain privada y segura, capaz de albergar vulnerabilidades como un modo de ofrecer a una comunidad federada un repositorio común e íntegro. Estas vulnerabilidades se tratarán como una

adaptación de las de CVE (que en VECsg, se les ha dado el nombre de registros VEC), por lo que seguirán unas pautas de formato preestablecidas. La idea es que este repositorio sea cerrado y propio del consorcio de una Smart Grid, de manera que sólo pueden acceder las entidades que lo conformen. Para proporcionar una adecuada conexión con el repositorio, la cual deberá ser segura, se implementarán las siguientes condiciones dentro de plataforma web, VECsg:

- **Gestión de vulnerabilidades:** las entidades federadas podrán crear vulnerabilidades siguiendo el formato que establece CVE o modificarlas, estando sujetas a ciertas restricciones (ej. mediante reglas predefinidas de políticas de seguridad) para que las transacciones puedan llegar a ser efectuadas. Por ejemplo, para llevar a cabo modificaciones en registros VEC existentes puede surgir la necesidad previa de que el consorcio al completo o su mayoría esté de acuerdo con el proceso de inserción o modificación.

- **Gestión de usuarios:** además de la creación, modificación o borrado de usuarios que podrá llevar a cabo el administrador, mediante RBAC (Role Based Access Control) se definen una serie de roles que van a determinar los permisos de acceso y tratamiento de las vulnerabilidades (ej. consultas y acceso a determinada información). Esto debe ser así puesto que en las infraestructuras críticas participan una serie de entes federados y cada uno desempeña un rol distinto dentro la red de Smart Grid con responsabilidades distintas en dicha red.

- **Servicio de búsqueda de registros VEC específicos dentro del repositorio:** mediante un sistema completo de filtros con numerosas opciones por fechas, dispositivos afectados, resueltas, etc.

- **Servicio de rendición de cuentas (o *accountability* en inglés) y trazabilidad:** dirigido al administrador para que en caso de que se añadan o se modifiquen vulnerabilidades de forma ilícita, éste pueda identificar quién ha originado la manipulación y pueda saber con todo lujo de detalles en qué ha consistido.

- **Presentación correcta de los registros VEC al usuario final:** basada en interfaces intuitivas y fáciles de usar con detalles e información específica de las vulnerabilidades, según los perfiles y roles de los usuarios.

- **Servicios de seguridad:** para garantizar al menos confidencialidad, integridad y disponibilidad de los datos, cuidando la comunicación en TLS v1.3, el almacenamiento seguro en el repositorio, el acceso a la red de Blockchain, y el acceso a la propia plataforma web, teniendo en cuenta las capacidades de OAuth 2.0.

Todo esto con el objetivo de mantener un entorno de registro seguro de vulnerabilidades para que las infraestructuras críticas puedan hacer una gestión accesible y segura de las vulnerabilidades y exposiciones de su consorcio cerrado.

1.3. Estructura del documento

A la largo de este documento se expondrá qué es y en qué consiste la plataforma de gestión segura de vulnerabilidades desarrollada, denominada VECsg. Además se dará contexto a este sistema, definiendo el concepto de Smart Grid y explicando sus tecnologías fundamentales. También, se analizarán las amenazas del sector de la energía para comprender qué motiva a crear un sistema de estas características y se comparará con otras plataformas similares existentes. En los capítulos siguientes se hablará de la metodología elegida para desarrollar un proyecto como este y será justificada y desarrollada, al igual que la planificación seguida para poder completar el sistema en el tiempo dado.

A continuación, en el apartado de **Análisis de requisitos**, se hará un amplio recorrido por todos los requisitos y funcionalidades con las que debe cumplir este sistema, para detallar y definir enteramente, esta especie de contrato a nivel de software que se establece con el cliente.

En la sección de **Diseño y Desarrollo**, se expondrán las decisiones de diseño tomadas para construir VECsg y su arquitectura, se comentarán las fases de desarrollo por las que ha pasado un sistema como este y se tratarán con detalles de la implementación del contrato inteligente y del resto de elementos que componen el sistema. Le sucede una sección de **Pruebas**, donde se recapitularán todas las pruebas automatizadas realizadas al sistema.

Finalmente, se acabará hablando sobre el futuro de este proyecto y la posibilidad de que evolucione a la vez que lo hace la Smart Grid.

En los diferentes anexos que se incluyen al final de la memoria, se pueden encontrar una descripción textual extendida de los casos de uso de los requisitos expresados y también un manual de usuario completo donde se recogen todo lo que se puede hacer con el sistema VECsg, así como los pasos que hay que seguir en cada caso o que se requiere por parte del usuario. En el último apéndice, se ha querido incluir la implementación en detalle de métodos y funciones de las distintas partes que forman la arquitectura de VECsg.

1.4. Glosario de siglas y acrónimos

A continuación, se definen las siglas y acrónimos más frecuentes de esta memoria, claves para entender el contexto del TFG, su planteamiento y construcción.

Tabla 1: Siglas y acrónimos relacionados con tecnologías y herramientas

Acrónimo	Término original	Significado
ADA	*Advanced Distribution Automation*	Automatización Avanzada de la Distribución
AMI	*Advanced Metering Infrastructure*	Infraestructura de Medición Avanzada
CVE	*Common Vulnerabilities and Exposures*	Vulnerabilidades y Exposiciones Comunes
CA	*Certification Authority*	Autoridad Certificadora
DER	*Distributed Energy Resources*	Recursos Energérticos Distibuidos
DSO	*Distribution System Operators*	Operadores del Sistema de Distribución
gRPCS	*Google Remote Procedure Call Secure*	Llamada a Procedimiento Remoto Segura de Google
HTTPS	*HyperText Transfer Protocol Secure*	Protocolo Seguro de Transferencia de Hipertexto
ICS	*Industrial Control System*	Sistema de Control Industrial
JSON	*Javascript Object Notation*	Notación de Objetos de Javascript

Tabla 2: Continuación: Siglas y acrónimos relacionados con tecnologías y herramientas

MISP	*Malware Information Sharing Platform*	Plataforma de Intercambio de Información sobre Software Malicioso
NVD	*National Vulnerability Database*	Base de datos de Vulnerabilidades de Estados Unidos
PLC	*Programmable Logic Controller*	Controlador Lógico Programable
REST	*Representational State Transfer*	Transferencia de Estado Representacional en la Web
RTU	*Remote Terminal Unit*	Unidad Terminal Remota
SCADA	*Supervisory Control And Data Acquisition*	Control de Supervisión y Adquisición de Datos
SMTP	*Simple Mail Transfer Protocol*	Protocolo de Transferencia Simple de Correo
STIX	*Structured Threat Information eXpression*	Expresión Estructurada de Información sobre Amenazas
TAXXI	*Trusted Automated eXchange of Intelligence Information*	Intercambio Automatizado y Confiable de Información de Inteligencia
TCP/IP	*Transfer Control Protocol/ Internet Protocol*	Protocolo de Control de Transmisión/-Protocolo de Internet

Tabla 3: Siglas y acrónimos referidas a organizaciones

Acrónimo	Término original	Significado
CISA	*Cybersecurity and Infrastructure Security Agency*	Agencia Estadounidense de Ciberseguridad y Seguridad de las Infraestructuras
ENISA	*European Union Agency for Network and Information Security*	Agencia Europea de Seguridad de las Redes y de la Información
INCIBE	Instituto Nacional de Ciberseguridad	Organización Española de Prevención y Protección ante Amenazas Cibernéticas
MITRE	*The Mitre Corporation*	Organización de I+D del MIT (Instituto Tecnológico de Massachusetts)
NIST	*National Institute of Standards and Technology*	Instituto Nacional de Estándares y Tecnología
OTAN	*Organización del Tratado del Atlántico Norte*	Alianza militar intergubernamental de los países de Norteamérica y Europa

2

Estado del arte

2.1. Smart Grid

La Smart Grid o red eléctrica inteligente del futuro, es el sistema de generación, transmisión y distribución de energía eléctrica de nueva generación que busca la digitalización de la red y la inclusión de diversas tecnologías software y hardware para mejorar la gestión y la operación de la red. Surge como evolución necesaria del sistema eléctrico tradicional con el objetivo de mejorar la fiabilidad, calidad y eficiencia de este [3]. Esta nueva concepción de red eléctrica apuesta por las energías renovables, consumidores más involucrados en el proceso de generación y conscientes de lo que ocurre en la red, aprovechando el potencial de la información y la comunicación en los dos sentidos dentro del sistema eléctrico. En esta nueva red no solo circula energía eléctrica, sino que los diferentes nodos se conectan para intercambiar datos e información entre ellos permitiendo una mejor gestión y coordinación de toda la red [4].

Aparecen tres tecnologías fundamentales [5] que son las que dotan de sentido a la Smart Grid y hacen que pueda ser una realidad. Por un lado, la Infraestructura de Medición Avanzada (del inglés, *Advanced Metering Infrastructure* (AMI)), que permite que se obtengan grandes cantidades de datos del sistema en tiempo real, por otro los centros de recursos energéticos distribuidos (que en inglés reciben el nombre de *Distributed Energy Resources* (DER)) y que surgen para cambiar la forma en la que se genera la electricidad, en centros tradicionales, centralizados y lejos de las ciudades. Por último, la tecnología de automatización avanzada de la distribución (*Advanced Distribution Automation* en inglés, (ADA)) para gestionar y operar el complejo modelo de red eléctrica de la Smart Grid.

2.1.1. Infraestructura de Medición Avanzada

En la Smart Grid se cuenta con un amplio sistema de mediciones y bases de datos con información actualizada al momento (15-30 minutos), formando la llamada infraestructura de medición avanzada [6], en la cual se permite la comunicación de la información en los dos sentidos para beneficio de todos los miembros de la red eléctrica. El punto de partida son los numerosos electrodomésticos y dispositivos inteligentes del hogar que comunican su consumo con contadores inteligentes o *smart meters* [7]. Estos dispositivos permiten obtener mediciones fiables, incluyen sistemas de detección de robos de electricidad, y son claves para la identificación de fallos y posterior análisis para que no se vuelvan a dar. Luego, esta información es enviada mediante distintas tecnologías de red (Zigbee, Wimax, WAN) [8]. Diversos concentradores de datos intervienen para agrupar la información de diferentes regiones que se redirige al sistema central de telemetría de la Smart Grid. El sistema central usa estos datos para para conocer el estado general de la red, regular la producción de energía en la entidades generadoras de energía o también ofrecer esta información de vuelta a los consumidores una vez analizada. Gracias a ellos, pueden hacer una mejor autogestión del consumo, conociendo datos reales y precisos de los precios del kWh en ese momento y también tienen la oportunidad de acceder a mecanismos como la respuesta a la demanda (del inglés *demand response*).

La respuesta a la demanda [9], es un compromiso que se adquiere con los consumidores, para la gestión activa de la demanda, posibilitado por la existencia de más información que permite una mejor coordinación entre estos y los proveedores. Consiste en que los electrodomésticos y dispositivos se abstengan de consumir electricidad en determinados momentos y lo hagan después. Por ejemplo, en los momentos de sobrecarga de la red eléctrica, para poder gestionar mejor los momentos donde el sistema eléctrico está cerca de su límite y evitar que se sature. Esto es posible gracias a que, en los hogares, naves industriales y resto de recintos, donde se producen estos elevados consumos, existen *smart meters* y electrodomésticos inteligentes (lavadoras, aires acondicionados, ...) que permiten hacer esta gestión activa de la demanda y cortar el consumo cuando, por ejemplo, en la red se producen picos de consumo. A cambio del esfuerzo e inconveniente que provoca al usuario estas limitaciones en el consumo, se acuerdan bonos y descuentos en la factura.

2.1.2. Recursos Energéticos Distribuidos

Ahora los recursos de generación no se van a concentrar en centrales de energía eléctrica a kilómetros de distancia de las ciudades, con la consecuente perdida de energía en los largos trayectos de transporte o la posibilidad de que estas grandes líneas de transmisión resulten dañadas y haya cortes en el suministro. Sino que, los recursos energéticos (generación y almacenamiento) van a estar distribuidos, situándose cerca de donde va a ser consumida la energía, apostando sobre todo por energías renovables como la eólica o placas y calentadores solares [10]. Estos recursos pueden estar organizados en torno a *microgrids*, que son conjuntos de unidades distribuidas de producción y almacenamiento eléctrico, controlados y gestionados de forma autónoma, los cuales se pueden aislar de la red eléctrica y funcionar independientemente. Esto permite conseguir el aislamiento suficiente, para proteger al área local de grandes cortes en la red general, también, se pueden operar mediante una planta de energía virtual (VPP del inglés *Virtual Power Plant*), que es un sistema de gestión de recursos distribuido, mucho más versátil que esta basado principalmente en software y no es tan dependiente de la infraestructura. Es decir, el software mediante un arquitectura web segura, agrega distintos elementos de la red, que no tienen porque estar especialmente cerca, en una sola central eléctrica virtual que es operada por este sistema para dar suministro a un área menos localizada que las *microgrids*. Estas formas de agregación, permiten agrupar la suficiente capacidad productiva como para competir con las centrales eléctricas tradicionales, pudiendo satisfacer el área local, e incluso, aportar energía a la red [11]. De este modo, se apuesta por la generación compartida y la redundancia de la energía, para evitar la fuerte dependencia con los grandes generadores de energía (centrales de carbón, nucleares, hidroeléctricas, ...), la consecuencia de esto, es que la red eléctrica queda mucho más equilibrada, en cuanto al pulso entre oferta y demanda, y es más fácil de gestionar, lo que al final se traduce en una eficiencia mejorada y menos costes [12].

Gracias a los DER, se dice que en esta red la energía puede tener dos sentidos, es decir, los consumidores ejercen también de productores, ya que pueden colaborar en la red adquiriendo dispositivos de generación energética para particulares y agregándose a una *microgrid*, a través de la cual vender su energía a la Smart Grid [13]. Otro de los

aspectos clave es el almacenamiento de energía distribuido, ya que los usuarios de la red pueden construir sus propios sistemas de baterías para aportar electricidad al sistema en momentos donde esta sea más valiosa (tiempos de pico de consumo), o incluso, se pueden utilizar como almacén de energía las baterías de los coches eléctricos [14], que cada día están más presentes en nuestras carreteras [15], ya que la ciudadanía se suma a un cambio que es esencial para que el transporte sea más eficiente y ecológico. En esto coinciden organizaciones supranacionales, como la Unión Europea, que ha anunciado medidas, tales como la prohibición de fabricar vehículos de motor de combustión en 2035 [16], en pos de la descarbonización de los transportes y la movilidad sostenible. En cualquier caso, en la Smar Grid propone este almacenaje distribuido como otra manera de reducir la carga de la red y apoyar a energías más inestables y variables como la solar, la eólica, etc. por la posibilidad que ofrece de recurrir a la energía almacenada cuando se requiera [17] [18].

2.1.3. Automatización Avanzada de la Distribución

La tecnología ADA, llevada a cabo por los nuevos operadores del sistema de distribución (los llamados DSO -del ingles *distribution system operators*-) [19] y en el centro de control, consiste en automatizar la operación de la red eléctrica de medio y bajo voltaje, gracias a la información en tiempo real de todo el sistema que se tiene gracias a la AMI. Puesto que son muchos los nuevos flujos de corriente eléctrica y los nodos que aparecen en la Smart Grid debido a la gran descentralización de la producción de los DER, esta automatización es muy necesaria para poder llevar a cabo una gestión eficiente en este complicado planteamiento de distribución eléctrica. Entre los retos a cumplir se encuentra la operación inteligente de la red e incluir en ella la energía aportada por las heterogéneas y dispersas *microgrids* [20]. Uno de los sistema con los que se espera poder tener este control sobre los datos y que permita ejecutar acciones de operación consecuentes, es el potente sistema de adquisición de datos y control de supervisión, SCADA (*Supervisory Control And Data Acquisition*) [21]. Este es un software que ya cuenta con años de implantación en el sector industrial y energético, que permite el control remoto y la automatización de acciones, en estaciones como las de distribución o transmisión, pero habría que ampliar su alcance para no solo abarcar estos sistemas tradicionales, sino también aprovechar todo su potencial sumando a las *microgrids*, los edificios, etc. [22]

Con este nuevo modelo, también se pretende facilitar el mantenimiento de la red y que esta sea mucho más resiliente, porque la automatización va a permitir que a partir de la información recogida de los numerosos medidores y contadores inteligentes distribuidos por toda la red, se tomen decisiones automatizadas desde el organismo de control para optimizar la gestión y el equilibrio de la red, además de dotarla de la capacidad de autorrecuperarse. Estas decisiones bien informadas pueden estar relacionadas con respuestas a picos de consumo basadas en predicciones de patrones observados en los consumidores o con soluciones a errores y fallos en los elementos de la red [23]. Estos errores se pueden detectar más fácilmente gracias a la enorme cantidad de medidores que están enviando información del sistema constantemente y se pueden solventar con mecanismos como la reconducción del flujo de suministro eléctrico por otras líneas para que el servicio no se interrumpa o en muchos casos ese error podrá ser tratado telemáticamente sin que los operarios tengan que desplazarse físicamente a solucionar el problema. Todo ello, gracias al nuevo planteamiento tecnológico de telegestión apoyado por nuevos dispositivos que trabajan de forma coordinada como los relés inteligentes o los interruptores remotos automatizados [24].

2.1.4. Estructura de la Smart Grid

Como hemos visto en la sección anterior, en una red de Smart Grid conviven una serie de entidades diferentes que realizan roles específicos y bien definidos, con la posibilidad de conformar un consorcio federado para la buena gestión de servicios de control y distribución de energía a los usuarios finales. En esta sección, volvemos a nombrar algunas de ellas, pero esta vez mostrando las relaciones entre dichas entidades y en base al modelo conceptual de referencia (figura 1) [25] propuesto por el NIST (que en inglés, es el *National Institute of Standards and Technology* de Estados Unidos) [26]:

-Centro de control: es la institución que hace de director de orquesta del resto de entidades, puesto que por ella pasan todos los datos que le proporcionan un visión global del sistema en tiempo real, y se pueda operar de forma precisa y automatizada un sistema tan complejo y distribuido como la Smart Grid.

-Sistema de generación de energía, incluyendo también los DER: se trata de las grandes estaciones de generación de energía tanto las tradicionales (centrales de carbón,

de gas, nucleares, …), como las renovables (centros fotovoltaicos, eólicos, de energía solar térmica, ...) que hasta ahora son las que dotan de energía eléctrica a la red. Pero, en la red eléctrica inteligente también entran a jugar un papel fundamental los recursos energéticos distribuidos (DER) como productores de energía. Estas nuevas entidades introducen una serie de retos técnicos (tales como incluir un flujo en sentido contrario en la red tradicional o la regulación equilibrada del precio de la energía producida en los DER con la producción convencional), que son necesarios resolver para permitir la integración plena de los recursos energéticos distribuidos, en la red [27].

-Sistemas de transmisión y de distribución: son infraestructuras críticas que se encargan del transporte de energía de alto y medio voltaje respectivamente. Con la digitalización y la conexión de numerosos dispositivos IoT, va a mejorar la monitorización ante errores y caídas, y la operación de estas infraestructuras. Pero sobre todo, va a adquirir gran importancia el sistema de distribución sobre el de transmisión, gracias a los DER. Si en la red tradicional, la energía solo tenía un sentido desde las centrales en las que se generaba hasta los consumidores, con los sistemas de transmisión y distribución ejerciendo el papel de transportistas, en la Smart Grid, la electricidad puede circular en los dos sentidos, al poder satisfacerse la demanda energética generando electricidad cerca de donde se consume. Y para este caso, el sistema más cercano a los consumidores es el de distribución, por tanto, la exigencia y el desempeño de este sistema va a ser mayor. También, los centros de control han de aumentar su labor operativa en esta parte de la red y surgen entidades como los DSO que deberán colaborar con los DER en las áreas locales para gestionar la energía generada ahí [28].

-Empresas proveedoras de servicios: dentro de este dominio se encuentran las empresas que proporcionan servicios eléctricos y dan acceso al suministro a los clientes, pero no se quedan ahí, ya que son las que posibilitan a los consumidores ser parte activa de la red gracias a la AMI. Para ello, ofrecen a los clientes mecanismos para la gestión de su energía, permitiéndoles conocer en tiempo real los datos del precio de la electricidad. A su vez, plantean nuevas opciones en el contrato como la *demand response* que, como ya explicamos, a cambio de descuentos en la factura, el consumo del cliente puede ser limitado y pospuesto a otro momento en el tiempo, si por ejemplo, las lecturas indican que la red está saturada, permitiendo de esta manera que la carga dentro de la red eléctrica

se balancee correctamente. Igualmente, se encargan de montar y mantener la instalación necesaria para la Smart Grid, que trae consigo nuevas oportunidades de negocio con la aparición de los dispositivos inteligentes de última generación como los contadores o los electrodomésticos, más eficientes y cuyo consumo es mucho más fácil de gestionar.

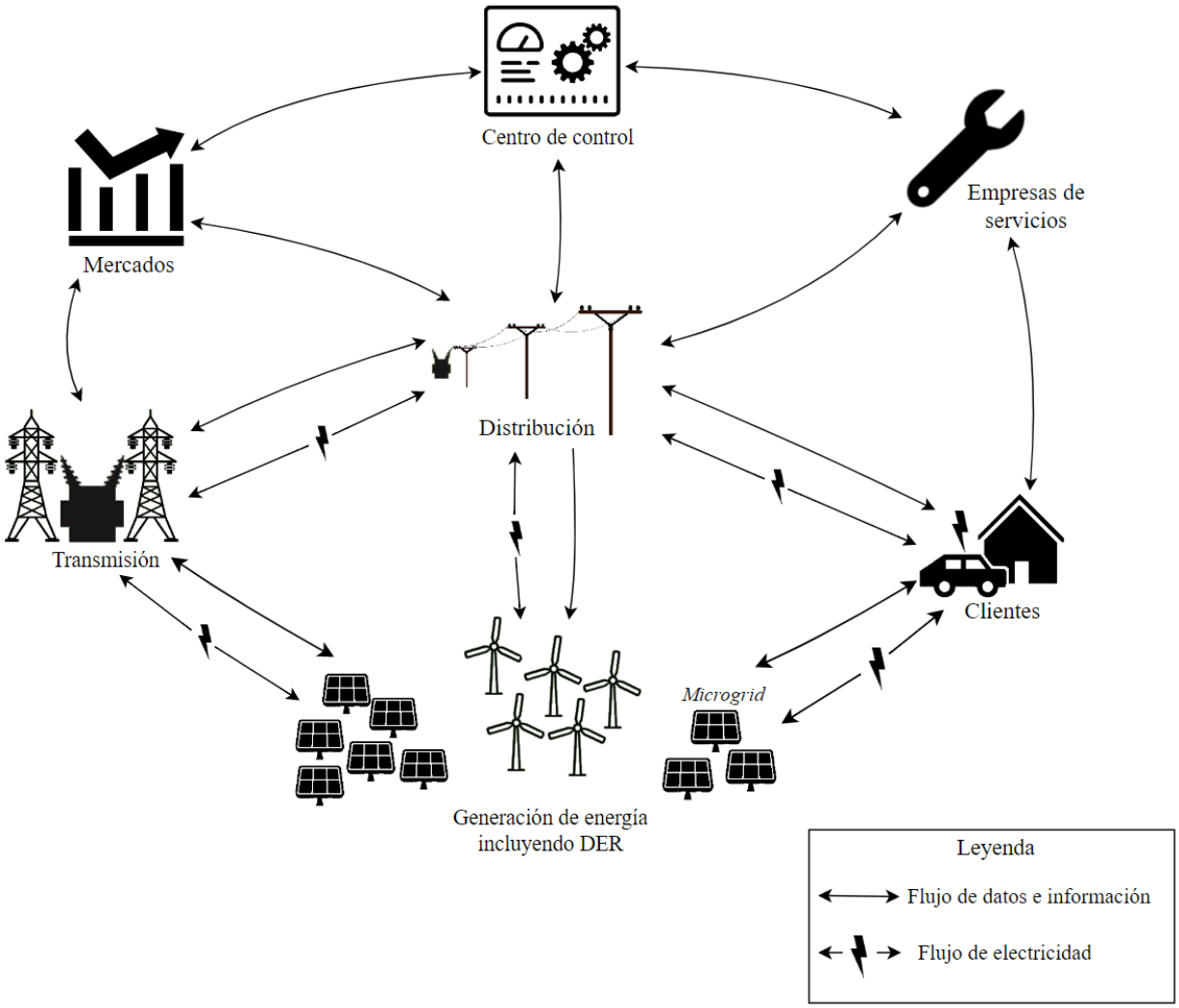
-Mercados: este es el dominio donde principalmente se comercializa la energía eléctrica, marcando precios basándose en datos veraces y actualizados de la red. En los mercados también se abre paso a los DER, permitiendo que se agreguen y combinen como estructuras únicas para aportar capacidad productiva del orden de decenas de MW (megavatios) a la red y convivir en el mercado junto con las fuentes de generación tradicionales de mayor capacidad (térmica, nuclear, hidroeléctrica, etc.).

-Clientes: son los grandes beneficiados de este sistema al adquirir un rol más relevante, al aprovecharse de flujo de información en dos direcciones para tener datos de precios más afinados en tiempo real y poder acceder a la energía según nuevos planes de consumo como la *demand response* que beneficia tanto a ellos como a fla red. También, es posible poder aportar con su energía producida o la almacenada en los coches eléctricos de sus propias ubicaciones y hogares.

Como hemos visto, las ventajas de la Smart Grid son muchas. Sin embargo, afronta problemas a los que hay que prestar especial atención, que ya hemos mencionado, como es el de la ciberseguridad [29]. Al contar con miles de dispositivos conectados, existe una gran colección de datos que administrar y controlar adecuadamente para preservar la confidencialidad e integridad de los mismos, así como, para proteger derechos básicos de los usuarios individuales, tales como, su privacidad [30]. Esta se puede ver ampliamente afectada ante la gran cantidad de datos sobre formas y hábitos de consumo de los consumidores que se recogen en los contadores inteligentes que se implantan en sus hogares.

Además de la violación y manipulación maliciosa que se puede hacer de estos datos, en la Smart Grid entran en juego una gran cantidad de dispositivos inteligentes a los que los ciberdelincuentes pueden acceder remotamente y mandarles que ejecuten acciones no legítimas, por ejemplo, para abrir circuitos enteros de la red eléctrica. Un grave error sería permitir que individuos malintencionados tuviesen capacidad de controlar sistemas software de tipo SCADA [31] usados para la operación de la red.

Figura 1: Estructura de elementos de la Smart Grid inspirada en el modelo del NIST



Dado el caso, podrían adquirir el poder de manejar con total libertad el sistema eléctrico y ocasionar problemas severos en la Smart Grid, desde cortes temporales del suministro, hasta llegar a sabotear y dejar fuera de servicio la infraestructura física. También, podrían atacar a los propios dispositivos conectados en la Smart Grid, con la ayuda de algún comando de tipo *nmap* [32] o algún software que escanee las vulnerabilidades de todo el sistema —por ejemplo, herramientas de escáner como Nessus [33] u OpenVAS [34]— puedan extraer agujeros de seguridad en los dispositivos y al explotarlos consigan provocar efectos críticos como averías, comportamientos anómalos o la destrucción total de estos sistemas, en los que existen controladores y operadores de la red, aparatos para

el control de la seguridad de las infraestructuras, dispositivos auxiliares, etc.

2.2. Amenazas en el sector de la energía

El sector energético es uno de los más perseguidos por entidades maliciosas de nivel avanzado. Al tratarse de un sector crítico para que millones de personas puedan realizar su vida diaria con normalidad y del que dependen otros sectores como el industrial, el de transportes o el de la salud, las motivaciones para atacar y sabotear la red eléctrica son varias: robo de datos, control de sistemas críticos, presiones a gobiernos, muestras de poder o propaganda del miedo, entre otras razones. Tal y como muestran los informes, durante el 2021 [35], el sector de la energía fue el tercero más atacado, quedando al mismo nivel que el industrial. Y todo indica que, estos riesgos de seguridad y amenazas podrían aumentar debido a los pasos que hay que dar para evolucionar hacia una red eléctrica más inteligente e interconectada. El hecho de que se modernice va a permitir que, por un lado, los dispositivos desactualizados y anticuados que se siguen utilizando, los conocidos como dispositivos *legacy* o heredados, se sustituyan por otros más seguros. Pero, por otro lado, la digitalización del sistema eléctrico y de toda la cadena de suministro hace que aparezcan nuevos puntos de ataque y que la gestión de la seguridad sea más complicada debido a las nuevas operaciones eléctricas que se informatizan [36]. En este punto, también entran en juego tecnologías más disruptivas, con algoritmos y redes neuronales de inteligencia artificial, y dispositivos más innovadores, fundamentales en el funcionamiento de la avanzada Smart Grid. Estos, al ser nuevos, no han podido ser probados durante tanto tiempo en condiciones de trabajo reales. Por tanto, pueden surgir muchas vulnerabilidades desconocidas hasta el momento, las llamadas vulnerabilidades de día cero (del inglés, *zero-day*), las cuales pueden ser descubiertas y aprovechadas por los ciberdelincuentes. En muchos casos, estas vulnerabilidades suelen permanecer durante meses en manos de los atacantes, sin ser conocidas ni resueltas hasta que los responsables del sistema las descubren. Durante este tiempo, individuos malintencionados pueden haber estado tomando ventaja de esta brecha de seguridad, explotándola o obteniendo beneficios económicos al vender ilegalmente este conocimiento oculto de como vulnerar un sistema [37]. Por eso, es necesario contar con un plan avanzado y unificado de seguridad, que tome también en cuenta estás vulnerabilidades de día cero, aunque el esfuerzo para llevarlo a cabo en

un sistema de estas características sea mayor (en funcionamiento las 24 horas del día, componentes del sistema eléctrico distribuidos geográficamente y muy dependientes ante fallos, ...), es fundamental para el correcto funcionamiento de la red eléctrica en el modelo de sociedad actual [38].

Entre los ataques y malware más comunes en el sector de la electricidad [39] podemos encontrar los famosos troyanos de acceso remoto que permiten al atacante tomar el control de equipos a distancia abriendo una puerta trasera o *backdoor*, el spyware utilizado para robar datos e información crítica de forma sigilosa. Otro tipo de programa malicioso muy extendido, es el ransomware que deja a los sistemas informáticos inutilizables y exige como rescate grandes sumas de dinero para desencriptar los datos y que esa máquina pueda seguir ejerciendo sus funciones de control u operación de la red con normalidad. También, cabe destacar los ataques de ingeniería social o de phishing. Este tipo de ataques consisten en el envío de correos falsos con apariencia de auténticos a instituciones importantes de la red eléctrica, los cuales contienen direcciones webs infectadas o adjuntos que incluyen otro malware que se puede instalar de forma silenciosa en estos equipos críticos. Tales como troyanos de acceso remoto, spyware, ransomware e incluso gusanos que pueden infectar a toda la lista de contactos de email, o incluso, alcanzar una red LAN con dispositivos de control industrial (de tipo PLC —del inglés *Programmable Logic Controller*— o RTU —cuyas siglas significan *Remote Terminal Unit*—) e infectarlos a todos haciendo que modifiquen su funcionamiento normal. Como ocurrió en el caso de Stuxnet en 2009, en el que se infectaron los controladores de tipo PLC de centrifugadoras nucleares de una planta iraní, con el objetivo de destruirlas, donde el gusano se insertó a través de una memoria USB [40].

En el mundo de los dispositivos embebidos o IoT [41] como los *smart meters*, no está libre de malware, sin embargo, al contar con unas características particulares, son explotados de forma distinta que los ordenadores personales. Son dispositivos que funcionan con firmware por lo que hay que gestionar adecuadamente sus actualizaciones y verificar que no se instalan parches de firmware fraudulentos. Al ser dispositivos que tienen una potencia computacional reducida, se pueden provocar ataques de denegación de servicio y afectar a sus factores físicos limitados como la batería [42]. Además, en función del protocolo de red que utilicen se puede tener acceso a los datos que manejan y captan

con sus sensores aunque para ello normalmente hay que estar físicamente cerca de los dispositivos. Los dispositivos IoT sufren una vulnerabilidad muy extendida que los hace muy propensos a que sean infectados por algún malware, como son las configuraciones insuficientes de seguridad y la baja calidad de sus contraseñas [43]. Por ejemplo, el gusano *Mirai* [44] aprovecha esta vulnerabilidad y los convierte en dispositivos secuestrados que usa para colaborar en un ataque de denegación de servicio distribuido, simplemente aprovechando estas configuraciones vulnerables, como que se mantengan las credenciales por defecto del dispositivo.

2.3. Ataques avanzados al sistema eléctrico

Uno de los grandes ataques a infraestructuras eléctricas fue el sufrido por Ucrania a finales de 2015 [45], que tuvo como consecuencia que casi 250.000 personas se quedaron sin suministro eléctrico durante varias horas. El ataque se relaciona principalmente con un grupo de hackers rusos que forman parte del cuerpo militar de inteligencia del gobierno del mismo país, conocidos como *Sandworm* [46]. Un ataque de tal magnitud fue preparado durante años, con acciones previas de espionaje y robo de datos críticos e identidades. El grueso del ataque se realizó gracias al malware *BlackEnergy*, un troyano de acceso remoto adaptado para sistemas de control industrial (conocidos en inglés como ICS o *Industrial Control Systems*) [47]. Este permitió acceder de forma remota a los dispositivos críticos infectados con este backdoor, estableciendo una comunicación mediante peticiones POST de HTTP con servidores manejados y controlados por los atacantes (conocidos en inglés como *Command & Control servers* [48]). De este modo, tenían acceso a los centros de control de distintas redes eléctricas y usando los sistemas SCADA, fueron apagando los circuitos y cortando el suministro de gran parte del oeste de Ucrania en apenas media hora [49]. Además, aplicaron otro software malicioso llamado *KillDisk* con un propósito muy concreto, borrar por completo los discos duros de los ordenadores infectados, con el objetivo de eliminar información crítica y dificultar lo máximo posible la recuperación del ataque [50].

En este ataque de gran complejidad de ejecución se aprovecharon de numerosas vulnerabilidades de los dispositivos informáticos de aquel momento. Como, por ejemplo, es el caso de la vulnerabilidad CVE-2014-4114 [51] (descrita según el estándar de nomencla-

tura de descripción de vulnerabilidades CVE —que en inglés se conocen como *Common Vulnerabilities and Exposures* [52]—), que afectaba al software de *Microsoft Office* y que mediante la solicitud al usuario de ejecución de macros, si éste la aceptaba, se desencadenaba la instalación y la ejecución de scripts maliciosos en el dispositivo. En agosto de 2014, más de un año antes de que se llevase a cabo el sabotaje, iniciaron una campaña de phishing donde se hacían pasar por entidades legítimas para enviar documentos de *Power Point* corruptos y aprovechando esta vulnerabilidad, instalar encubierto, el software *BlackEnergy* en todos los sistemas importantes de control de la red eléctrica para permanecer latente hasta el día del ataque, el 23 de diciembre de 2015 cuando el oeste de Ucrania quedó en la penumbra.

En este ciberataque de gran magnitud, también fueron objetivos de ataque los ordenadores embebidos de *Moxa*, de la serie UC 7408 [53] que hacían de conversores a Ethernet. Estos dispositivos tenían vulnerabilidades conocidas y registradas en la NVD (en inglés *National Vulnerability Database*) de Estados Unidos [54] como el CVE-2014-6271[55], que permitía a los hackers ejecutar código malicioso de forma remota en estos dispositivos o las vulnerabilidades CVE-2014-7186 [56] y CVE-2014-7187 [57] que abrían la posibilidad de ejecutar un ataque de denegación de servicio sobre estos sistemas embebidos. Esto fue aprovechado por los ciberdelincuentes para cortar la conexión remota con las subestaciones y que desde el centro de control no pudiesen mandar instrucciones de recuperación y reparación, teniendo que trasladarse físicamente a cada subestación para volver a levantar el circuito. Como se puede observar, eran vulnerabilidades que se conocían y se sabía de su existencia un año antes de que se produjera el ataque sobre Ucrania, por lo que si se hubiese realizado la labor de comprobar estos dispositivos y mantener los parches de firmware actualizados se podría haber evitado parte de la catástrofe.

Otros ataques de este tipo con un alcance mortal, se pueden realizar gracias a malware como *Trisis/Triton* [58] que busca anular los sistemas de seguridad de las infraestructuras (SIS) (por sus siglas en inglés -*Safety Instrumented System*-) los cuales se encargan de monitorizar el estado actual de la plantas y comprobar si se alcanzan niveles de medición que hagan saltar las alarmas. En ese caso entran en acción e intentan recuperar el balance normal o apagar los sistemas físicos manteniendo la seguridad de las instalaciones y las personas [59]. Este ataque fue puesto en práctica en agosto de 2017, para intentar destruir

una planta petroquímica de Arabia Saudí, saboteando los sistemas de seguridad con el malware *Trisis/Triton* y luego hacer explotar las máquinas con otro software de tipo troyano — aunque estas pretensiones no se llegaron a cumplir porque el malware actuó de forma anómala y dió indicios de su existencia en el sistema. *Triton* funcionaba explotando una vulnerabilidad de día cero del sistema SIS de *Schneider Electric* denominado *Triconex* que permitía una escala de privilegios en este dispositivo para conseguir su control total, como se describe en el registro CVE que se publicaría tiempo después de este ataque (CVE-2018-7522) [60]. Aunque este primer intento de hacer un ataque en gran escala usando el malware *Trisis* resultara fallido, no quiere decir que el problema haya desaparecido, puesto que siguen en funcionamiento un gran números de dispositivos desactualizados que son vulnerables a la exposición CVE-2018-7522. También, es posible que los ciberdelincuentes aprendan de los errores y desarrollen versiones más modernas con las que realizar ataques futuros con más efectividad y adapten el malware para ser aplicado a otras industrias como la energética [61].

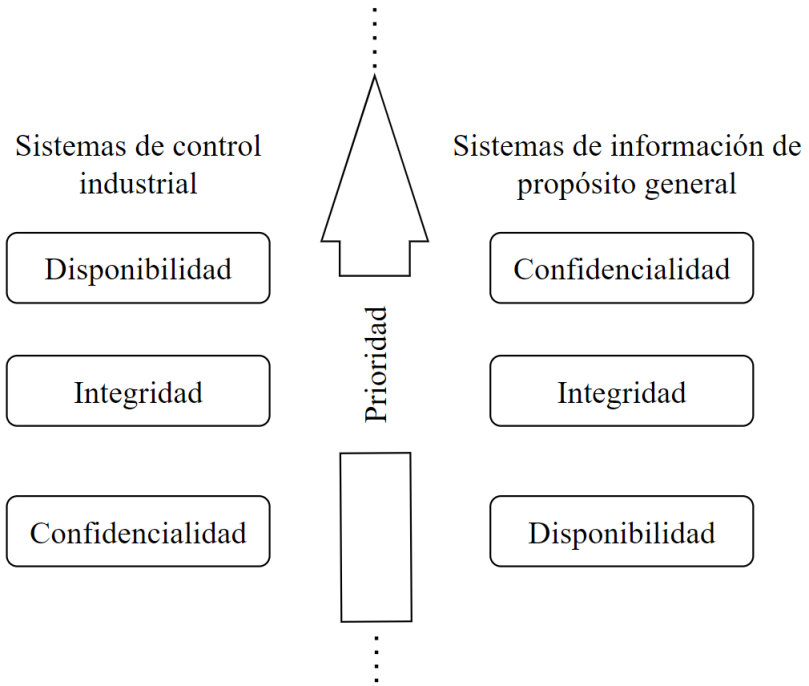
2.4. Gestión de datos en la Smart Grid

Garantizar la seguridad en un entorno de Smart Grid es complicado porque surgen nuevos factores y hay que adaptar nuevos roles, en una red que se redefine y adquiere un enfoque más óptimo, eficiente e informatizado. La Smart Grid debe considerarse como una estructura de dos caras en cuanto al tratamiento de los datos, a causa de su naturaleza. Por un lado, actúa como un sistema de control industrial donde, la prioridad debe ser asegurar que los datos de control de la infraestructura física están siempre disponibles para que la red pueda ser operada con normalidad y sin interrupciones. De modo que otros aspectos como la confidencialidad de los datos no se cuida tanto, intentando reducir las operaciones realizadas sobre ellos que limiten su disponibilidad, como puede ser el cifrado. Esto es radicalmente diferente cuando consideramos la parte de la Smart Grid en la que se tratan datos de usuarios recogidos, por ejemplo, en los *smart meters*. En dicho caso, la Smart Grid se considera desde el punto de vista de un sistema de información de propósito general, donde, lo más relevante es que los datos de los usuarios estén lo más seguros posible para preservar su privacidad e intimidad. Otra preocupación que puede darse, es la integridad de los datos, por ejemplo que cuando se consulten históricos de

consumo, no se hayan modificado los datos del usuario sin ninguna justificación, cosa que podría confundir e indignar al cliente.

Estas grandes diferencias que hay dentro de la misma red según el enfoque de prioridad que se le dé a los datos se materializa en la guía de políticas de seguridad conocido como CIA (del lenguaje anglosajón, Confidentiality, Integrity and Availability) [62]. De modo que, cuando el foco se encuentra en el tratamiento de datos de los usuarios, se sigue justo el orden de prioridad de la tríada CIA — confidencialidad, integridad y disponibilidad. Sin embargo, cuando se ve la Smart Grid como un sistema de control industrial automatizado, la tríada se invierte (AIC) y las prioridades cambian completamente (figura 2). Estas diversas formas de planteamiento que se observan y están integradas a la vez en este sistema, demuestran lo compleja y versátil que es la Smart Grid y lo mucho que se complica y especializa la gestión de su seguridad [62].

Figura 2: Modelo CIA/AIC de protección ante amenazas cibernéticas



2.5. Relación entre ataques y vulnerabilidades del sistema

De los numerosos ataques sobre la red eléctrica que se han tratado, se puede entrever que muchas de las vulnerabilidades estaban ya registradas en repositorios públicos y eran conocidas. De modo que, si se hubiese cumplido con la responsabilidad de estudiar estas vulnerabilidades en el sistema propio y se hubiera actuado con cautela con los dispositivos afectados, se podría haber opuesto una gran barrera de defensa a los ataques derivados de estas brechas de seguridad. Otra situación observada, es que los ciberdelincuentes utilizan vulnerabilidades no conocidas (*zero-day*) que comprometen gravemente la seguridad de los dispositivos. Al no saberse nada de ellas, se debe hacer el trabajo de aprender y adquirir conocimiento de la experiencia del ataque y aplicar las medidas de protección adecuadas ante este tipo de ataques tan particular y dañino. Intentando realizar predicciones bien informadas de cuales van a ser las próximas vulnerabilidades de día cero explotadas y así, conseguir anticiparse a los movimientos inesperados del ciberdelincuente. En estos casos, es muy conveniente que las organizaciones añadan en fuentes de conocimiento compartidas las vulnerabilidades *zero-day* descubiertas para que se sepa de su existencia de forma generalizada y se encuentre una solución lo antes posible a estas brechas de seguridad críticas.

Llegados a este punto, se puede apreciar que es un esfuerzo útil mantener los equipos con las últimas versiones y parches de seguridad, realizar un análisis profundo para monitorizar todas las vulnerabilidades existentes en el sistema, incluso investigando alguna vulnerabilidad *zero-day* nueva que se pueda descubrir, y que la comunidad y ciertos agentes añadan actualizaciones y novedades que se vayan descubriendo de esa vulnerabilidad, para que la información sea mucho más completa y valiosa. Poniendo en valor, el hecho de que se comparta entre organizaciones información relevante sobre vulnerabilidades para que esta sea contrastada y verificada y se pueda utilizar este conocimiento aumentado para mejorar la protección de la organización.

2.5.1. Repositorio de vulnerabilidades como protección contra ataques

La importancia de tener un repositorio fiable y privado de vulnerabilidades reside en la posibilidad que ofrece a las organizaciones de tener siempre al alcance una fuente de

conocimiento sobre debilidades de los sistemas de información. Agencias de seguridad como la CISA [63], encargada de la ciberseguridad y la protección de infraestructuras de Estados Unidos, se sirven de las vulnerabilidades que se comparten en estos listados abiertos, para crear sus propias listas de vulnerabilidades peligrosas, como el catálogo de vulnerabilidades explotadas conocidas de la directiva del CISA 22-01 (*BOD 22-01 Catalog of Known Exploited Vulnerabilities*) [64] que elaboran como parte de sus planes de defensa ante ataques y amenazas cibernéticas. Gracias a que se ponen en conocimiento general vulnerabilidades CVE, este tipo de organizaciones pueden valorar y detectar si son realmente críticas. Dado el caso, las añaden a sus propios catálogos de vulnerabilidades y se toman acciones contra las empresas afectadas. Por ejemplo, presionándolas para que pongan solución a estas brechas e incluso llegando a cortar todo el tráfico desde entidades federadas a estos sitios vulnerables como ocurrió con *Atlassian* [65], a principios de junio de 2022, debido a una vulnerabilidad de alto grado de peligrosidad. Todo con el objetivo de no ignorar estas advertencias y posibles ataques explotables que toman la forma de registros CVE, para aspirar al máximo nivel posible de seguridad y protección de los sistemas informáticos.

2.6. Aplicaciones similares de gestión de vulnerabilidades

En la actualidad existen otros sistemas que buscan gestionar y compartir vulnerabilidades con la comunidad como es el caso de la plataforma de CVE de MITRE [66], que expone al público vulnerabilidades y exposiciones comunes usando un formato común y reconocible (los llamados registros CVE), una vez que estos errores han sido comunicados y tratados por los fabricantes y organizaciones afectadas. Se trata de una plataforma que tiene un amplio recorrido y es fuente fiable y bien reconocida de fallos que afectan a la seguridad del hardware y el software, entre hackers y la comunidad afectada por estas vulnerabilidades.

Otra organización que también se preocupa por mantener un listado amplio y actualizado de vulnerabilidades informáticas es el NIST, que en su base de datos, NVD, desarrolla con algo más de detalle registros del CVE de MITRE añadiendo parámetros más precisos e interesantes sobre el alcance, el origen, los privilegios necesarios, etc. de la vulnerabilidad y mostrando la información de cada registro CVE más ordenada y clara

para encontrar fácilmente comunicados de las instituciones afectas o ejemplos de *exploits* de la vulnerabilidad.

Otra herramienta que igualmente se dedica a dar a conocer vulnerabilidades, es la opción de *Vulnerabilidades* para la *Alerta temprana* del INCIBE [67]. Desde aquí, el instituto de ciberseguridad español ofrece las vulnerabilidades con el modelo de nomenclatura de CVE, pero traducidas al español para facilitar la difusión de estas debilidades de los sistemas entre la comunidad de hispanohablantes. Encontraremos en esta página las características principales de cada vulnerabilidad bien organizadas y con una buena visualización, además de contar con un sistema de filtros completo y preciso de registros CVE.

Estas plataformas han servido de inspiración y han dejado su huella en el sistema VECsg, algunas de ellas como la NVD del NIST nos sirven como fuente de datos o también se puede apreciar en el formato tan similar de los registros VEC con los que MITRE difunde en su plataforma de CVE o la capacidad que ofrece INCIBE con su herramienta *Vulnerabilidades/AlertaTemprana* para encontrar numerosos registros CVE con información ampliada mediante avanzados filtros con diversas opciones de búsqueda. Pero en VECsg se busca ampliar toda esta funcionalidad para hacerla más privada y segura para las organizaciones de la Smart Grid gracias a las ventajas que ofrece la tecnología blockchain en cuanto a inmutabilidad, integridad y seguridad de los datos.

3

Metodología y planificación

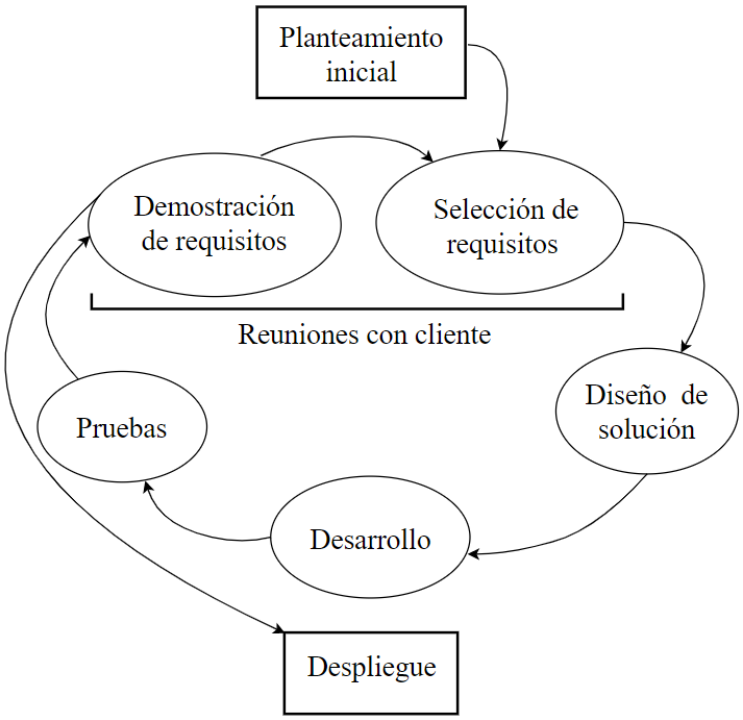
En este capítulo se va a describir cuál ha sido la organización temporal y las fases de la metodología elegida, en las que se ha dividido todo el trabajo necesario para producir un sistema como VECsg.

3.1. Metodología de desarrollo iterativa e incremental

Tras observar con detenimiento las características de este proyecto (mediana dimensión, equipo individual, disponibilidad del cliente, ...) se tomó la decisión de llevar a cabo una metodología de desarrollo iterativa e incremental, basada en *sprints*. La intención que motivó esta decisión fue que el proyecto creciera y mejorase en sucesivos ciclos al final de los cuales el cliente realiza una valoración de lo construido hasta ese momento. En este TFG, la tutora ha ejercido ese rol de cliente y con ella se ha contactado reuniones telemáticas al final de cada *sprint* para realizar dicha revisión.

La forma de organizar el desarrollo del sistema software ha sido clara y bien pautada desde el inicio (figura 3). En un primer momento se llevaron a cabo varias reuniones iniciales con el cliente para definir en qué iba a consistir el proyecto y fijar su alcance. De ellas, se obtuvo una lista de requisitos generales de la aplicación (se puede encontrar un análisis profundo en la sección 4.1). A partir de aquí, es cuando empieza el primer *sprint*. De todos los requisitos planteados fue necesario seleccionar los más prioritarios y resolver todas las dudas con el cliente que se tuviesen sobre la definición de ellos. Entonces, se procedía a diseñar la solución tomando en cuenta todas las restricciones y exigencias de seguridad, confidencialidad y disponibilidad del sistema.

Figura 3: Esquema de la metodología iterativo e incremental implementada



El *sprint* continuaba con el desarrollo de software y las pruebas para comprobar que los requisitos definidos y elegidos para el *sprint* estuviesen implementados correctamente. Al final del *sprint* se producía una revisión de los requisitos con el cliente, mostrándole lo implementado hasta el momento para ver si se satisfacían sus necesidades. Gracias a que se contaba con los comentarios y el *feedback* del cliente al final de cada *sprint* se podía verificar si este se expresó correctamente y si como analistas, fuimos lo suficientemente precisos a la hora de tomar estos requisitos. Dado el caso, se realizaban en el *sprint* siguiente los ajustes necesarios para corregir en la implementación, las suposiciones incorrectas o malentendidos que podían ocurrir.

Una vez que el cliente daba el visto bueno se daba por terminado el *sprint* y a partir del producto desarrollado hasta ese momento se iniciaba un nuevo ciclo completo repitiendo todos los pasos descritos hasta ahora. Así, tras varias iteraciones el sistema se iba completando y desarrollando incrementalmente. En cuanto todos los requisitos definidos al principio fueron aplicados al sistema y se tuvo la aprobación del cliente, tal y como se acordó, el sistema es dado por finalizado. De modo que ya solo queda el paso final de despliegue e instalación del software en la plataforma del cliente, que tras la entrega

del software tendrá que llevar a cabo el propio cliente para lo que se incluye entre los entregables un manual de instalación. Junto con el producto software, se ofrece además un servicio de mantenimiento y resolución de dudas sobre el sistema durante varios meses después de la entrega del software.

3.2. Planificación basada en diagramas de Gantt

Esta planificación se realizó con diagramas de Gantt porque permite ordenar temporalmente las tareas y expresar de forma muy clara las dependencias entre tareas, de manera que se puede deducir fácilmente cual es el camino crítico de tareas a realizar que fijan el límite inferior de tiempo, lo que al menos se va a tardar en completar el sistema.

La planificación temporal ha sido resultado de varias iteraciones, ya que, se iba actualizando y concretando conforme se acercaba cada *sprint*. Poco antes de que estos empezaran, se colocaban las tareas que habían sido más detalladas y se estimaban sus duraciones en ese momento en el que se contaba con la mayor precisión posible.

A continuación, se puede observar la planificación temporal realizada mediante varios diagramas de Gantt divididos por *sprints*, desde el primero (*sprint* 0), donde se realizan todas las configuraciones iniciales, hasta el *sprint* 6, donde se realizan sobre todo tareas de optimización y mejora del sistema.

En estos diagramas, se puede ver en el eje horizontal la progresión temporal de las tareas que se detallan en la columna de la parte izquierda. Otro detalle importante es como se representan las dependencias entre tareas, si una de ellas empieza con un borde vertical más oscuro y una pequeña flecha apuntándole, quiere decir que esta tarea depende de la tarea de la que surge la flecha, por tanto, debe terminar una para poder empezar la otra, si no es el caso, la tarea se puede realizar cuando se prefiera, incluso de forma paralela a otras.

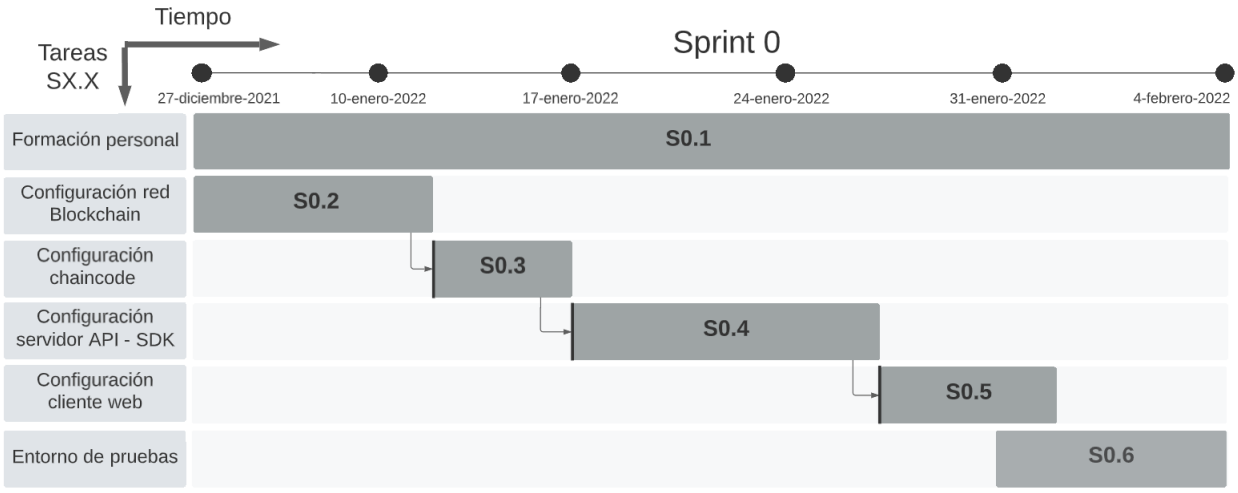


Figura 4: Sprint 0

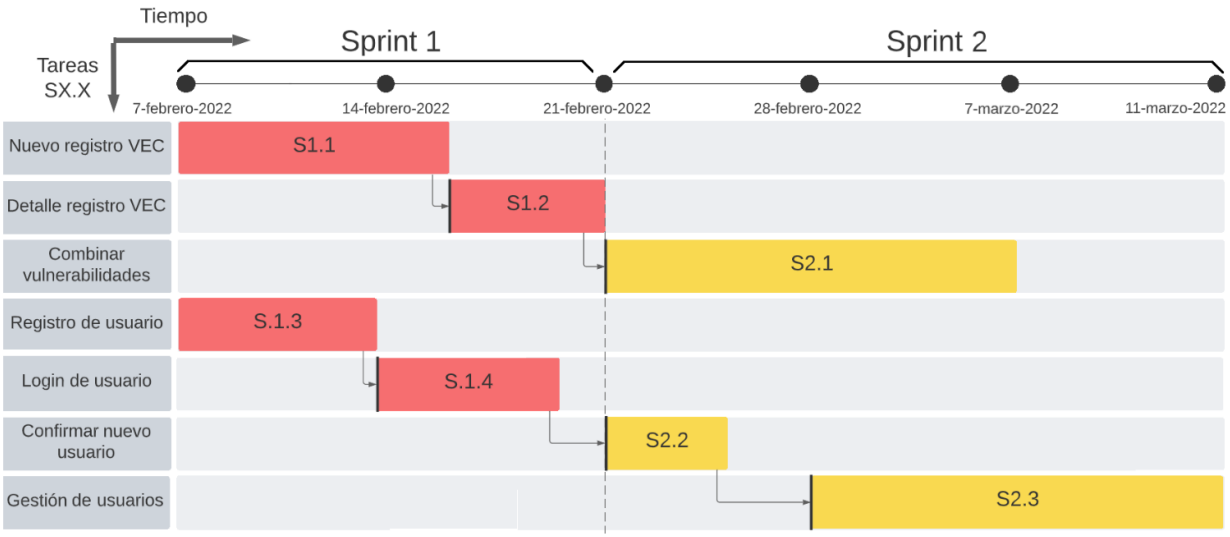


Figura 5: Sprint 1 y 2

Durante los cuatro primeros *sprints*, se desarrollan las características más relevantes de VECsg. De ahí que se traten de *sprints* con gran carga de trabajo y con muchas dependencias entre tareas, sobre todo en los primeros. De hecho, en las primeras iteraciones se puede observar que hay dos caminos, uno de tareas sobre registros VEC y otro para los usuarios. En los *sprints* 3 y 4 se centran todos los esfuerzos en los registros VEC, y es en la quinta iteración donde se termina de completar la funcionalidad requerida para los usuarios.

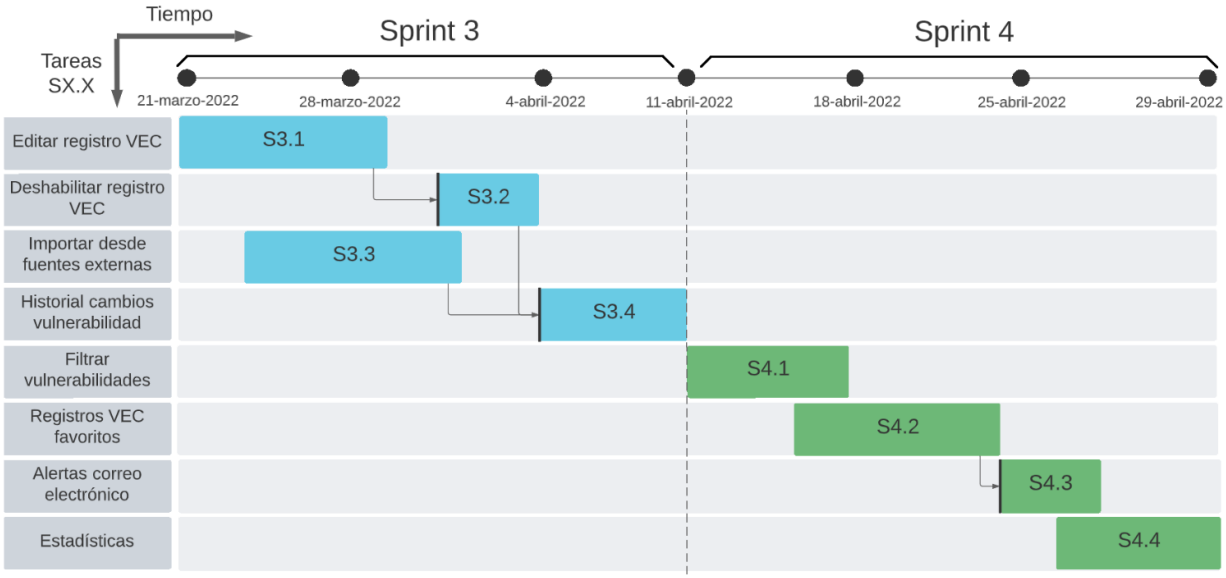


Figura 6: Sprint 3 y 4

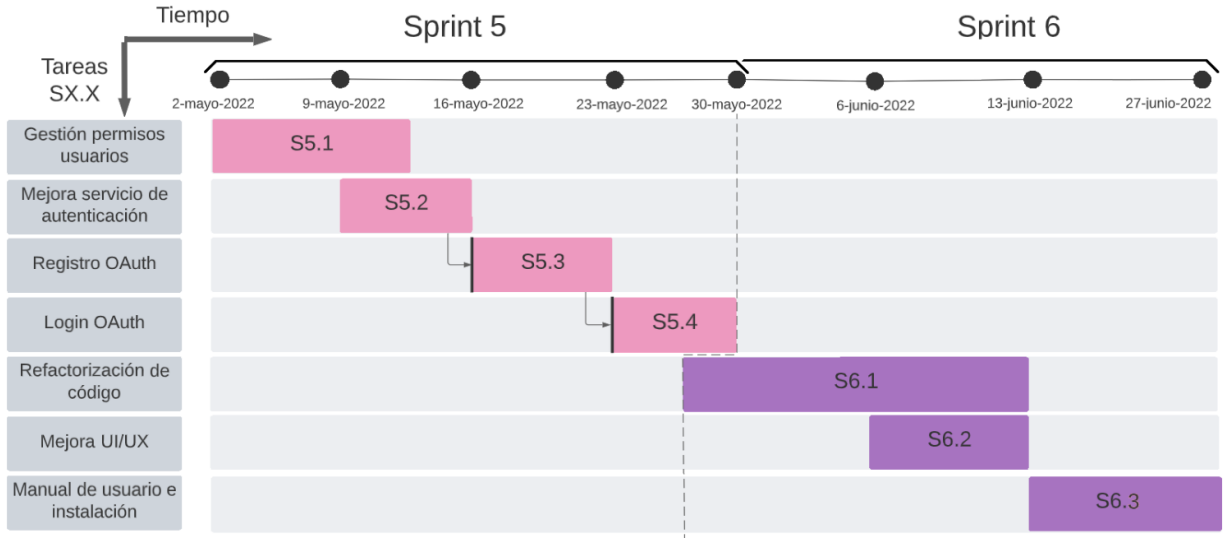


Figura 7: Sprint 5 y 6

Tras la iteración número 6, para completar este Trabajo Fin de Grado solo faltaba documentar todo el proceso de desarrollo seguido y recoger las decisiones tomadas en cuanto al sistema, así como un resumen detallado de todas las herramientas que ofrece VECsg para la gestión de vulnerabilidades de un consorcio de organizaciones.

4

Análisis de requisitos

4.1. Requisitos

En las reuniones iniciales con el cliente, rol ejercido por la tutora del TFG, se definieron una serie de requisitos que se aprobaron y acordaron implementar, tras un exhaustivo análisis donde fueron precisados y priorizados. Estos requisitos se dividen en dos tipos: los primeros, son los requisitos funcionales que describen claramente la funcionalidad que se requiere en el sistema, los otros, son los requisitos no funcionales, que incluyen condiciones, limitaciones y reglas de cómo debe ser elaborada y ofrecida la funcionalidad descrita anteriormente.

4.1.1. Requisitos funcionales

A continuación, se enumeran los requisitos de funcionamiento, (abreviados como RF de Requisitos Funcionales). Estos requisitos son los que debe cumplir el sistema VECsg para alcanzar su objetivo principal de ser un repositorio seguro de vulnerabilidades compartidas entre los miembros del consorcio de un sistema federado como la Smart Grid:

RF.1.- Gestión de registros VEC. El sistema tiene que permitir la gestión de registros VEC.

RF.1.1.- Crear un nuevo registro VEC. El usuario debe poder añadir una nueva vulnerabilidad en el sistema con todos los atributos propios de un registro VEC, tales como descripción, dispositivo afectado, entidad, solución o referencias, entre otros.

RF.1.2.- Modificar un registro VEC existente. El usuario podrá cambiar datos de un registro VEC, que ya existan con anterioridad en la plataforma, por múltiples razones.

RF.1.3.- Combinar registros VEC. El usuario debe poder combinar vulnerabilidades en una sola, cuando se detecte redundancia entre los datos nuevos que se quieren introducir y los registros ya almacenados.

RF.1.4.- Importar registros VEC de fuentes externas de vulnerabilidades. El sistema debe permitir la inclusión de registros de vulnerabilidades, obtenidas de otra fuente (concretamente, de la base de datos de vulnerabilidades, NVD del NIST).

RF.1.5.- Marcar un registro VEC como favorito. El usuario debe poder seguir una vulnerabilidad marcándola como favorita y a partir de entonces, será informado de todo lo que ocurra con esa vulnerabilidad, mediante correo electrónico y en la plataforma web.

RF.1.6.- Consultar últimos cambios de un registro VEC favorito. Un usuario cualquiera, podrá consultar qué datos han cambiado en la última modificación de uno de sus registros favoritos.

RF.1.7.- Filtrar registros VEC por parámetros. El usuario podrá acceder fácilmente a los registros VEC que esté interesado gracias a un completo mecanismo de búsqueda que debe permitir filtrar por fecha, riesgo, solución, etc.

RF.1.8.- Explorar el historial de un registro VEC. Un usuario administrador podrá consultar todos movimientos que han ocurrido sobre un registro VEC en el pasado, accediendo a todo lujo de detalles sobre las acciones realizadas o los datos cambiados, e incluso, el autor de dichos cambios.

RF.1.9.- Desactivar un registro VEC. Un administrador podrá desactivar un registro VEC para que deje de estar disponible para el resto de usuarios y sea enviado a la lista de desactivados.

RF.1.10.- Consultar el estado actual de la plataforma. El usuario debe poder acceder a información ordenada y estructurada sobre el estado actual de los datos globales que maneja la plataforma.

RF.2.- Acceso al sistema. El sistema tiene que gestionar la autenticación y la autorización tanto de nuevos, como de usuarios ya aprobados.

RF.2.1.- Registrarse en el sistema. Un usuario podrá registrarse en el sistema con sus datos y inscribiéndose a una organización, quedando a la espera de ser aprobado su acceso al sistema.

RF.2.2.- Iniciar sesión en la plataforma. El usuario podrá iniciar sesión cargando su perfil de la plataforma que le permita acceso a las vulnerabilidades del sistema, según su rol.

RF.2.3.- Registrarse en la plataforma con OAuth. Un usuario podrá crear un nuevo usuario en el sistema, utilizando el estándar de autorización OAuth. Con la posibilidad de combinar datos del usuario ofrecidos por el servicio de autenticación, con datos introducidos por el propio usuario.

RF.2.4.- Iniciar sesión en la aplicación mediante OAuth. El usuario podrá iniciar sesión directamente con la ayuda del servicio de autenticación de OAuth.

RF.2.5.- Aprobar usuarios que solicitan unirse a la plataforma. Los usuarios existentes podrán aprobar usuarios nuevos que soliciten unirse a la plataforma, mediante comunicaciones que se enviarán a sus respectivos correos electrónicos.

RF.3.- Gestión de cuentas de usuarios. El sistema tiene que permitir administrar los datos almacenados de los usuarios.

RF.3.1.- Crear usuarios Un usuario administrador debe ser capaz de crear nuevos usuarios que cuenten directamente con acceso al sistema.

RF.3.2.- Editar usuarios Un administrador podrá cambiar datos de los usuarios ya creados, como sus correos de notificación o sus roles, entre otros.

RF.3.3.- Activar/Desactivar usuarios . Un administrador podrá desactivar usuarios para quitarles el acceso a la plataforma, en caso de negligencia. Posteriormente el administrador puede volver a activar el usuario para otorgarle de nuevo la posibilidad de entrar en el sistema, una vez aclarado lo ocurrido.

4.1.2. Requisitos no funcionales

Ahora se presentan los requisitos no funcionales de VECsg (RNF), que son igual de importantes que los primeros, porque indican condiciones u obligaciones que debe cumplir el sistema, en la forma en la que este proporciona la funcionalidad. Si no se cumplen estos requisitos no funcionales, puede ser que las necesidades del cliente no sean satisfechas, aunque se hayan implementado todos los requisitos funcionales.

RNF.1 El sistema tiene que permitir el almacenamiento seguro de vulnerabilidades, así como, garantizar la seguridad de las conexiones.

RNF.2 El sistema tiene que ofrecer un buen rendimiento y contar con tiempos de respuesta cortos.

RNF.3 El sistema tiene que estar construido con interfaces intuitivas y fáciles de usar.

RNF.4 El sistema deberá asegurar la integridad y la inmutabilidad de los datos, de manera que, las instancias creadas deberán permanecer siempre en el sistema.

RNF.5 El sistema tiene que ser trazable y permitir señalar responsables de acciones que se consideren anómalas y malintencionadas.

RNF.6 El sistema será cerrado y con un sistema estricto de privilegios y roles (espectador, gestor y administrador).

RNF.7 El sistema debe permitir un control estricto sobre los nuevos usuarios, que deben solicitar el acceso a la plataforma.

4.2. Diagrama de casos de uso

En VECsg, existen tres tipos de roles. El más básico es el usuario **espectador**, el cual tiene acceso a la gran parte de la funcionalidad de VECsg, pero no puede realizar transacciones en la blockchain y, por tanto, no puede realizar operaciones de escritura. El papel fundamental de este usuario es consultar los registros VEC creados por otros. El usuario **gestor**, ya cuenta con unos permisos más elevados, por lo que a parte de las consultas a los registros VEC, va a poder crear nuevos y editar los existentes.

El último rol es el de **administrador**, de gran relevancia dentro de la plataforma, que, además de poder realizar la gestión de registros VEC completa, va a poder llevar a cabo la gestión de usuarios de la plataforma, acceder al historial de las vulnerabilidades, desactivar registros VEC o importarlos desde fuentes externas.

A continuación, se muestra el diagrama de casos de uso (véase la figura 8) donde se puede ver cómo se distribuyen las funcionalidades que ofrece VECsg entre los distintos roles de espectador o gestor, y más adelante en el diagrama de la figura 9, se pueden apreciar todas las características disponibles para los administradores.

Figura 8: Diagrama de casos de uso del usuario espectador y gestor

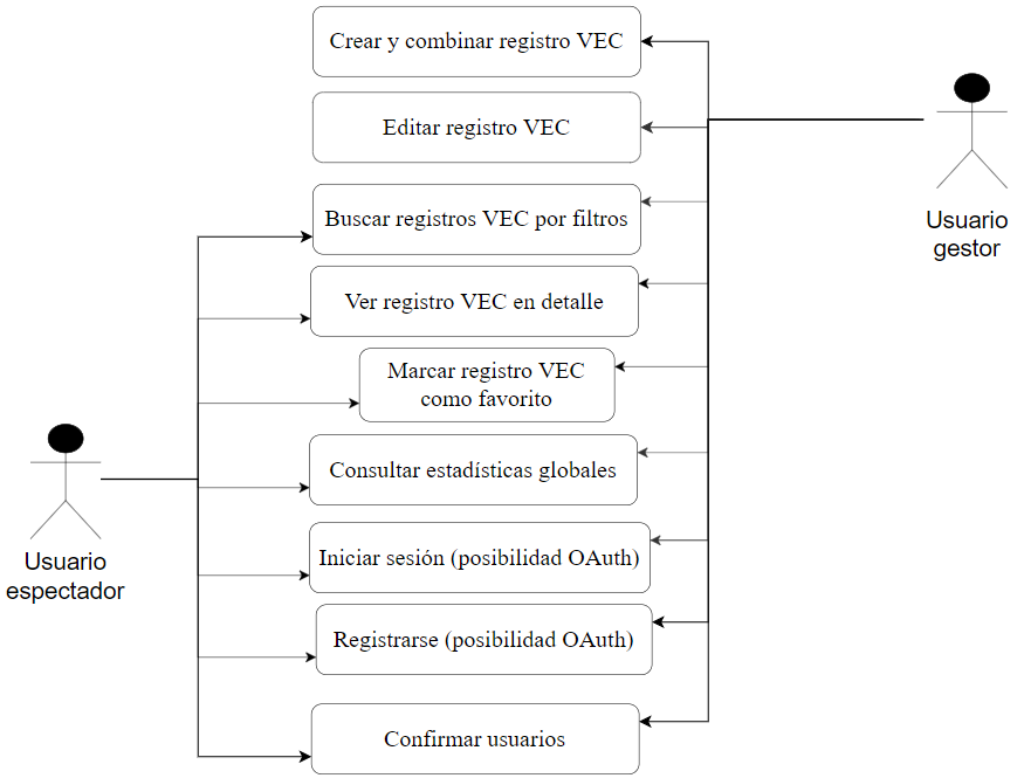
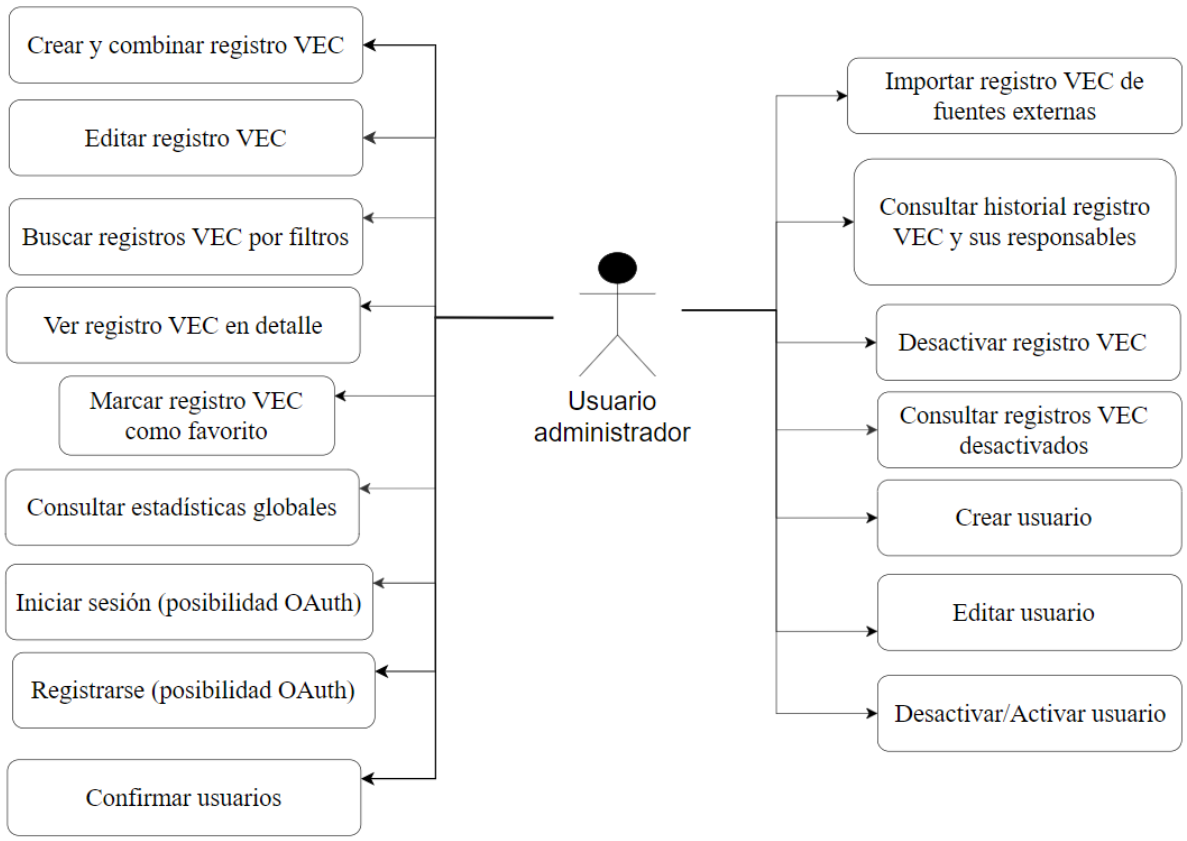


Figura 9: Diagrama de casos de uso de un administrador



En estos diagramas gráficos se pueden consultar y comprender de un solo vistazo cuales son las acciones que va a poder realizar cada tipo de usuario en el sistema, pero si se requiere conocer en profundidad estos casos de usos y observar el desarrollo completo de los diferentes escenarios, en el apéndice A acerca de los **Casos de uso** se encuentra la descripción textual rigurosa de cada uno de ellos.

5

Diseño y Desarrollo

5.1. Fundamentos

El sistema consiste en una plataforma web, denominada VECsg. Al estar construida sobre una red de blockchain, para desplegarla se han de levantar varios nodos a lo largo de las entidades el consorcio de la Smart Grid. En ella, se almacenan y gestionan de forma segura vulnerabilidades que se han denominado "registros VEC", de Vulnerabilidades y Exposiciones Comunes, la traducción literal de las CVE de MITRE.

La razón del nombre reside en que los registros que se almacenan en VECsg, están inspirados en el estándar de nomenclatura y de forma que siguen las vulnerabilidades en MITRE. Se han tomado partes de su formato como los CVE-ID para identificarlos claramente o atributos esenciales como la descripción o las referencias, pero también se ha ampliado para adaptarlo a la forma que toman las vulnerabilidades en VECsg y a al dominio concreto de la Smart Grid (tipos de estaciones eléctricas afectadas, riesgos, posibles soluciones y parches, etc.). Se puede ver con más detalle cómo esta formado un registro VEC en la figura 10.

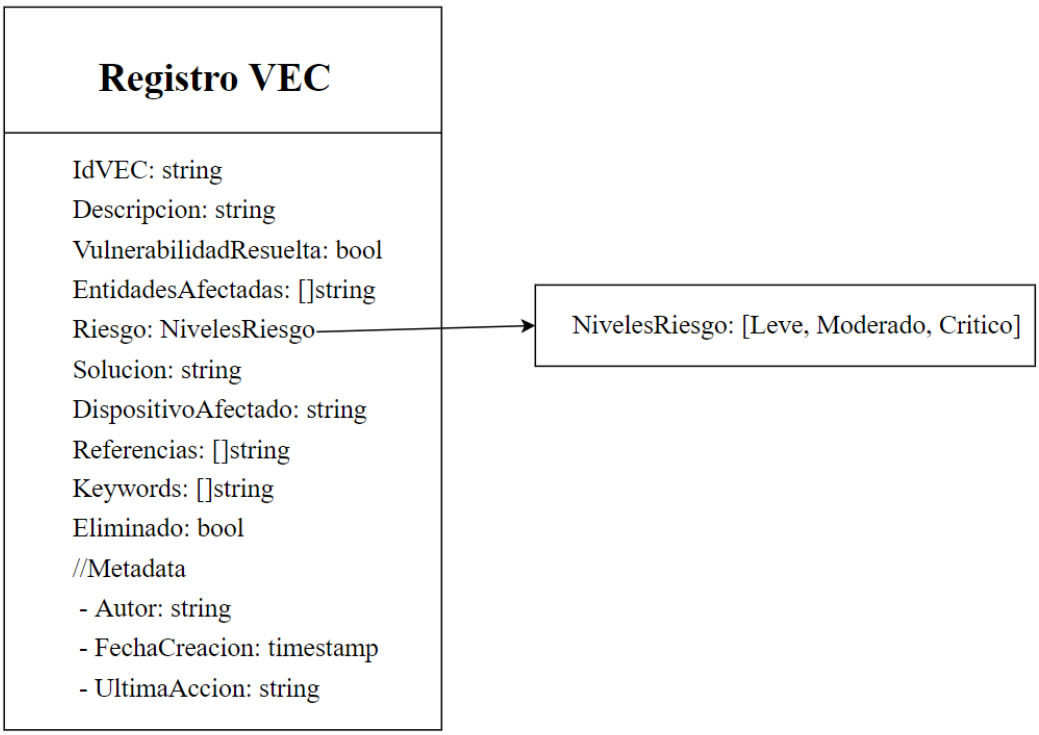


Figura 10: Diagrama de operaciones definidas en el chaincode

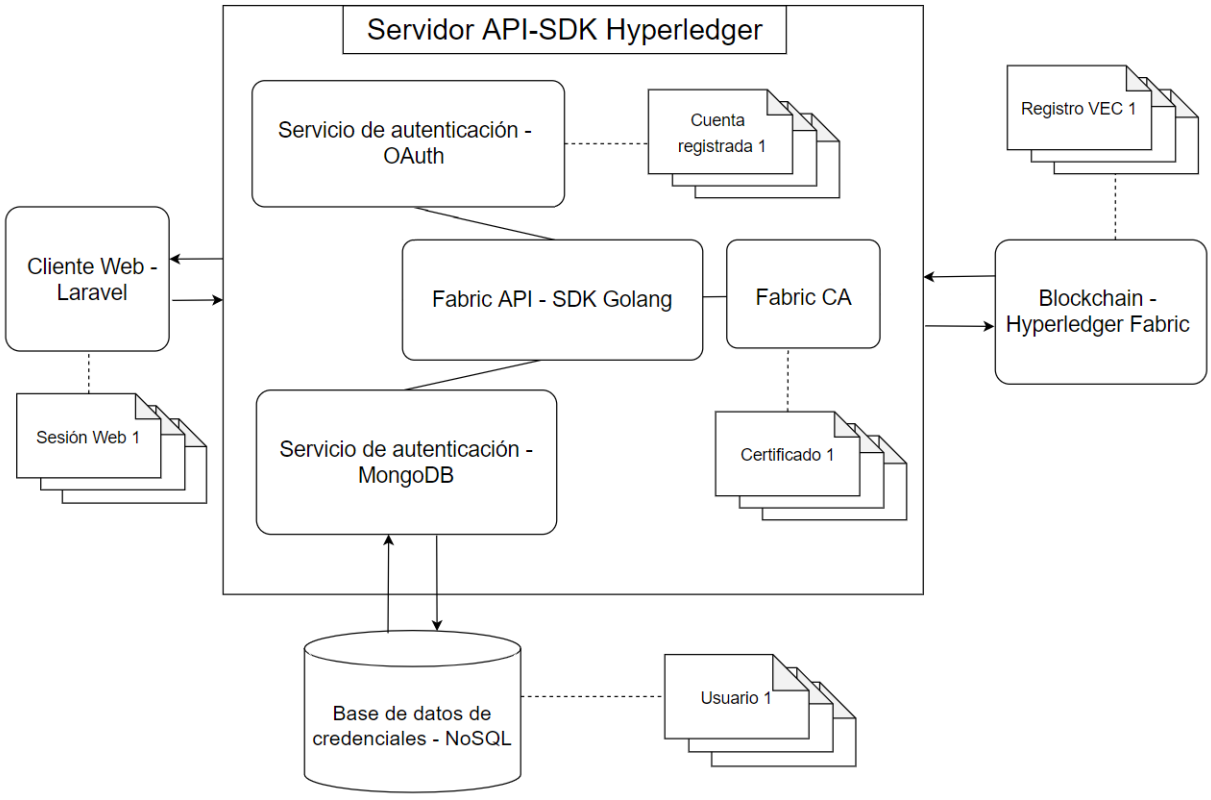
Estos activos críticos se persiten y comparten en la red de blockchain con las organizaciones del consorcio, gracias a una compleja arquitectura.

5.2. Arquitectura del sistema

El sistema está construido sobre una red de blockchain, basada en Hyperledger Fabric [68], entonces para que los usuarios del consorcio puedan acceder a los datos de las vulnerabilidades de la cadena de bloques, existen una serie de aplicaciones y programas intermedios que se comunican entre sí. El usuario va interactuar directamente con el cliente web, que ha sido construido en Laravel, un framework de PHP para construir aplicaciones web [69]. Esta aplicación cliente contiene todas las vistas y define la interacción del usuario con la plataforma, ya que se encarga de llamar a la API RESTful que se comunica con la red de blockchain. Esta API se ha definido usando Gin-Gonic [70], un framework de Go [71] para construir el backend de la aplicación web, implementa un SDK de Hyperledger Fabric basado en Go [72] que es el que permite ejecutar operaciones sobre la red de blockchain.

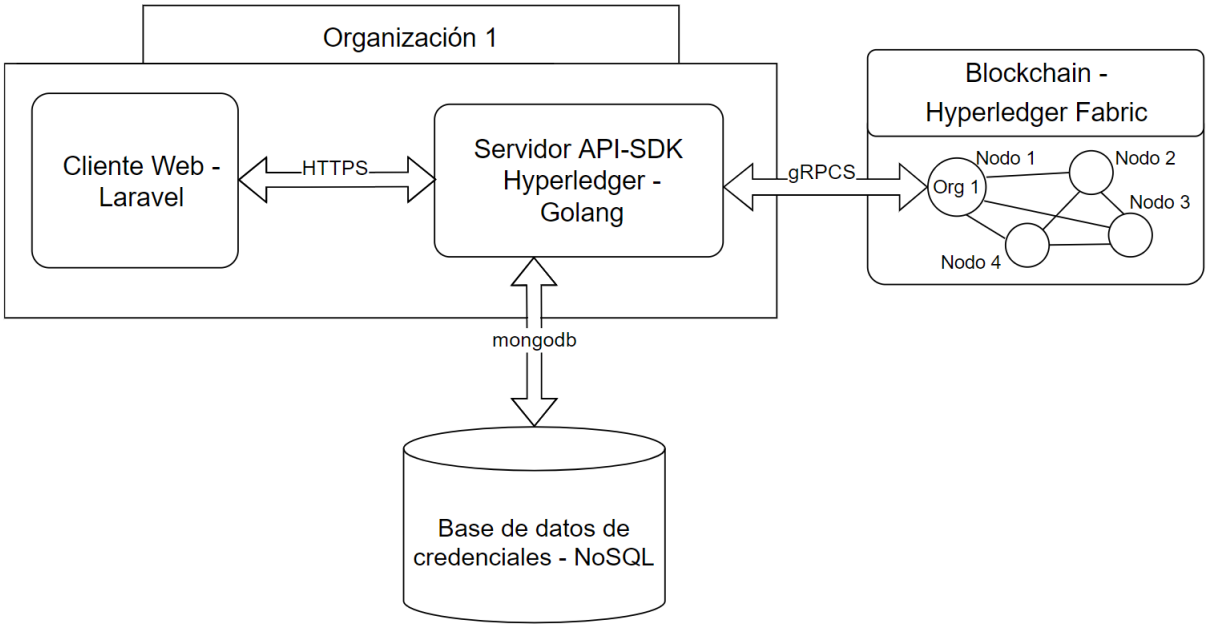
La API que usa el SDK mecionado, permite gestionar la identidad y autorización de un usuario como miembro de una organización. Para la autenticación de los usuarios se cuentan con dos servicios clave y alternativos. De modo que, el usuario puede acceder a la plataforma mediante OAuth, para lo cual se hará uso del servicio correspondiente, o en cambio, si el usuario quiere autentificarse de forma normal, con usuario y contraseña, entonces se debe pasar por el servicio de autenticación acoplado al servidor Gin-Gonic que hace de API de Hyperledger, para acceder a la base de datos de usuarios NoSQL, para poder comprobar que las credenciales insertadas son correctas. Una vez autenticados, la Fabric CA de la organización, que es el nombre que reciben las autoridades certificadores en Hyperledger Fabric, se encarga de autorizar al usuario, gestionando sus permisos y generando un certificado que verifica su entidad dentro de la blockchain. En la figura 11 se puede ver con gran detalle como es la estructura interna de VECsg y cómo se consigue conectar la aplicación web a la blockchain.

Figura 11: Arquitectura en detalle de la red de VECsg



Para entender mejor como se organiza la red que hemos construido, en la figura 12, se puede apreciar un diagrama simplificado que expone como se organizan los diferentes nodos y aplicaciones, y las comunicaciones que existen entre ellos.

Figura 12: Arquitectura general de la plataforma VECsg



Como se puede apreciar, el cliente web y el servidor API, se encuentran en el ámbito de una misma organización y se comunican entre sí por HTTPS, que es el protocolo por excelencia de la Web para enviar datos protegiendo su integridad y confidencialidad. Para poder acceder a la blockchain una organización tiene que desplegar un par cliente/servidor de VECsg y configurar su perfil de conexión particular, acción que podrá llevarse a cabo sin ningún tipo de problema, por todas la organizaciones pertenecientes al consorcio de la Smart Grid. Así, los usuarios de la organización podrán conectarse a la red de bloques en cadena de Hyperledger, pudiendo conectarse a su propio nodo dentro de la blockchain o a otro según el perfil de conexión mencionado anteriormente. De nuevo, para garantizar la seguridad de los datos estos se envían cifrados entre el servidor y la red de Hyperledger Fabric, mediante el protocolo gRPCS [73], un protocolo para la ejecución de procedimientos de forma remota en otras máquinas, desarrollado por Google que se ofrece como alternativa más rápida y ligera que la arquitectura REST, que a diferencia de este no funciona con JSON, sino con protobúferes [74] a través de los cuales se transfieren datos con una interfaz predefinida.

Por último, cabe destacar la conexión con la base de datos MongoDB, a la que el servidor que utiliza el SDK de Hyperledger, se conecta mediante un protocolo propio de MongoDB que añade una capa sobre TCP/IP, conocido como *MongoDB Wire Protocol* [75], que para poder establecer comunicación con la base datos, exige que se proporcione un certificado, garantizando la seguridad de acceso a la información crítica de los usuarios almacenados.

5.3. Fases del desarrollo

Para comprender cuales han sido las fases de desarrollo del VECsg, puede servir de gran ayuda consultar la planificación del desarrollo expuesta en el capítulo 3 sobre **Metología y planificación**. Aunque para el desarrollo se llevaron a cabo los *sprints*, en aspectos generales se puede decir que el desarrollo se realizó en las siguientes fases:

5.3.1. Formación, instalaciones y configuraciones iniciales

Duración 27/12/2021 - 04/02/2022

Una de las primeras acciones que hubo que llevar a cabo, fue consultar mucha información y documentación sobre las tecnologías que se iban a utilizar para fabricar VECsg: Hyperledger Fabric, MongoDB, Docker, etc. Gracias a esta formación inicial se pudieron ir instalando y configurando las diferentes herramientas que se necesitaron para que funcionasen las aplicaciones que es necesario desplegar en las diferentes capas. Una de las mayores dificultadas, en un sistema que exige ser seguro, fue precisamente la configuración del protocolo TLS, concretamente la gestión de los certificados que es necesario compartir con la red para el correcto funcionamiento de este protocolo. Pero al final se pudo completar gracias a que crearon certificados para todos los nodos y se configuraron las rutas de su localización.

5.3.2. Desarrollo paralelo de registros VEC y usuarios

Duración 07/02/2022 - 11/03/2022

Una vez pasado ese primer *sprint* inicial, en las dos iteraciones siguientes empieza el desarrollo de algunas de las funcionalidades que debe ofrecer VECsg. Para ello, se llevó

a cabo en paralelo el desarrollo de los dominio de aplicación de registro VEC y usuario. Esto se hizo así para que, por ejemplo, se pudiesen empezar a crear registros VEC dentro de una organización para la cual se había registrado un nuevo usuario, que también tenía que poder iniciar sesión en nuestro sistema.

En esta fase también se desarrolla la capacidad de combinar registros VEC, tras el análisis exhaustivo de registros VEC similares, que se realiza cada vez que se crea un registro VEC nuevo para evitar la redudancia de datos en la blockchain.

5.3.3. Desarrollo centrado en registros VEC

Duración 21/03/2022 - 29/04/2022

Durante los *sprints* 3 y 4, se centran todos los esfuerzos en desarrollar las funcionalidad restantes de los registros VEC. Por ejemplo, ya no solo se crean nuevos registros VEC, si no que se permiten editarlos y borrarlos. Una vez completadas estas tareas se pudo llevar a cabo la característica para obtener el historial de un registro VEC, mediante el que se recogen todas estas acciones que se realizan sobre él. El historial de cambios y el filtro de registro VEC fueron dos operaciones interesantes de desarrollar porque hubo que explorar las posiblidades que la API de Hyperledger Fabric ofrecía sobre los contratos inteligentes. Para conseguir el historial completo bastó con llamar a una función de la API, ya que como veremos en el siguiente sección la propia estructura interna de la cadena de bloques facilita hacer este tipo de consultas de trazabilidad. Para filtrar los registros VEC se tuvo que investigar cómo hacer consultas complejas entre todos los datos del estado último de la blockchain y en el apéndice C, que trata de la **Implementación de VECSG** se puede comprobar cómo se consiguió llevar a cabo.

Por último, se implementaron los registros VEC favoritos, que permite que el usuario guarde ciertos registros como especiales y sea notificado de todo lo que pase con ellos. Estas notificaciones al usuario, tuvieron que desarrollarse tanto como comunicaciones externas por correo electrónico, como con avisos y llamadas de atención al usuario dentro de VECsg para que fuese consciente de estos cambios.

5.3.4. Desarrollo centrado en usuarios

Duración 02/05/2022 - 30/05/2022

Una vez resueltos todos los requisitos sobre registros VEC, era necesario prestar atención a los usuarios. Este *sprint* número 5, se empezó arreglando las comunicaciones al resto de usuarios de la organización, cuando un usuario se registra. Ya que, como sabemos, VECsg es un sistema cerrado donde los nuevos clientes solo podrá acceder a la plataforma una sean confirmados y aceptados mediante las invitaciones que tiene que aceptar el resto.

También, se puso gran esfuerzo en el sistema de permisos de los usuarios, para controlar de forma estricta a qué activos pueden tener acceso los usuario y qué tipo de acciones pueden llevar a cabo. Y se terminó de completar el sistema de autenticación de usuarios. Por un lado se habilitó el inicio de sesión y el registro de usuarios mediante el protocolo OAuth [76]. Por otro lado, se identificó que la gestión de las credenciales que se hacían en el SDK de Hyperledger Fabric era deficiente, por lo que tuvo que añadirse al servidor de VECsg, un servicio aclopado de autenticación basado en MongoDB para almacenar de forma ligera y de rápido acceso datos para corroborar el inicio de sesión del usuario. Este se encarga solo de la autenticación porque la autorización y los permisos del usuario se delegan en la Fabric CA de Hyperledger Fabric.

5.3.5. Refactorización, optimización y documentación

Duración 30/05/2022 - 27/06/2022

Por último, se llevo a cabo un *sprint* completo de tareas para pulir y dejar bien documentado el proyecto. Así, se llevaron a cabo refactorizaciones generales del código, aplicando diversos principios de código limpio, se mejoró la interfaz gráfica y se producieron varios manuales: de usuaro y de instalación.

5.4. Tecnologías utilizadas

En esta sección, vamos a tratar las numerosas tecnologías utilizadas en la creación de un sistema de estas características. La más importante de ellas es el proyecto Hyperledger Fabric que nos permite tener una red blockchain con las características de exclusividad y rendimiento buscadas para la gestión seguro de los activos críticos: las vulnerabilidades. Por otro lado, también tienen gran peso otras tecnologías, como docker, que facilita el despliegue y otras operaciones de DevOps, también Gitlab, para la gestión de repositorios de código y sus versiones, y por supuesto frameworks de frontend y de backend.

5.4.1. Blockchain (Hyperledger Fabric)

Tal y como comentábamos anteriormente, para desarrollar el repositorio de vulnerabilidades pensando en que sea seguro y no manipulable por diseño, se ha decido utilizar la tecnología blockchain. Es muy importante que no se introduzcan anomalías con datos incorrectos en este tipo de repositorio porque perderían su confianza y validez. A diferencia de otros tipos de tecnología blockchain, más conocidos como los públicas, como ya mencionamos en la introducción, nuestro sistema está construido sobre una red blockchain privada y permisionada, es decir, no todo el mundo puede acceder a ella y crear bloques y transacciones, si no que solo usuarios seleccionados tendrán acceso a esta información. Así se permite a las empresas e instituciones un acceso seguro, con poca latencia y sin necesidad de tener que pagar comisiones por transacción, gracias al tipo de red de blockchain que se puede desplegar usando Hyperledger Fabric [68].

En un sistema que va a estar lleno de fallos y errores que comprometen seriamente a las organizaciones de la Smart Grid es imprescindible la garantía de seguridad. Por eso nuestra red de blockchain va a estar alineada con los principios de la tríada CIA, que ya mencionamos cuando hablamos de los activos con los que se trata en la Smart Grid, pero que ahora se aplican a los activos de nuestro propio sistema (las vulnerabilidades de VECsg).

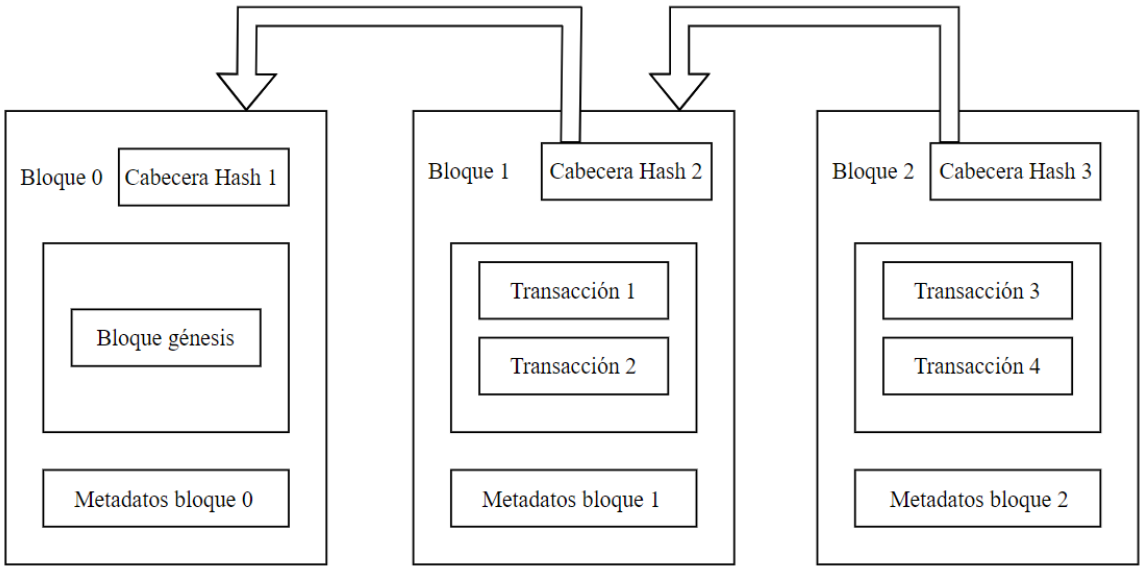
-Acceso: base de datos distribuidas en varios nodos por lo que si se cae, siempre se puede acceder a la información almacenada en otro espacio físico, no está toda la información

crítica centralizada en una sola base de datos. Eso es en cuanto a lectura, pero para poder escribir datos se necesita llegar a un consenso para que las nuevas transacciones que se creen sean verificadas y firmadas e incluirlas en un bloque. En este caso, gracias al protocolo de tolerancia a fallos de Hyperledger Fabric denominado CFT, se pueden crear transacciones válidas solo con la mitad de los nodos disponibles [77].

-Confidencialidad: utiliza los protocolos de seguridad más avanzados, protegiendo todas las conexiones con TLS. La red de Hyperledger Fabric se caracteriza, además, por ser un sistema permisionado, destinado a empresas que requieren una alta confidencialidad en sus transacciones [78]. Así, gracias a un sistema rígido de permisos y roles de usuarios, para los que Hyperledger asegura su identidad con rigurosidad, las vulnerabilidades aportadas por un usuario, solo son vistas por otros usuarios identificados y autorizados, nunca por usuarios impostores [79].

-Integridad: se asegura la integridad de los datos gracias a la tecnología blockchain que almacena la información en bloques, que apuntan a otros bloques creados anteriormente. Esta cabecera incluye un hash creado a partir del contenido del propio bloque (transacciones validadas) y el hash del bloque anterior de modo que estos quedan fuertemente ligados y se hacen inmutables [80].

Figura 13: Estructura básica de una cadena de bloques



Una red de blockchain privada permisionada está formada por una serie de nodos en los que se despliega la red, que al ser privada se le puede dar la confianza a ciertos nodos, para que cuando aparezcan nodos deshonestos, sus acciones puedan ser bloqueadas por el número mayor de nodos legítimos interesados en la integridad y la veracidad de la red de blockchain. Gran parte de esta confianza reside en un nodo especial de Hyperledger Fabric, conocido en inglés como *orderer*. El nodo *orderer* se encarga de ordenar las transacciones en el tiempo y generar los bloques, que envía a los nodos de todas las organizaciones para que hagan una última validación y seguidamente se aplique la transacción a cada copia local que tienen esos nodos, de la blockchain [81].

Esta manera de alcanzar el consenso, entregando la confianza a nodos determinados, quiere decir que en Hyperledger Fabric se está aplicando el protocolo de validación de transacciones conocido como prueba de autoridad o, en inglés, *proof of authority* [82]. Entre otras cosas proporciona tolerancia a fallos y mejora el rendimiento con respecto a otro tipo de redes blockchain en las que, por ejemplo, sí se lleve a cabo el minado de la información como forma de validación.

En nuestro caso la red la forman numerosas organizaciones que se agrupan dentro de un consorcio o sistema federado de la Smart Grid. A través de esta red, es posible controlar qué entidades pueden acceder a los datos persistidos en la cadena de bloques, evitando que usuarios fuera del consorcio accedan a ellos o creen nuevas transacciones provocando cambios no deseados. Para ello, en el caso de la tecnología Hyperledger Fabric, las organizacion se unen a un canal [83], en el que se mantendrá una base de datos distribuida la cual se irá actualizando conforme se realizan actualizaciones sobre el estado actual de los activos de la red de blockchain.

Las operaciones que se pueden realizar sobre los datos del libro de cuentas distribuido (o *ledger* según el proyecto Fabric), se definen en el contrato inteligente, que en la tecnología Hyperledger Fabric denominan *chaincode*, y que también tiene que ser añadido a la blockchain y aprobado por las entidades como una transacción más. A partir de ahí, solo los usuarios de las organizaciones que se hayan recogido en las políticas de aprobación (las denominadas *endorsement policies* en Hyperledger Fabric) podrán realizar transacciones

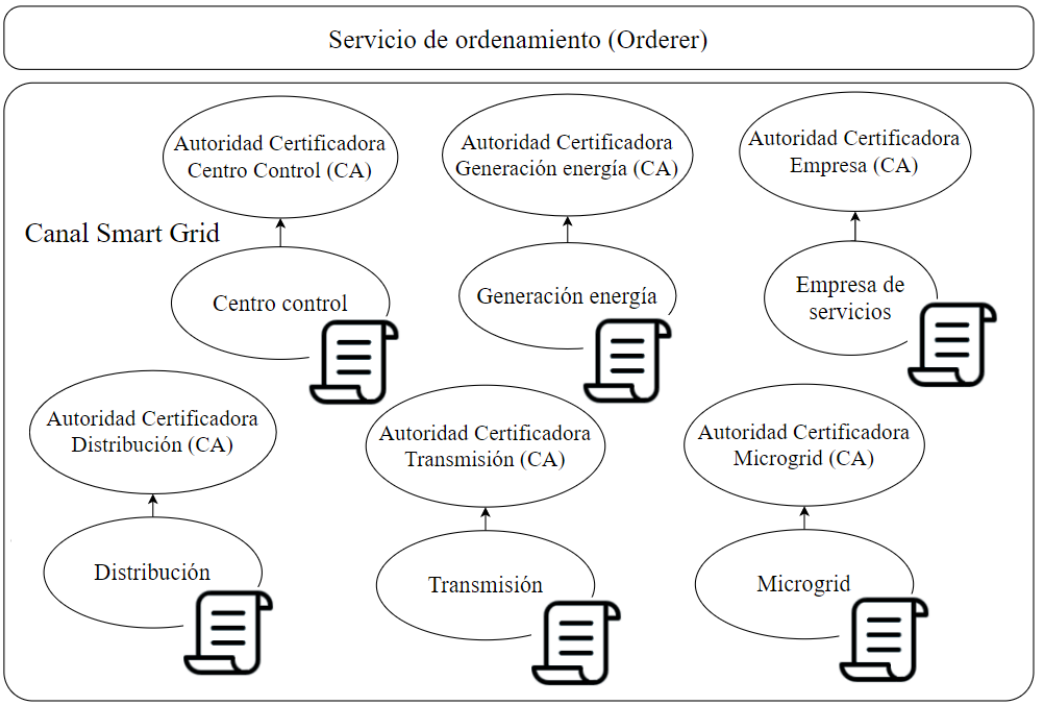
correctas. Más adelante, en la sección 5.5 sobre el chaincode en VECsg, se comentarán los detalles la implementación concreta en este sistema.

La tecnología blockchain, por ser una cadena de bloques ofrece la posibilidad de volver atrás en el tiempo recorriendo dicha cadena hacia atrás. De este modo, gracias a la propia naturaleza del sistema, poder llevar una trazabilidad de las vulnerabilidades, accediendo a su historial es bastante sencillo. Esto permite ver un completo registro de los cambios que han ido produciéndose a lo largo del tiempo. En nuestro sistema se habilitará solo a usuarios autorizados, un historial de movimientos para cada registro VEC, en el que se podrá apreciar con detalle cuales son las acciones que se han realizado y quién ha sido el autor de dichos cambios.

Y finalmente, otra de las razones por las que se ha elegido esta tecnología de red blockchain permisionada, es que, mediante autoridades certificadoras (CA) para cada organización, que son entidades de confianza que generan certificados que aseguran la identidad de algún ente, se van a gestionar la identidad de cada usuario que participa en la blockchain.

Los individuos que colaboran en el repositorio aportando conocimiento sobre vulnerabilidades están claramente localizados e identificados por los administradores del sistema, mediante certificados digitales, sin los cuales, no pueden hacerse efectivos sus cambios, porque estos son imprescindibles para autenticarse ante los nodos de la red de la blockchain. Esto permite que VECsg ofrezca un servicio de *accountability* o rendición de cuentas en caso de que haya que identificar al responsable de algún movimiento sospechoso en las vulnerabilidades y exposiciones registradas, por lo que también está garantizado el no repudio de las operaciones realizadas por un usuario en el sistema [84].

Figura 14: Estructura organizacional de la red de blockchain



5.4.2. Otras tecnologías

Docker

Esta tecnología de virtualización de contenedores permite con la instalación de diferentes artefactos de Hyperledger Fabric, como los nodos de las organizaciones u otros nodos auxiliares como la Fabric CA o el contenedor donde de ejecuta el chaincode, que ofrecen funcionalidades complementarias a los nodos de las organizaciones. Gracias a herramientas como Docker Compose [85] los contenedores de los nodos de las distintas organizaciones se pueden levantar a la vez y añadir todos ellos a una misma red para que se comuniquen entre sí.

CouchDB

CouchDB se define a sí misma como una base de datos NoSQL ligera, que permite almacenar datos binarios con formato JSON [86]. Está presente en cada nodo de la blockchain para mantener una copia del estado actual de los activos de la cadena de bloques. Entre otros beneficios permite ejecutar consultas complejas con un alto rendimiento debido a que se pueden optimizar creando índices de los datos por campos [87].

MongoDB

MongoDB [88] es un sistema de base de datos NoSQl, no dirigido a crear bases de datos relacionales, sino que en este caso, los datos se guardan como archivos binarios en un formato llamado BSON (similar al JSON pero con codificación binaria). En MongoDB, las bases de datos son documentales, esto quiere decir que en ellas se almacenan documentos. Un documento es un objeto de pares clave-valor en formato JSON, que se agrupa con otros de su misma forma en una colección y al final estas colecciones se agrupan y forman una misma base datos. Esta nueva estructura comparada con las bases de datos relaciones nos permiten ser más flexibles con los datos y tener un mejor rendimiento, por ejemplo, en las consultas se pueden obtener todos los datos que se quieran directamente si se tienen en un mismo documento, no hay que unir datos que se consultan en diferentes tablas como en el modelo relacional [89]. En este caso, MongoDB nos permite tener un almacenamiento persistente de usuarios y de rápido acceso, para el servicio de autenticación que se adhiere al servidor del SDK y que necesita para tener un sitio seguro contra el que comprobar las credenciales y contraseñas que introduce el usuario al iniciar sesión.

Gin-Gonic

Gin-Gonic [70] es un framework de desarrollo web escrito en Go (Golang). Es muy similar a otro de los grandes framework web en Go, conocido como Martini, pero el primero ofrece un rendimiento mucho mayor. Gracias a Gin-Gonic se ha podido crear fácilmente una API RESTful que acepte peticiones HTTPS y procese los datos encriptados que se reciben en ellas. Este servidor web, usando el SDK de Hyperledger, es el encargado de conectarse de forma segura mediante el protocolo gRPCS, mencionado antes, con los nodos y el resto de artefactos de la red blockchain de Hyperledger, para gestionar las consultas y las transacciones que el usuario quiera realizar.

Laravel

Laravel [69] es el framework de PHP basado en plantillas que nos permite crear fácilmente un cliente web que permite hacer llamadas a la API RESTful y mostrar los resultados por pantalla, en el cual se puede utilizar JavaScript [90] para hacer más diná-

mica la plataforma web.

Cypress.io

Cypress.io [91] es una herramienta de pruebas de extremo a extremo, las cuales consisten en revisar de manera transversal todo el sistema con las acciones que hace el usuario final, por lo que se obtiene una batería de pruebas muy focalizada en las funcionalidades y los requisitos del sistema. Además Cypress.io cuenta con una suite gráfica de testing muy cómoda donde se puede ir viendo el resultado de las pruebas e ir haciendo viajes hacia atrás en el tiempo en los tests realizados, para ver que acciones se han hecho y cómo ha reaccionado la web ante posibles fallos. Cypress ofrece una API de funciones de testing muy sencilla e intuitiva para simular todos los comportamientos de un usuario en una página web.

Mailtrap

Mailtrap es un servicio de pruebas de envío de correos electrónicos, que ofrece un servidor que utiliza el SMTP, el cual es un extendido protocolo que permite el envío de correos entre servidores a través de internet. A este servidor puedan enviarse las diferentes comunicaciones que hace nuestra plataforma web para comprobar que todo está correctamente configurado y tanto la visualización como los datos de los correos son correctos. En su capa gratuita este servicio de correo ofrece un de límite de correos recibidos lo suficientemente holgado como para probar las notificaciones de un sistema como VECsg.

Gitlab

Gitlab ha sido usado como repositorio tanto como para el proyecto del cliente web como para los archivos que formaban la API REST que se comunicaba con la red de blockchain.

Gmail y Google Meets

Utilizado principalmente para intercambiar dudas, preguntas y documentos importantes con la tutora (cliente) y realizar las reuniones remotas que se han necesitado.

5.5. Definición del smart contract

Para cumplir con los requisitos del sistema, se han definido las siguientes funciones en el contrato inteligente o *chaincode*, los cuales constituyen las posibles operaciones a realizar sobre la red de blockchain y que solo pueden ser ejecutadas por los miembros del canal.

Hay que tener en cuenta que estos métodos solo van a poder ser invocados por usuarios que dispongan de los roles exigidos en cada caso y siempre son ejecutados en el contexto de una organización, salvo las funciones de consulta de datos generales y registros VEC, las cuales permiten que se intercambie conocimiento entre organizaciones.

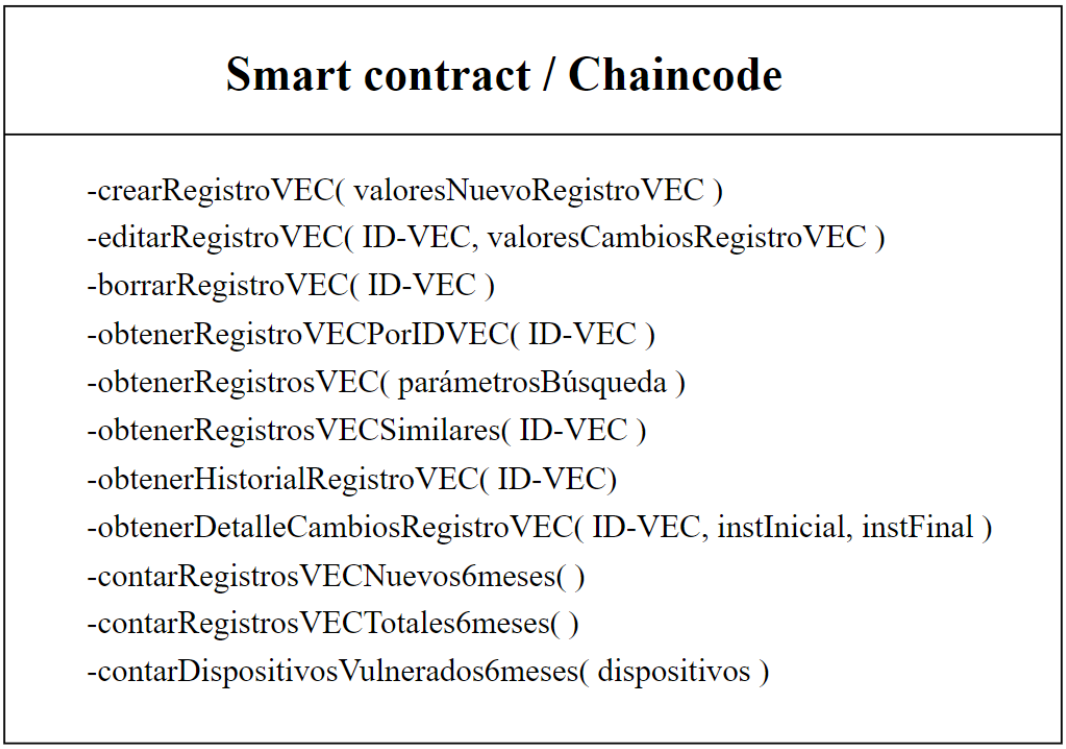


Figura 15: Diagrama de operaciones definidas en el chaincode

crearRegistroVEC(valoresNuevoRegistroVEC) — gracias a este método se pueden crear nuevos registros VEC. Los *valoresRegistroVECCrear* son los que constituyen el nuevo registro VEC, tales como la descripción, la fecha de creación, el autor, la solución, el dispositivo o las entidades de la Smart Grid afectadas, entre otros valores que se exigen para poder construir lo que hemos llamado un "registro VEC" según el estándar de

nomenclatura y definición por el que se rige VECsg. El único valor que no se proporciona directamente es el ID-VEC, el identificador del registro VEC, que es generado automáticamente para cada registro VEC nuevo y tiene la siguiente forma: "VEC-YYYY-NNNN", donde "YYYY" representa el año en el que se creó el registro VEC, y "NNNN" son cifras arbitrarias que se otorgan a un registro VEC para distinguirlo del resto del mismo año. El número de cifras en "NNNN" puede crecer siempre que se necesite ampliar el rango de los ID-VEC (5, 6, 7 cifras, etc.). Para poder añadir a la blockchain nuevas transacciones y ejecutar este tipo de métodos, es necesario que el usuario tenga el rol de gestor o de administrador, ya que esta operación conlleva proponer una transacción y escribir en la cadena de bloques.

editarRegistroVEC(ID-VEC, valoresCambiosRegistroVEC) — esta función se utiliza para editar un registro VEC con los valores, con los que se llame a esta función del smart contract, teniendo en cuenta que hay que buscar un registro VEC al que aplicarle los cambios. Este registro será buscado por su ID-VEC y si no se encuentra en el repositorio distribuido de la red blockchain, el *chaincode* devolverá un mensaje de error. Este método también es útil para cuando se quieren fusionar o combinar registros VEC, ya que permite tomar los datos comparados del registro nuevo y del que ya existía y volcarlos en el segundo para que en éste, se mantenga la combinación actualizada. Al igual que antes, un usuario no podrá modificar un registro VEC si es solo espectador, se exige que su rol sea de gestor o administrador.

borrarRegistroVEC(ID-VEC) — utilizando este método se puede obtener un registro VEC y borrarlo. Teniendo en cuenta que es un borrado lógico, por lo que la vulnerabilidad y su historial siempre permanecerán en la cadena de bloques. Solo un usuario administrador puede desactivar registros VEC.

obtenerRegistroVECPorIDVEC(ID-VEC) — esta operación permite obtener un registro VEC, mediante su ID-VEC, para así poder acceder a la información que hay almacenada sobre él. Cualquier tipo de usuario puede realizar esta consulta, incluso un espectador.

obtenerRegistrosVEC(parámetrosBúsqueda) — este método permite obtener un listado completo de registros VEC, que se pueden filtrar por distintos parámetros (fecha, existencia de solución, descripción, etc.). También, permite ordenarlos por fecha descendente, e incluso, ofrece soporte para paginación de registros VEC.

obtenerRegistrosVECSimilares(ID-VEC) — gracias a él se pueden analizar que registros VEC existen similares a uno dado, cuyo identificador se pasa como parámetro. Estos registros VEC semejantes son útiles, por ejemplo, cuando se está creando un nuevo registro VEC y se quiere evitar la redundancia en la blockchain. En ese caso, se busca si existe algún registro similar para poder combinarlos en uno solo, que cuente con la información de ambos.

obtenerHistorialRegistroVEC(ID-VEC) — sirve para obtener información sobre todos los movimientos que han ocurrido sobre un registro VEC en concreto. Así se puede conocer, qué acciones se realizaron, quiénes realizaron está acciones e instantes temporales. Solo un administrador podrá consultar todos los responsables y cambios en el pasado de un registro VEC.

obtenerDetalleCambiosRegistroVEC(ID-VEC, instInicial, instFinal) — este método se utiliza para obtener detalles de cuáles fueron los datos que cambiaron en un registro VEC entre dos instantes de tiempo y sirve de apoyo a la función de obtener el historial porque amplía con creces la información de carácter general que proporciona este último. Estos detalles expandidos, solo pueden ser accedidos por usuarios con alto nivel de permisos (administrador).

contarRegistrosVECNuevos6meses(), contarRegistrosVECTotales6meses(), contarDispositivosVulnerados6meses(dispositivos) — todas estas funciones son usadas principalmente para obtener recuentos de los registros VEC y poder mostrar estos datos en gráficas de estadísticas. En general, los métodos calculan el total de entidades en los últimos seis meses. Estas entidades pueden ser, registros VEC nuevos, acumulados o registros que afectan a los dispositivos aportados como parámetro.

En el apéndice C sobre la **Implementación de VECsg**, se puede encontrar una explicación en detalle de la implementación de algunos de estos métodos del *chaincode*, así como de los métodos de otros elementos que se sitúan entre el cliente y la blockchain, como el servidor creado usando el SDK de Hyperledger Fabric, por los que tiene que pasar la información antes de llegar al usuario para, por ejemplo, poder verificar correctamente su identidad o mostrarle los activos obtenidos de la blockchain en una página web con un diseño agradable e intuitivo.

6

Pruebas

En este capítulo, se explican las pruebas realizadas sobre VECsg, el cual al tratarse de un sistema que maneja información crítica debe ofrecer altos niveles de estabilidad y ausencia de fallos, ello se verifica mediante unas exhaustivas pruebas que debe superar favorablemente.

6.1. Características de las pruebas del sistema

Para probar el sistema se ha tomado como guía la relación de requisitos (véase sección **4.1** sobre **Requisitos**), y se realizaron pruebas centradas en cada uno de ellos específicamente. Unas pruebas que han consistido en comprobar el funcionamiento normal en su flujo correcto y también, en introducir estados de error, en los que el sistema debe responder correctamente ante los fallos introducidos a conciencia.

El tipo de pruebas realizadas sobre VECsg son *end-to-end*, unas pruebas que consisten en la automatización de las acciones del usuario y observar el comportamiento de la plataforma en estos flujos completos, para finalmente, comprobar el estado en el que queda la aplicación y verificar si coincide con el esperado o no.

Concretamente, la herramienta utilizada ha sido Cypress.io [91]. Un software de testing *end-to-end* de creciente popularidad, que permite recrear las acciones del usuario en un sitio web mediante comandos simples, centrado siempre en la automatización. Entre las muchas facilidades que incluye este software de pruebas que supera en muchos aspectos al antiguo *Selenium* [92], se encuentran, un sistema de esperas a la carga de los elementos y una interfaz gráfica desde donde lanzar las pruebas y observar cómo se robotizan las interacciones y los flujos dentro de la web mientras se van mostrando los resultados fallidos o no, que se van consiguiendo progresivamente.

Entre las funcionalidades probadas está, la creación de registros VEC, la combina-

ción, las búsquedas complejas o la gestión de usuarios, con el objetivo de maximizar la cobertura de requisitos probados, de manera que, se pueda verificar que el sistema funciona correctamente según lo exigido por el cliente y dentro del alcance que él mismo ha definido.

6.2. Batería de pruebas

A continuación, se muestra un resumen de las pruebas realizadas sobre el sistema, juntos con los resultados obtenidos. En caso de que la prueba no haya tenido un desempeño favorable, se indicará con detalle la justificación de ese resultado. Cada prueba, además, está relacionada con el conjunto de requisitos que cubre y del cual verifica su correcto funcionamiento.

Tabla 4: Pruebas sobre los registros VEC

Pruebas realizadas	**Requisitos**	**Resultado**
Creación de un nuevo registro VEC con diversos atributos arbitrarios	RF.1	CORRECTO
Validación de campos incorrectos o incompletos del formulario de creación de registros VEC	RF.1.1	CORRECTO
Modificación de distintos valores de un registro VEC	RF.1.2	CORRECTO
Combinación de registros VEC que son similares	RF.1.3	CORRECTO
Historial del registro VEC con resumen de los acciones sobre un registro VEC	RF.1.5	CORRECTO
Historial del registro VEC con resumen de los cambios de un registro VEC	RF.1.5	CORRECTO
Búsqueda específica de registros VEC con diversos parámetros	RF.1.6	CORRECTO
Desactivar registros VEC permanentemente	RF.1.7	CORRECTO
Marcar un registro VEC para guardarlo como favorito	RF.1.8	CORRECTO
Los gráficos y estadísticas se actualizan correctamente con nuevos registros	RF.1.9	CORRECTO

Tabla 5: Pruebas sobre gestión de usuarios de la plataforma

Pruebas realizadas	**Requisitos**	**Resultado**
Creación de un nuevo usuario ya confirmado por el administrador	RF.2.1	CORRECTO
Creación de nuevos miembros con nombres de usuario ya existentes en el sistema	RF.2.1	CORRECTO
Modificación de los datos del perfil de un usuario	RF.2.2	CORRECTO
Desactivar usuario quitándole el acceso a la plataforma	RF.2.3	CORRECTO
Volver a activar usuario comprobando el envío del aviso de su reingreso en la plataforma	RF.2.3	CORRECTO

Tabla 6: Pruebas sobre el registro y el inicio de sesión de usuarios

Pruebas realizadas	**Requisitos**	**Resultado**
Inicio de sesión de un usuario confirmado	RF.3.1	CORRECTO
Inicio de sesión con credenciales incorrectas	RF.3.1	CORRECTO
Inicio de sesión de un usuario que aún no ha sido aprobado	RF.3.1	CORRECTO
Registro de un usuario quedando pendiente de aprobación	RF.3.2	CORRECTO
Registro de un usuario con datos erróneos (usuario ya existente)	RF.3.2	CORRECTO
Registro en la plataforma a través de OAuth	RF.3.3	CORRECTO
Inicio de sesión en el sistema mediante OAuth	RF.3.4	CORRECTO
Aceptar en la plataforma a un usuario que está esperando ser confirmado	RF.3.5	CORRECTO

7

Conclusiones y Líneas Futuras

7.1. Conclusiones

En este TFG se proporciona una plataforma de tratamiento y almacenamiento seguro y efectivo de información crítica como son las vulnerabilidades que afectan a una institución de un sistema federado. La plataforma, que ha recibido el nombre de VEGsg, busca cumplir con las necesidades de las organizaciones en la nueva red eléctrica inteligente, ofreciendo un entorno de gestión de vulnerabilidades controlado y cerrado con usuarios identificados y reconocidos, permanente disponibilidad de vulnerabilidades que se aseguran que son legítimas y no han sufrido modificaciones maliciosas, sistemas de alerta ante vulnerabilidades críticas y nuevos agujeros de seguridad en los dispositivos, etc. Habilitando un repositorio de vulnerabilidades guardadas en un formato reconocido por todos, similar al de las vulnerabilidades y exposiciones comunes de CVE, para que puedan aportar mucho más valor y se pueda recurrir a fuentes de conocimiento externas para completar con información actualizada, el inventario de vulnerabilidades de VECsg.

Un sistema como el de VECsg, con las opciones que ofrece de seguridad, inmutabilidad o control de acceso de usuarios, por estar construido sobre una red de blockchain, permite a las organizaciones tener una manera fácil y segura de gestionar sus problemas relacionados con vulnerabilidades, amenazas y ataques. Gracias a herramientas como VECsg, que permiten la prevención y la actuación temprana, se evitan gran parte de las acciones malintencionadas de los ciberdelicuentes al no poder aprovecharse de las vulnerabilidades, porque las entidades de la Smart Grid han tenido la información y el conocimiento necesario para tomar acciones de mantenimiento, reparaciones, parches de seguridad, etc.,

sobre los dispositivos y los sistema informáticos en general, resolviéndose eficazmente estos agujeros de seguridad, que podrían haber comprometido todo el sistema eléctrico.

Además, que VECsg permita a una organización compartir datos de registros VEC con otras, habilita la posibilidad de crear un depósito colectivo de vulnerabilidades. Y es que, para una organización también es muy importante lo que ocurre en las otras y puede aportar mucho valor a los problemas y brechas de sus propios sistemas. Debido a que no hay ninguna organización inmune a ser expuesta a ataques, sobre todo, si se ignoran las vulnerabilidades existentes y no se llevan a cabo buenas prácticas y políticas de seguridad. Por ejemplo, se pueden dar casos de ataques, como el que ocurrió con el programa malicioso WannaCry [93], un *ransomware* de tipo gusano, que infectó a cientos de miles de ordenadores en poco tiempo, precisamente, explotando vulnerabilidades ya resueltas, pero muchos dispositivos aún no se habían actualizado y cayeron infectados, perjudicando a toda la red de ordenadores conectados. Por tanto, se puede comprobar como la sinergia y la colaboración entre organizaciones es importante para hacer frente a grandes amenazas. Una base de conocimiento compartida permite esta comunicación de agujeros de seguridad entre organizaciones y sirve como un recurso fundamental para fomentar aspectos relacionados con la ciberinteligencia de amenazas en el futuro [94], usada a afrontar los nuevos y más avanzados ataques que puedan tener como objetivo a las entidades del consorcio, con el fin de proteger el complejo sistema eléctrico informatizado que propone la Smart Grid.

Por último, la seguridad también va a estar garantizada dentro del propio consorcio, por si aparece algún intento de realizar modificaciones anómalas de las vulnerabilidades por parte de los usuarios aceptados y respaldados por el consorcio, porque, gracias al amplio registro de movimientos y trazabilidad que ofrece VECsg, los administradores del consorcio no van a tener ningún problema en identificarlos y aplicar la medidas oportunas a los usuarios responsables, evitándose sabotajes internos que corrompan las vulnerabilidades y hagan que se pierda la confianza y la legitimidad de la información almacenada en VECsg.

La digitalización de la red eléctrica es un proceso que tiene mucho aún camino por recorrer para que la Smart Grid pueda llegar a ofrecer sus ventajas en todo su esplendor, por lo que la evolución en este ámbito es constante y cada vez surgen más proyectos

dedicados a esta nueva concepción de la red eléctrica [95]. Por eso, ahora es el momento perfecto para plantear y desarrollar proyectos como este, que se preocupen por preservar la seguridad en la Smart Grid desde su etapa temprana y experimental, permitiendo que esta avance teniendo a su disposición todo un abanico de herramientas de ciberseguridad. Siguiendo este camino, en el futuro se podrá contar con un sistema eléctrico robusto, gracias a plataformas como VECsg, que con su base de conocimiento sobre vulnerabilidades, otorgan estabilidad y confianza a la nueva Smart Grid.

7.2. Líneas Futuras

Como se ha explorado con profundidad el ámbito de la Smart Grid y los problemas de ciberseguridad que le afectan, los repositorios de vulnerabilidades CVE y la posibilidad de compartir este conocimiento entre organizaciones, son numerosas las líneas futuras que han surgido y se han querido considerar para seguir explotando el potencial de la plataforma VECsg, pero que se han dejado fuera para que este proyecto no se excediese en su alcance.

Una de las funciones más interesantes es la inclusión de más fuentes de información para importar registros VEC de distintos sitios externos y no solo de la NVD del NIST. Esto permitiría contar con información más amplia y actualizada de forma automática, pudiendo llegar a incluir en VECsg soluciones a las vulnerabilidades, de estas entidades externas. Para ello, existen sistemas de intercambio de amenazas, como MISP [96]. Se trata de una plataforma de código abierto para el almacenamiento y la difusión de inteligencia de amenazas y vulnerabilidades, donde participan agentes tan importantes como la OTAN y a las que se puede acceder fácilmente usando una API, en la que hay que pedir acceso a alguna de las organizaciones que gestionan la red MISP, como el CIRCL (*Computer Incident Response Center Luxembourg*) [97]. También, existen otras tecnologías como STIX y TAXII.

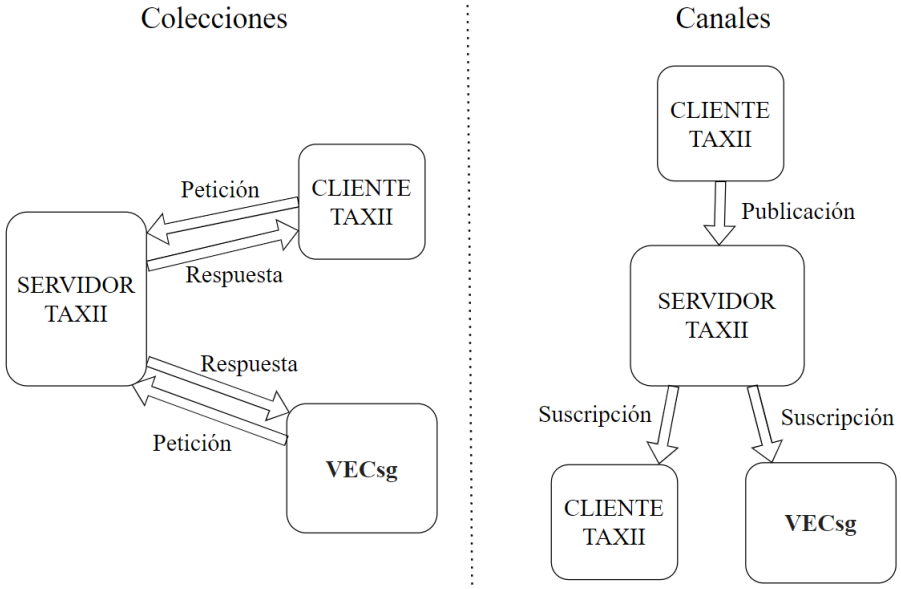
STIX [98] es un lenguaje que define un formato específico para datos de amenazas, orientado a que se pueda llevar a cabo un análisis de inteligencia de amenazas con estos datos. En él toman parte numerosas entidades diferentes que forman relaciones significativas entre sí. Una de estas entidades son las vulnerabilidades, las cuales dentro de la estructura de STIX se relacionan con infraestructuras, *malware*, actores malintenciona-

dos, olas de ataques, etc. Pero lo más interesante, aquí, es que en el lenguaje STIX las vulnerabilidades se expresan como registros CVE, es decir, con una nomenclatura que entiende y en la que está basada la plataforma web, VECsg.

La plataforma en la que se gestionan las entidades con amenazas escritas en STIX, es TAXII [99]. Esta ofrece varios servicios que permiten a las organizaciones interesada en el intercambio de amenazas, construir una red con diferentes estructuras. Por ejemplo, los nodos con información relevante pueden comunicarse entre sí uno a uno, formando una red de pares, depender todos de un repositorio común o suscribirse a un servicio de amenazas que les envíe información cuando la pidan los clientes o sea enviada por el propio servidor de TAXII por iniciativa propia [100].

La plataforma VECsg podría integrarse dentro de un sistema TAXII como cliente de información de inteligencia de amenazas, filtrando por vulnerabilidad y si existe, por su mitigación correspondiente. El cliente podría, por ejemplo, suscribirse a un servidor TAXII en el que otros clientes publiquen información de inteligencia de vulnerabilidades o simplemente interactuar con el repositorio de amenazas descritas en STIX, mediante peticiones y respuestas en una arquitectura de cliente-servidor típica (véase la figura 16).

Figura 16: Ejemplos de propuestas de comunicación de VECsg con la plataforma TAXII



Entre las posibilidades de expansión de este proyecto, también se han considerado exponer más datos de las empresas, pero manteniendo su privacidad. Esto se puede llevar a cabo utilizando funciones más avanzadas de Hyperledger Fabric como las colecciones de datos privadas o en inglés *private data collections* [101] para manejar información personal de los miembros de las organizaciones (nombres, números de teléfono, ...) de forma segura, reservada y que no se exponga abiertamente a todo el mundo. Por ejemplo, se podrían compartir cierta información personal que tenga relevancia junto con un registro VEC, pero restringiendo quién tiene acceso a esos datos protegidos de la persona responsable.

Otra de las funcionalidades a explorar en este sistema es, un sistema de gestión de riesgo dinámico de las vulnerabilidades que aporte más precisión, basándonos, por ejemplo, en el framework de criticidad de vulnerabilidades del NIST denominado CVSS (cuyas siglas significan en inglés *Common Vulnerability Scoring System*) [102] que permite dar a cada vulnerabilidad una puntuación a partir de diferentes parámetros como la dificultad de explotar la vulnerabilidad, el problema que supone para el acceso, la confidencialidad y la integridad, si se puede replicar en la red o solo en local, etc. De manera que se pueda realizar una mejor clasificación y organización de los registros VEC en la plataforma por su riesgo, más avanzado que el sistema de riesgos actual que se basta con una clasificación de las vulnerabilidades más general en leves, moderadas o críticas.

Por último, pero también muy interesante, se propone un sistema de alertas para que se avise al usuario en momentos en los que se requiera su atención. Por ejemplo, cuando se ha creado un nuevo registro VEC crítico o ha habido algún cambio en algunos de los registros VEC que seguía. Para implementar todo este sistema de alertas se plantea desarrollar, como segunda línea futura, una aplicación móvil creada en Android/iOS que permita a los usuarios iniciar sesión con el mismo perfil que en la plataforma web de VECsg y estar enterados de la actividad sobre las vulnerabilidades que ellos elijan, mediante notificaciones directamente a su dispositivo móvil.

Bibliografía

[1] J. Scott, "Smart Grids - The European Technology Platform for Electricity Networks of the Future," 01-2009. [En línea]. Disponible en: https://www.researchgate.net/publication/251286919_Smart_Grids_-_The_European_Technology_Platform_for_Electricity_Networks_of_the_Future

[2] IoT Solutions Team, "IoT And Energy: The Smart Network And Optimised Usage," *IoT Solutions World Congress*. [En línea]. Disponible en: https://www.iotsworldcongress.com/iot-and-energy-the-smart-network-and-optimised-usage/, [Accedido: 30-05-2022].

[3] M. Taha, "Advantages and Recent Advances of Smart Energy Grid," *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 5, pp. 1739–1746, 2020. [En línea]. Disponible en: https://www.beei.org/index.php/EEI/article/view/2358

[4] A. Sendin, J. Matanza, y R. Ferrus, "The Smart Grid," en *Smart Grid Telecommunications: Fundamentals and Technologies in the 5G Era*, 2021, pp. 1–39. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/9536326/authors

[5] R. Estévez, "La Smart Grid y sus Tecnologías," *Ecointeligencia*, 06-03-2014. [En línea]. Disponible en: https://www.ecointeligencia.com/2014/03/smart-grid-tecnologias/, [Accedido: 31-05-2022].

[6] I. Kaur, "Chapter 29 - Metering Architecture of Smart Grid," en *Design, Analysis, and Applications of Renewable Energy Systems*, ser. Advances in Nonlinear Dynamics and Chaos (ANDC), A. T. Azar y N. A. Kamal, Eds. Academic Press, 2021, pp. 687–704. [En línea]. Disponible en: https://www.sciencedirect.com/science/article/pii/B9780128245552000307

[7] European Commission, "Smart Grids and Meters," Smart grids and smart meters enable better management of energy networks and more efficient consumption., 2022. [En línea]. Disponible en: https://energy.ec.europa.eu/topics/markets-and-consumers/smart-grids-and-meters_en

[8] D. Baimel, S. Tapuchi, y N. Baimel, "Smart Grid Communication Technologies," *Journal of Power and Energy Engineering*, 2016, 04, 1-8. doi: 10.4236/jpee.2016.48001.

[9] P. Siano, "Demand Response and Smart Grids—A Survey," *Renewable and Sustainable Energy Reviews*, vol. 30, pp. 461–478, 2014. [En línea]. Disponible en: https://www.sciencedirect.com/science/article/pii/S1364032113007211

[10] Cummins, "Benefits of Distributed Energy Resources," *Cummins Newsroom*, 02-11-2021. [En línea]. Disponible en: https://www.cummins.com/news/2021/11/02/benefits-distributed-energy-resources

[11] P. Asmus, "Microgrids, Virtual Power Plants and Our Distributed Energy Future," *The Electricity Journal*, vol. 23, no. 10, pp. 72–82, 2010. [En línea]. Disponible en: https://www.sciencedirect.com/science/article/pii/S1040619010002873

[12] J. Zhou, "How Distributed Energy Resources (DER) Augment Smart Grid, Energy Savings," *FacilitiesNet*, 19-10-2020. [En línea]. Disponible en: https://www.facilitiesnet.com/energyefficiency/article/How-Distributed-Energy-Resources-DER-Augment-Smart-Grid-Energy-Savings--19065

[13] SAP, "Distributed Energy Resources (DER) and the Rise of the Prosumer," 2022. [En línea]. Disponible en: https://insights.sap.com/distributed-energy-resources-der-and-the-rise-of-the-prosumer/

[14] European Commission, "Supporting Innovative Solutions for Smart Grids and Storage," 2020, pp. 6-7. [En línea]. Disponible en: https://ec.europa.eu/inea/sites/default/files/h2020_sgs_brochure_2017_web.pdf

[15] A. Pérez, "Las Ventas de Coches Eléctricos Nuevos y de Ocasión Continúan al Alza en España," *Híbridos y Eléctricos*, 24-04-2022. [En línea]. Disponible en: https://www.hibridosyelectricos.com/articulo/sector/ventas-coches-electricos-nuevos-ocasion-continuan-alza-espana/20220422180327057068.html

[16] M. Herráez, "2035, Fecha que la Comisión Europea Pone para Acabar con los Motores de Combustión Diésel y Gasolina," *Auto Bild*, 19-05-2022. [En línea]. Disponible en: https://www.autobild.es/noticias/2035-fecha-comision-europea-pone-acabar-motores-combustion-diesel-gasolina-1063275

[17] P. D. Falco, G. Carpinelli, y F. Mottola, "Preface to -Distributed Energy Storage Devices in Smart Grids-," en *Distributed Energy Storage Devices in Smart Grids*. Energies, 04-2020, pp. IX–X. [En línea]. Disponible en: https://www.mdpi.com/books/pdfview/book/2191

[18] A. Aktaş, "Chapter 10 - The Importance of Energy Storage in Solar and Wind Energy, Hybrid Renewable Energy Systems," en *Advances in Clean Energy Technologies*, A. K. Azad, Ed. Academic Press, 2021, pp. 377–403. [En línea]. Disponible en: https://www.sciencedirect.com/science/article/pii/B9780128212219000104

[19] D. Butcher, "The DSO Model: The Ojectives of a Separated System Operator," *Capgemini*, 12-12-2018. [En línea]. Disponible en: https://www.capgemini.com/gb-en/2018/12/the-dso-model-the-objectives-of-a-separated-system-operator/, [Accedido: 01-06-2022].

[20] F. Zavoda, "Advanced Distribution Automation (ADA) Applications and Power Quality in Smart Grids," en *2010 China International Conference on Electricity Distribution, CICED 2010*, 10-2010, pp. 1–7. [En línea]. Disponible en: https://www.researchgate.net/publication/224225734_Advanced_distribution_automation_ADA_applications_and_power_quality_in_Smart_Grids

[21] A. Thakur, "SCADA Systems," *Engineers Garage*, 07-2019. [En línea]. Disponible en: https://www.engineersgarage.com/scada-systems/, [Accedido: 01-06-2022].

[22] P. Sachdeva, "The Role of Advanced Distribution Automation in Smart Grid," *International Journal of Engineering Research Technology (IJERT)*, vol. 9, ed. 2, 02-2020. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/5590075

[23] S. Azad, F. Sabrina, y S. Wasimi, "Transformation of Smart Grid using Machine Learning," en *2019 29th Australasian Universities Power Engineering Conference (AUPEC)*, 2019, pp. 1–6. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/9084590

[24] YourTechDiet, "7 Distribution Automation Devices Explained," [En línea]. Disponible en: https://yourtechdiet.com/blogs/distribution-automation-devices, [Accedido: 01-06-2022].

[25] A. Gopstein, C. Nguyen, C. O'Fallon, N. Hastings, y D. Wollman, "NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 4.0," *Special Publication (NIST SP)*, NIST, 18-02-2021. [En línea]. Disponible en: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=931882

[26] "NIST: National Institute of Standards and Technology," [En línea]. Disponible en: https://www.nist.gov/, [Accedido: 10-06-2022].

[27] Michael Emmanuel and Ramesh Rayudu and Ian Welch, "Distributed Energy Resources Operations in the Modern Grid," *July 2017 eNewsletter*, IEEE Smart Grid, 07-2017. [En línea]. Disponible en: https://smartgrid.ieee.org/bulletins/july-2017/distributed-energy-resources-operations-in-the-modern-grid

[28] M. Birk, J. P. Chaves-Ávila, T. Gómez, y R. Tabors, "TSO/DSO Coordination in a Context of Distributed Energy Resource Penetration," *Working Paper Series*, MIT CEEPR, 10-2017. [En línea]. Disponible en: http://hdl.handle.net/11531/14296

[29] R. K. Pandey y M. Misra, "Cyber Security Threats — Smart Grid Infrastructure," en *2016 National Power Systems Conference (NPSC)*, 2016, pp. 1–6. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/7858950

[30] S. Zeadally, A. K. Pathan, C. Alcaraz, y M. Badra, "Towards Privacy Protection in Smart Grid," *Wireless Pers Commun*, 2013, 73, 23–50. https://doi.org/10.1007/s11277-012-0939-1.

[31] Trend Micro, "One Flaw too Many: Vulnerabilities in SCADA Systems," *Trend Micro*, 16-12-2019. [En línea]. Disponible en: https:

//www.trendmicro.com/vinfo/us/security/news/vulnerabilities-and-exploits/one-flaw-too-many-vulnerabilities-in-scada-system, [Accedido: 05-06-2022].

[32] G. Lyon, "Nmap: Discover Your Network," [En línea]. Disponible en: https://nmap.org/, [Accedido: 05-06-2022].

[33] Tenable, "Nessus: Know Your Vulnerabilities and Disrupt Attack Paths," [En línea]. Disponible en: https://www.tenable.com/products/nessus, [Accedido: 10-06-2022].

[34] Greenbone, "OpenVAS – Open Vulnerability Assessment Scanner," [En línea]. Disponible en: https://www.openvas.org/, [Accedido: 05-06-2022].

[35] Kaspersky, "Threat Landscape for Industrial Automation Systems. Statistics for H2 2021," 03-03-2022. [En línea]. Disponible en: https://ics-cert.kaspersky.com/publications/reports/2022/03/03/threat-landscape-for-industrial-automation-systems-statistics-for-h2-2021/

[36] P. Haji Mirzaee, M. Shojafar, H. Cruickshank, y R. Tafazolli, "Smart Grid Security and Privacy: From Conventional to Machine Learning Issues (Threats and Countermeasures)," *IEEE Access*, vol. 10, pp. 1–34, 04 2022. [En línea]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9772626

[37] J. Bogna, "What Are Zero-Day Exploits and Attacks?" *PCMag*, 24-03-2022, [En línea]. Disponible en: https://www.pcmag.com/how-to/what-are-zero-day-exploits-and-attacks, [Accedido: 05-06-2022].

[38] D. Breg, "Preparing for Energy Ifndustry Cyberattacks," *WSJ*, 21-04-2022. [En línea]. Disponible en: https://www.wsj.com/articles/preparing-for-energy-industry-cyberattacks-11650575213

[39] Kaspersky, "Cyberthreats for ICS in Energy in Europe. Q1 2020," 31-08-2020. [En línea]. Disponible en: https://ics-cert.kaspersky.com/publications/reports/2020/08/31/cyberthreats-for-ics-in-energy-in-europe-q1-2020/

[40] BBC, "El Virus que Tomó Control de Mil Máquinas y les Ordenó Autodestruirse," 11-10-2015. [En línea]. Disponible en: https://www.bbc.com/mundo/noticias/2015/10/151007_iwonder_finde_tecnologia_virus_stuxnet

[41] N. Corrado, C. Pizarro, C. Gorena, P. Aguado, L. Cuadrado, J. Arias, y P. Salas, "Ciberseguridad en el Sector Eléctrico," *Amenazas para sistemas TI y OT*, Deloitte, 12-2020. [En línea]. Disponible en: https://www2.deloitte.com/content/dam/Deloitte/pe/Documents/risk/cl-ciberseguridad-en-el-sector-electrico-diciembre-2020.pdf

[42] R. Smith, D. Palin, P. Ioulianou, V. Vassilakis, y S. Shahandashti, "Battery draining attacks against edge computing nodes in IoT networks," *Cyber-Physical Systems*, 2020, ISSN 2333-5785 https://doi.org/10.1080/23335777.2020.1716268.

[43] A. Guzman, D. Miessler, V. Rudresh, y C. Smith, "OWASP Internet of Things Top 10," póster informativo, 2018. [En línea]. Disponible en: https://owasp.org/www-pdf-archive/OWASP-IoT-Top-10-2018-final.pdf

[44] OSI, "Botnet: Mirai," [En línea]. Disponible en: https://www.osi.es/es/servicio-antibotnet/info/mirai, [Accedido: 06-06-2022].

[45] J. Styczynski y N. Beach-Westmoreland, "When the Lights Went Out," 2019. [En línea]. Disponible en: https://www.boozallen.com/content/dam/boozallen/documents/2016/09/ukraine-report-when-the-lights-went-out.pdf

[46] The United States Department of Justice, "Six Russian GRU Officers Charged in Connection with Worldwide Deployment of Destructive Malware and Other Disruptive Actions in Cyberspace," 19-10-2020. [En línea]. Disponible en: https://www.justice.gov/opa/pr/six-russian-gru-officers-charged-connection-worldwide-deployment-destructive-malware-and

[47] MITRE ATT&CK, "Standard Application Layer Protocol Techniques," 21-05-2020. [En línea]. Disponible en: https://attack.mitre.org/techniques/T0869/, [Accedido: 06-06-2022].

[48] Trend Micro, "Command and Control [CC] Server," 21-02-2020. [En línea]. Disponible en: https://www.trendmicro.com/vinfo/us/security/definition/command-and-control-server, [Accedido: 06-06-2022].

[49] A. Greenberg, "How an Entire Nation Became Russia's Test Lab for Cyberwar," *Wired*, 20-06-2017. [En línea]. Disponible en: https://www.wired.com/story/russian-hackers-attack-ukraine/

[50] McAfee, "How Pseudo-ransomware KillDisk Creates a Smoke Screen for Cybercriminals," 19-01-2018. [En línea]. Disponible en: https://www.mcafee.com/blogs/enterprise/pseudo-ransomware-killdisk-creates-smoke-screen-cybercriminals/

[51] MITRE, "CVE-2014-4114," 12-06-2014. [En línea]. Disponible en: https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2014-4114, [Accedido: 07-06-2022].

[52] MITRE, "About CVE Records," [En línea]. Disponible en: https://cve.mitre.org/cve/identifiers/, [Accedido: 07-06-2022].

[53] Moxa, "UC-7408 Series," [En línea]. Disponible en: https://www.moxa.com/en/products/phased-out-products/uc-7408-series, [Accedido: 07-06-2022].

[54] National Institute of Standards and Tecnology, "National Vulnerability Database (NVD) - NIST," [En línea]. Disponible en: https://nvd.nist.gov/, [Accedido: 10-06-2022].

[55] NVD, "CVE-2014-6271," 24-09-2014. [En línea]. Disponible en: https://nvd.nist.gov/vuln/detail/cve-2014-6271, [Accedido: 07-06-2022].

[56] NVD, "CVE-2014-7186," 28-09-2014. [En línea]. Disponible en: https://nvd.nist.gov/vuln/detail/cve-2014-7186, [Accedido: 07-06-2022].

[57] NVD, "CVE-2014-7187," 28-09-2014. [En línea]. Disponible en: https://nvd.nist.gov/vuln/detail/cve-2014-7187, [Accedido: 07-06-2022].

[58] M. Gilesarchive, "Triton is the world's most murderous malware, and it's spreading," *MIT Technology Review*, 05-03-2019. [En línea]. Disponible en: https://www.technologyreview.com/2019/03/05/103328/cybersecurity-critical-infrastructure-triton-malware/

[59] E. Echave, "Triton/Trisis/Hatman; un Malware para SIS," *Enredando con redes*, 14-02-2021. [En línea]. Disponible en: https://enredandoconredes.com/2021/02/14/triton-trisis-hatman-un-malware-para-sis, [Accedido: 06-06-2022].

[60] MITRE, "CVE-2018-7522," 26-02-2018. [En línea]. Disponible en: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-7522, [Accedido: 08-06-2022].

[61] FBI, "Triton Malware Remains Threat to Global Critical Infrastructure Industrial Control Systems (ICS)," Industry Notification, 24-03-2022. [En línea]. Disponible en: https://www.ic3.gov/Media/News/2022/220325.pdf

[62] ENISA, "Smart Grid Security," Annex II. Security aspects of the smart grid, 25-04-2012, pp. 19-21. [En línea]. Disponible en: https://www.enisa.europa.eu/topics/critical-information-infrastructures-and-services/smart-grids/smart-grids-and-smart-metering/ENISA_Annex%20II%20-%20Security%20Aspects%20of%20Smart%20Grid.pdf

[63] "CISA: Cybersecurity and Infrastructure Security Agency," [En línea]. Disponible en: https://www.cisa.gov/, [Accedido: 10-06-2022].

[64] CISA, "CISA adds one Known Exploited Vulnerability (CVE-2022-26134) to Catalog ," 02-06-2022. [En línea]. Disponible en: https://www.cisa.gov/uscert/ncas/current-activity/2022/06/02/cisa-adds-one-known-exploited-vulnerability-cve-2022-26134-catalog

[65] Atlassian, "Confluence. El Espacio de Trabajo Remoto para tu Equipo," [En línea]. Disponible en: https://www.atlassian.com/software/confluence, [Accedido: 09-06-2022].

[66] MITRE, "Common Vulnerabilities and Exposures - CVE," [En línea]. Disponible en: https://www.cve.org/, [Accedido: 10-06-2022].

[67] INCIBE, "Alerta Temprana/Vulnerabilidades," [En línea]. Disponible en: https://www.incibe-cert.es/alerta-temprana/vulnerabilidades, [Accedido: 11-06-2022].

[68] Hyperledger, "A Blockchain Platform for the Enterprise," *Hyperledger Fabric Docs*, 21-02-2020. [En línea]. Disponible en: https://hyperledger-fabric.readthedocs.io/en/release-2.2/, [Accedido: 12-06-2022].

[69] T. Otwell, "The PHP Framework for Web Artisans," 06-2011. [En línea]. Disponible en: https://laravel.com/, [Accedido: 17-06-2022].

[70] Gin Team, "Gin Web Framework," [En línea]. Disponible en: https://gin-gonic.com/, [Accedido: 14-06-2022].

[71] The Golang Team, "The Go Programming Language," 03-2012. [En línea]. Disponible en: https://go.dev/, [Accedido: 17-06-2022].

[72] Hyperledger, "Hyperledger Fabric Client SDK for Go," 03-2012. [En línea]. Disponible en: https://github.com/hyperledger/fabric-sdk-go, [Accedido: 17-06-2022].

[73] gRPC Authors, "gRPC: A High Performance, Open Source Universal RPC Framework," [En línea]. Disponible en: https://grpc.io/, [Accedido: 17-06-2022].

[74] The Google Team, "Protocol Buffers," *Google Developers Newsletter*. [En línea]. Disponible en: https://developers.google.com/protocol-buffers, [Accedido: 17-06-2022].

[75] MongoDB, Inc., "MongoDB Wire Protocol," *MongoDB Manual*. [En línea]. Disponible en: https://www.mongodb.com/docs/manual/reference/mongodb-wire-protocol/, [Accedido: 17-06-2022].

[76] IETF OAuth Working Group, "OAuth 2.0," *OAuth Community Site*. [En línea]. Disponible en: https://oauth.net/2/, [Accedido: 12-06-2022].

[77] Hyperledger, "Raft," *Hyperledger Fabric Docs*, 06-11-2020. [En línea]. Disponible en: https://hyperledger-fabric.readthedocs.io/en/release-2.2/orderer/ordering_service.html#raft, [Accedido: 12-06-2022].

[78] Hyperledger, "Privacidad y Confidencialidad," *Hyperledger Fabric Docs*, 17-09-2020. [En línea]. Disponible en: https://hyperledger-fabric.readthedocs.io/es/latest/whatis.html#privacidad-y-confidencialidad, [Accedido: 12-06-2022].

[79] Hyperledger, "Support Role-Based Access Control (RBAC)," *Hyperledger Fabric CA Docs*, 23-05-2021. [En línea]. Disponible en: https://hyperledger-fabric-ca.readthedocs.io/en/latest/users-guide.html#attribute-based-access-control, [Accedido: 13-06-2022].

[80] Hyperledger, "Ledger," *Hyperledger Fabric Docs*, 06-05-2022. [En línea]. Disponible en: https://hyperledger-fabric.readthedocs.io/en/release-2.2/ledger/ledger.html#blockchain, [Accedido: 12-06-2022].

[81] ——, "The Ordering Service," *Hyperledger Fabric Docs*, 06-11-2020. [En línea]. Disponible en: https://hyperledger-fabric.readthedocs.io/en/release-2.2/orderer/ordering_service.html, [Accedido: 12-06-2022].

[82] GeeksforGeeks Team, "Proof of Authority consensus," *GeeksforGeeks*, 11-05-2022. [En línea]. Disponible en: https://www.geeksforgeeks.org/proof-of-authority-consensus/, [Accedido: 12-06-2022].

[83] Hyperledger, "Channels," *Hyperledger Fabric Docs*, 06-05-2022. [En línea]. Disponible en: https://hyperledger-fabric.readthedocs.io/en/release-2.2/channels.html, [Accedido: 13-06-2022].

[84] Hyperledger, "Membership Service Providers (MSP)," *Hyperledger Fabric Docs*, 25-06-2020. [En línea]. Disponible en: https://hyperledger-fabric.readthedocs.io/en/release-2.2/, [Accedido: 12-06-2022].

[85] Docker Inc., "Docker Compose," *Docker Docs*. [En línea]. Disponible en: https://docs.docker.com/compose/, [Accedido: 13-06-2022].

[86] Apache Software Foundation, "CouchDB Relax," [En línea]. Disponible en: https://docs.couchdb.org/en/3.2.2-docs/intro/index.html, [Accedido: 13-06-2022].

[87] Hyperledger, "CouchDB as the State Database," *Hyperledger Fabric Docs*, 2020. [En línea]. Disponible en: https://hyperledger-fabric.readthedocs.io/es/latest/couchdb_as_state_database.html, [Accedido: 13-06-2022].

[88] MongoDB, Inc., "What Is MongoDB?" [En línea]. Disponible en: https://www.mongodb.com/who-uses-mongodb, [Accedido: 13-06-2022].

[89] MongoDB Inc., "Advantages of MongoDB," [En línea]. Disponible en: https://www.mongodb.com/advantages-of-mongodb, [Accedido: 13-06-2022].

[90] TC39, "ECMA-262: ECMAScript 2022 Language Specification," Ecma International, *13º Edición*, Junio 2022. [En línea]. Disponible en: https://www.ecma-international.org/wp-content/uploads/ECMA-262_13th_edition_june_2022.pdf

[91] Cypress.io, "Writing Your First E2E Test," 2018. [En línea]. Disponible en: https://www.cypress.io/, [Accedido: 14-06-2022].

[92] Selenium, "Selenium Automates Browsers. That's It!" 2004. [En línea]. Disponible en: https://www.selenium.dev/, [Accedido: 14-06-2022].

[93] Malwarebytes, "WannaCry," [En línea]. Disponible en: https://www.malwarebytes.com/wannacry, [Accedido: 15-06-2022].

[94] VulDB, "Cyber Threat Intelligence," 2021. [En línea]. Disponible en: https://vuldb.com/es/?kb.cti, [Accedido: 15-06-2022].

[95] T. McPhie y A. C. Parrondo, "Questions and Answers on the Fifth List of Energy Projects of Common Interest (PCIs)," *European Commission*, 19-11-2021. [En línea]. Disponible en: https://ec.europa.eu/commission/presscorner/detail/en/qanda_21_6093, [Accedido: 15-06-2022].

[96] C. Wagner, A. Dulaunoy, G. Wagener, y A. Iklody, "MISP: The Design and Implementation of a Collaborative Threat Intelligence Sharing Platform," en *Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security*. ACM, 2016, pp. 49–56. [En línea]. Disponible en: https://www.researchgate.net/publication/309413369_MISP_-The_Design_and_Implementation_of_a_Collaborative_Threat_Intelligence_Sharing_Platform

[97] CIRCL, “MISP - Open Source Threat Intelligence Platform,” [En línea]. Disponible en: https://www.circl.lu/services/misp-malware-information-sharing-platform/, [Accedido: 11-06-2022].

[98] OASIS, “Introduction to STIX,” [En línea]. Disponible en: https://oasis-open.github.io/cti-documentation/stix/intro.html, [Accedido: 11-06-2022].

[99] OASIS, “Introduction to TAXII,” [En línea]. Disponible en: https://oasis-open.github.io/cti-documentation/taxii/intro.html, [Accedido: 11-06-2022].

[100] ANOMALI, “What Are STIX/TAXII,” [En línea]. Disponible en: https://www.anomali.com/resources/what-are-stix-taxii, [Accedido: 11-06-2022].

[101] Hyperledger, “Private Data Collections: A High-Level Overview,” 23-10-2018. [En línea]. Disponible en: https://www.hyperledger.org/blog/2018/10/23/private-data-collections-a-high-level-overview, [Accedido: 11-06-2022].

[102] FIRST, “Common Vulnerability Scoring System SIG,” [En línea]. Disponible en: https://www.first.org/cvss/, [Accedido: 11-06-2022].

Apéndice A
Casos de uso

Los siguientes casos de uso han sido obtenidos a partir de las distintas combinaciones de los requisitos en las que el usuario puede interactuar con el sistema y recibir una respuesta de parte suya. Se incluye en este apéndice una detallada descripción de los diferentes escenarios que existen por cada caso de uso, tanto, el flujo normal de ejecución de cada funcionalidad, como, los diferentes casos de error que pueden llegar a darse.

A.1. Casos de uso de gestión de registros VEC

Caso de uso (RF.1.1 y RF.1.3): Crear un nuevo registro VEC o combinarlo	
Prioridad	Alta
Descripción	Un usuario podrá almacenar una vulnerabilidad en un registro VEC, que podrá crear o combinar con uno existente si su vulnerabilidad está ya registrada.
Precondiciones	El usuario debe haber iniciado sesión en la plataforma.
Actor principal	Usuario con perfil cargado de rol gestor o administrador.
Escenario principal	1.-El usuario accede a la pantalla de crear un registro VEC. 2.-El usuario introduce los atributos de la vulnerabilidad, como la descripción, la solución, referencias, palabras clave, etc. Y envía los datos al sistema para que compruebe si ya existen registros similares. 3.-El sistema muestra la opción para crear nuevo registro VEC, al no haberse encontrado similares. 4.-El usuario selecciona la opción de crear. 5.-El sistema notifica de que la operación se ha realizado con éxito. 6.-El usuario es redirigido a la pantalla principal.

Continuación: Caso de uso (RF.1.1 y RF.1.3): Crear un nuevo registro VEC o combinarlo	
Flujos alternativos	2.a-El usuario no rellena los campos obligatorias de descripción, dispositivos o solución (si confirmó que la vulnerabilidad está solucionada). 2.a.1-El sistema muestra mensajes de error en los campos que sean necesarios. 2.a.2-El usuario rellena esa información que se le solicita y envía los datos. 2.a.3-Sigue por el paso 3. 3.a.1-El usuario opta por combinar los registros VEC. 3.a.2-El sistema muestra una pantalla con datos de ambos registros VEC y controles para seleccionar los atributos de un registro para permanecer en la fusión. 3.a.3-El usuario elige por atributos, cual es el que considera más conveniente y envía los datos para que se combinen. Continúa por el paso 5.

Caso de uso (RF.1.2): Modificar un registro VEC existente	
Descripción	Un usuario podrá modificar un registro VEC, para especificar mejor a que entidades afecta, qué nivel de riesgo conlleva, e incluso, darle una solución.
Actor principal	Usuario con perfil cargado de rol gestor o administrador.
Prioridad	Alta
Precondiciones	El usuario debe haber iniciado sesión en la plataforma. Y el registro VEC que se quiere modificar debe existir y no estar desactivado.
Escenario principal	1.-El usuario accede a la pantalla de detalle de un registro VEC. 2.-El usuario selecciona la opción para editar el registro VEC. 3.-El sistema muestra un mensaje de aviso y pide el consentimiento del usuario para tratar sus datos y exigir responsabilidad por las modificaciones que se realicen, en caso de que se detecte alguna anomalía con las vulnerabilidades. 4.-El usuario da su consentimiento y accede a la pantalla para editar el registro VEC. 5.-El usuario modifica los campos del registro VEC que necesite y envía los datos. 6.-El sistema notifica al usuario que el registro VEC se ha editado correctamente. 7.-El usuario es redirigido a la pantalla principal.
Flujos alternativos	2.a-El usuario deja vacíos los campos obligatorias de descripción, dispositivos o solución (si confirmó que la vulnerabilidad está solucionada). 2.a.1-El sistema muestra mensajes de error en los campos que sean imprescindibles. 2.a.2-El usuario rellena eso campos y envía los datos. 2.a.3-Sigue por el paso 6. 6.a-El sistema notifica al usuario de que se ha producido algún error al modificar el registro VEC. Vuelta al paso 5.

Caso de uso (RF.1.4): Importar registros VEC de fuentes externas de vulnerabilidades	
Descripción	El administrador podrá elegir de entre un listado de vulnerabilidades traidas de otras fuentes de vulnerabilidades (NVD del NIST), cuales incorporar al sistema, adaptándolas al formato de los registros VEC.
Prioridad	Alta
Actor	Administrador
Precondiciones	Usuario administrador con sesión iniciada.
Escenario principal	1.-El usuario accede a la pantalla de importar registros VEC. 2.-El sistema ofrece al usuario una serie de registros CVE, por defecto, obtenidos de NVD para incorporar en la plataforma VECsg 3.-El usuario selecciona un registro CVE para incorporarlo. 4.-El sistema carga el CVE en una pantalla de gestión de registros VEC, donde adapta la vulnerabilidad al estándar de exposiciones de VECsg. 5.-El usuario completa los campos que estime convenientes y envía el nuevo registro VEC importado al sistema. 6.-El sistema muestra una notificación para informar de que el registro VEC se ha creado correctamente. 7.-El usuario es redirigido a la pantalla principal.
Flujos alternativos	2.a-El usuario da valores para buscar registros CVE en la NVD. 2.a.1-El sistema ofrece un listado de registros CVE correspondientes con esa búsqueda y sigue por el paso 3. 3.a-El usuario selecciona un registro CVE para descartarlo. 3.a.1-El sistema muestra la lista sin el registro y sigue por el paso 3. 5.a-El usuario deja vacíos los campos obligatorias de descripción, dispositivos o solución, cuando afirmó que sí tenia. 5.a.1-El sistema muestra mensajes de error en los campos que faltan. 5.a.2-El usuario rellena esos campos y los envia. Sigue el paso 6. 6.a-El sistema notifica al usuario de que se ha producido algún error al crear el registro VEC. Vuelta al paso 5.

Caso de uso (RF.1.5 y RF.1.6): Gestionar registos VEC favoritos	
Descripción	Un usuario podrá marcar como favorito un registro VEC para estar informado de cualquier cambio que se produzca en él, y luego poder consultarlo.
Prioridad	Media
Actor principal	Usuario con perfil cargado de rol espectador, gestor o administrador.
Actor secundario	Usuario con perfil cargado de rol gestor o administrador.
Precondiciones	El usuario debe haber iniciado sesión en la plataforma. Y debe existir algún registro VEC que este usuario no ha marcado como favorito.
Escenario principal	1.-El usuario accede al detalle de un registro VEC. 2.-El sistema carga la opción de marcar como favorito al comprobar que no es uno favorito del usuario. 3.-El usuario decide guardar el registro VEC en favoritos. 4.-El sistema actualiza la vista de detalle para que aparezca como favorito. 5.-Un segundo usuario realiza cambios sobre el registro VEC favorito del usuario principal. 6.-El sistema envía un correo notificando al usuario principal. 7.-El sistema muestra una notificación en la plataforma web para avisar al usuario de los cambios. 8.-El usuario accede a la lista de favoritos. 9.-El sistema muestra la lista de favoritos con ese registro VEC. 10.-El usuario accede a ese registro VEC. 11.-El sistema muestra el resumen del último cambio del registro.
Flujos alternativos	2.a-El sistema carga la opción de desmarcar como favorito al comprobar que el registro VEC, es uno favorito del usuario. 2.a.1-El usuario selecciona la opción de desmarcar como favorito. 2.a.2-El sistema actualiza la vista de detalle para que no aparezca como favorito. 2.a.3.-El usuario accede a la lista de favoritos. 2.a.4.-El sistema muestra la lista de favoritos sin ese registro VEC.

Caso de uso RF.1.7: Explorar el historial de un registro VEC	
Descripción	Un administrador podrá acceder a la información de todos los cambios que ha recibido un registro VEC para poder conocer exactamente cuáles han sido los cambios, cuándo se han producido y quién ha sido el responsable de estos cambios.
Prioridad	Media
Actor	Administrador
Precondiciones	El usuario debe tener permisos de administrador y dirigirse a la pantalla de detalle de un registro VEC.
Flujo normal	Seleccionar la pestaña para ver el historial del registro VEC y consultar cada cambio para ver más detalles.
Estado final	El administrador visualiza la pantalla de acceso al historial completo de un registro VEC y resumen bien informado de todos sus cambios.

Caso de uso RF.1.8: Filtrar registros VEC por parámetros	
Descripción	Un usuario podrá filtrar entre todos los registros VEC por estado, entidad que afecta, autor, fecha, etc., para facilitar el acceso a los registros VEC que le interesen.
Prioridad	Media
Actor principal	Usuario con perfil cargado de rol espectador, gestor o administrador
Precondiciones	El usuario debe haber iniciado sesión en la plataforma..
Escenario principal	1.-El usuario accede a la pantalla principal de VECsg. 2.-El usuario marca la opción para mostrar los filtros avanzados. 3.-El sistema muestra opciones de fecha, entidades, autores, dispositivos y riesgo, entre otras, para realizar una búsqueda. 4.-El usuario introduce los parametros que le interesen y realiza la búsqueda. 5.-El sistema muestra en la tabla de resultados los registros VEC obtenidos.
Flujos alternativos	5.a-El sistema no encuentra ningún registro VEC que coincida con la búsqueda y muestra un mensaje correspondiente.

Caso de uso RF.1.9: Desactivar un registro VEC	
Descripción	Un administrador podrá desactivar un registro VEC cuando ya no se considere relevante o necesario. Desactivar un registro VEC no hará que se elminine completamente de la aplicación, si no que desaparecerá de todas las listas y no aparecerá en ninguna búsqueda. Se incluirá en la lista de registros VEC desactivados, de acceso exclusivo para administradores.
Prioridad	Media
Actor	Administrador
Precondiciones	El usuario debe ser administrador y acceder al detalle de un registro VEC.
Flujo normal	Seleccionar la opción de desactivar el registro VEC.
Estado final	El registro VEC está desactivado y oculto en la plataforma.
Estado error	El sistema informa de cualquier error que haya podido ocurrir.

Caso de uso RF.1.10: Consultar el estado actual de la plataforma	
Descripción	Un usuario podrá acceder a diversos gráficos con información actual del sistema, para que de forma clara y visual pueda conocer cuántos registros VEC hay en la plataforma, cómo de críticos son o cuales son los dispositivos más vulnerables.
Prioridad	Baja
Actor	Usuario espectador, gestor o administrador.
Precondiciones	El usuario debe haber iniciado sesión.
Flujo normal	El usuario accede a la página de estadísticas.
Estado final	Pantalla cargada con información cuantitativa sobre los nuevos registros VEC y los acumulados, sus riesgos, los dispositivos más vulnerados, etc.

A.2. Casos de uso de autenticación y registro de usuarios

Caso de uso RF.2.1 y RF.2.5: Registrarse en el sistema	
Descripción	Un nuevo usuario puede acceder a VECsg, registrándose en la plataforma y recibiendo la aprobación del consorcio
Prioridad	Alta
Actor principal	Usuario no registrado
Actores secundarios	Resto de usuarios de la organización
Precondiciones	La persona que se quiere registrar debe ser un individuo conocido dentro del consorcio, puesto que, los impostores van a ser descartados.
Escenario principal	1.-El usuario accede a la pantalla de registrarse en VECsg. 2.-El usuario introduce sus datos personales y procede a registrarse. 3.-El sistema comprueba que los datos son correctos. 4.-El usuario es redirigido a una pantalla donde se le informa de que antes de entrar tiene que ser confirmada su petición por el consorcio. 4.1.-El sistema envía correos con la petición del usuario a diferentes miembros de su misma organización. 4.2.-Estos usuarios al recibir el correo, pueden aceptarlo en su organización en VECsg o rechazarlo. 4.3.-El usuario recibe los suficientes aprobados y se sigue por el paso 5. 5.-El sistema envía una notificación al usuario candidato, informándole de que ha sido aceptado. 6.-El usuario accede correctamente a la aplicación con su perfil y su rol propios.

Continuación: Caso de uso RF.2.1 y RF.2.5: Registrarse en el sistema	
Flujos alternativos	2.a.-El usuario se registra mediante OAauth. 2.a.1-El sistema muesta el formulario de registro con parte de los datos obtenidos de OAuth. 2.a.2-El usuario termina de completar su registro y envía los datos. 2.a.3-Sigue por el paso 3. 2.b-El sistema devuelve un mensaje de error, porque ya existe un usuario con el nombre introducido. 2.b.1-Vuelve al paso 2. 3.a-El sistema muestra un mensaje de error, debido a que los datos del usuario son incompletos o incorrectos. 3.a.1-Vuelta al paso 2.

Caso de uso RF.2.2 y RF.2.4: Iniciar sesión en la plataforma	
Descripción	Introduciendo unas credenciales válidas, un usuario activo puede acceder a la plataforma y hacer uso de los servicios de VECsg.
Prioridad	Alta
Actor	Usuario
Precondiciones	El usuario debe estar registrado y contar con la aprobación del consorcio.
Escenario principal	1.-El usuario accede a la pantalla de login de VECsg. 2.-El usuario introduce sus credenciales y envía los datos. 3.-El sistema comprueba que las credenciales sean correctas. 4.-El sistema comprueba que las confirmaciones que ha recibido el usuario, son, al menos, una de un administrador o tres de usuarios. 5.-El usuario accede a la aplicación con su perfil y su rol cargados correctamente.
Flujos alternativos	2.a-El usuario decide iniciar sesión en VECsg, usando el servicio de autenticación OAuth. 2.a.1-Se comprueba que el usuario registrado con OAuth ya existe en el sistema. 2.a.2-Sigue por el paso 4. 2.a.1.a- El usuario aún no existe en el sistema. 2.a.1.a.1-Se continúa por el paso alternativo 2.a.1 del Caso de Uso *Registrarse en el sistema.* 3.a-Las credenciales del usuario no son correctas. 3.a.1-El sistema muestra un error claro y explicativo. 3.a.2-Vuelta al paso 2. 4.a-El sistema comprueba que el número de confirmaciones que ha recibido el usuario no es el suficiente. 4.a.1-El sistema muestra una pantalla donde se explica la situación. 4.a.2-El usuario es redirigido al paso 1.

A.3. Casos de uso de gestión de usuarios

Caso de uso RF.3.1 y RF.3.2: Crear y editar usuarios	
Descripción	Un usuario previamente registrado que haya sido confirmado por un número suficiente de miembros de sus organización podrá iniciar sesión en el sistema con su propio perfil cargado.
Prioridad	Alta
Actor	Administrador
Precondiciones	El usuario administrador debe acceder a la pantalla de gestionar usuarios.
Escenario principal.	1.-El usuario accede a la opción de crear nuevo usuario. 2.-El sistema carga un formulario para generar un nuevo usuario. 3.-El usuario rellena los campos y manda crear un nuevo usuario. 4.-El sistema envía una notificación de que el usuario se ha creado correctamente. 5.-El usuario es redirigido a la pantalla principial de gestionar usuarios, viendo el usuario nuevo. 6.-El usuario selecciona el usuario para modificarlo. 7.-El sistema muestra un formulario con los datos que se pueden editar (nombre de pila, correo electrónico, etc.). 8.-El usuario edita los campos y manda ejecutar los cambios. 9.-El sistema envía una notificación de que el usuario se ha modificado sin problemas.
Flujos alternativos	4.a-El sistema devuelve un mensaje de error, porque ya existe un usuario con el nombre introducido o los datos del usuario son incompletos o incorrectos. 4.a.1-Vuelve al paso 3. 9.a-El sistema detecta algún error con los datos cambiados o al realizar la modificación, y avisa al usuario. 9.a.1-Vuelve al paso 8.

Caso de uso RF.3.3: Activar/Desactivar usuarios	
Descripción	Un administrador podrá desactivar usuarios para quitarles el acceso a la platafoma, pero hay que tener en cuenta que todos sus movimientos y acciones realizadas hasta la fecha no se perderán. Posteriormente el administrador puede volver a activar el usuario para otorgarle de nuevo la posibilidad de entrar en el sistema.
Prioridad	Alta
Actor	Administrador
Precondiciones	El usuario debe ser administrador y acceder a la pantalla para gestionar usuarios.
Flujo normal	Se selecciona la acción que se quiera llevar a cabo para cada usuario.
Estado final	Los usuarios que se quieran bloquear, quedaran desactivados y si es necesario, alguno ha podido se reactivado. Los usuarios reciben un correo si vuelven a ser activados
Estado error	El sistema informa al usuario ante cualquier error.

Apéndice B
Manual de usuario

En este manual de usuario se explicarán, paso por paso, todas y cada una de las funcionalidades que ofrece la plataforma web VECsg. Se ofrece un amplio recorrido a través de toda la aplicación, explorando sus flujos principales y descubriendo al usuario todas las posibilidades que ofrece este sistema.

B.1. Registro de un nuevo usuario en la aplicación

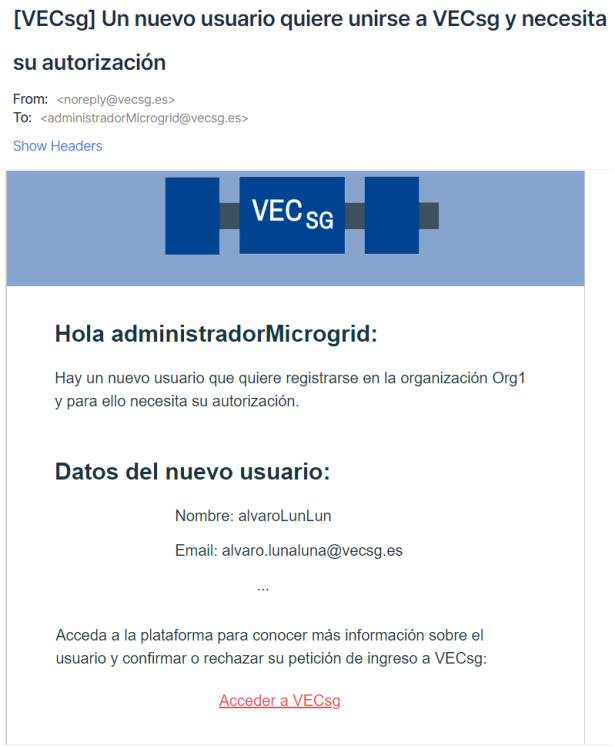
Para que un usuario pueda acceder a la plataforma debe crear un nuevo usuario y hacer una petición al consorcio para ser incluido en este dominio cerrado. Esto se consigue desde la pantalla de *Registrar Usuario* (véase la figura 1). Una vez creado el usuario, será necesario que el consorcio lo identifique y apruebe para poder acceder a VECsg, por lo que, hay que esperar a su decisión.

Figura 1: Detalle de la pantalla de *Registrar Usuario*

Figura 2: Pantalla de aviso de falta de confirmación del usuario

En ese momento reflejado en la figura 2, el resto de usuarios de la organización deben prestar atención al correo (figura 3) en el que se les propone verificar un nuevo usuario que quiere unirse a su organización en VECsg.

Figura 3: Correo que reciben los usuarios de la organización para evaluar al nuevo usuario



En cuanto acceda al enlace y el usuario inicie sesión en la plataforma, accederá a una pantalla para aceptar o rechazar al usuario, que se puede observar en la figura 4.

Figura 4: Pantalla para confirmar o rechazar al usuario nuevo

Cuando el usuario adquiera las suficientes confirmaciones, recibirá un aviso como el de la figura 5, de que ha sido aceptado por su organización en VECsg.

Figura 5: Correo felicitando al nuevo usuario por su admisión en la plataforma

B.2. Iniciar sesión en VECsg y página principal de VECsg

En el instante en el que sea aprobada su solicitud de entrar en la aplicación, ya puede acceder a la plataforma a través de la pantalla de *Inicio de sesión* de la figura 6.

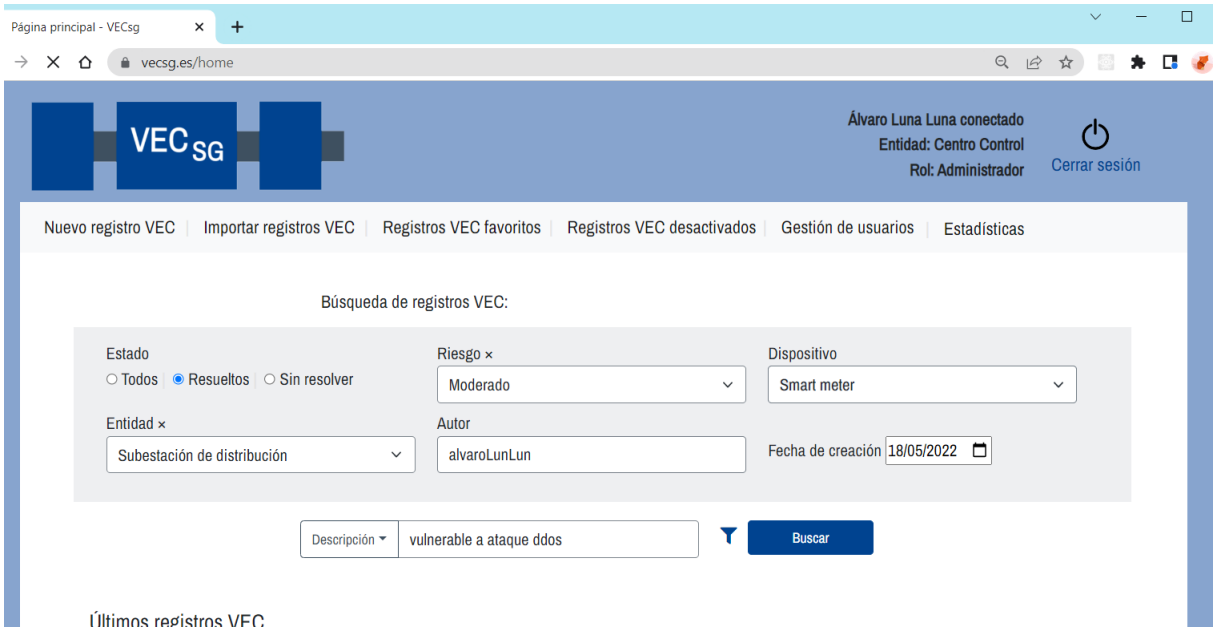
Figura 6: Inicio de sesión en la plataformaPantalla principal de VECsg

Entonces será redirigido a la página principal (figura 7) para poder acceder a todos los servicios que ofrece VECsg. Se podrán acceder desde el menú a las pantallas para crear registros VEC o para importarlos desde otras plataformas, consultar registros VEC desactivados o aquellos registros que el usuario considere más interesantes y que siga como favorito. También, desde aquí, se podrá acceder a estadísticas sobre los datos que circulan por el sistema, e incluso, llevar a cabo una gestión rigurosa de usuarios. Todas estos servicios están completamente disponibles para los administradores. En el caso de los usuarios estándar, están restringidas operaciones críticas como la gestión de usuarios o de registros VEC desactivados, estos últimos, simplemente dejan de estar visibles para ellos.

Figura 7: Pantalla principal de VECsg

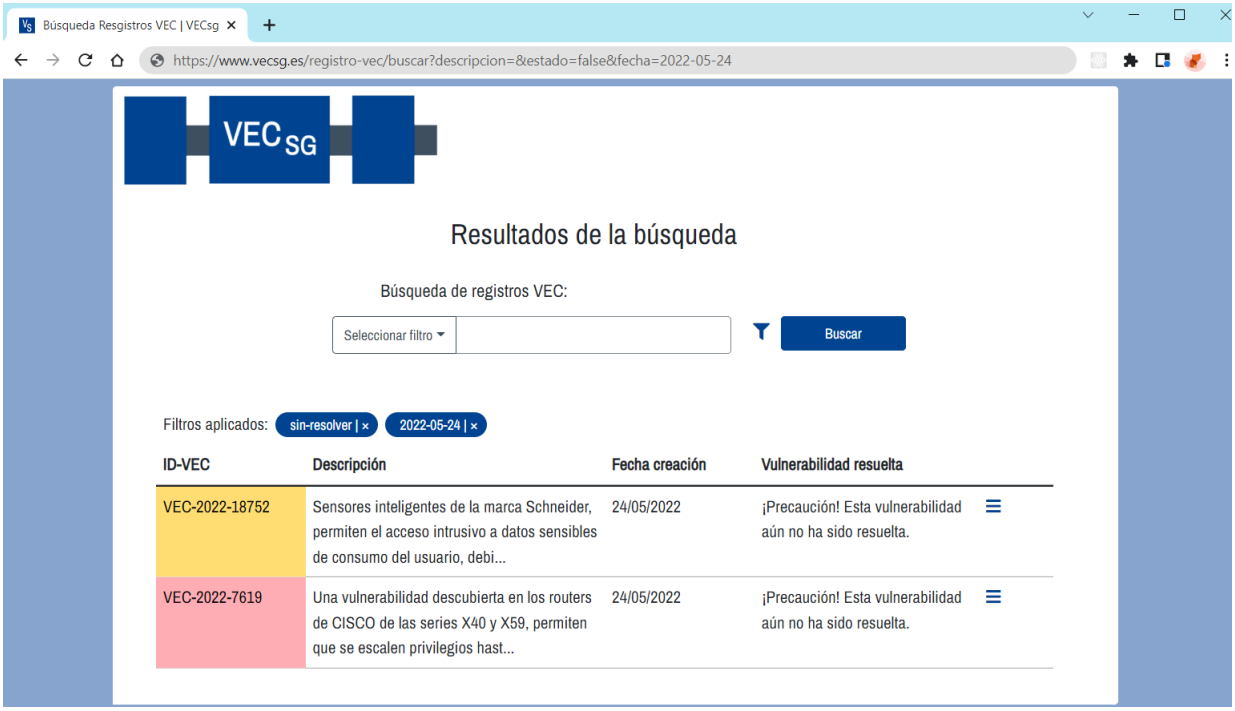
Una de las grandes posibilidades que ofrece VECsg, es un sistema completo de búsqueda, como el observado en la figura 8, para filtrar entre todos los registros VEC y obtener los que se quieran, realizando dicha búsqueda, por fechas, estado, entidades, dispositivo, etc.

Figura 8: Sistema de búsqueda de registros VEC por filtros



Por ejemplo, se pueden filtrar los registros VEC por un determinado día (24-05-2022), combinándolo con un parámetro que establezca que solo se tengan en cuenta los que aún no tienen solución, tal y como se muestra en la figura 9.

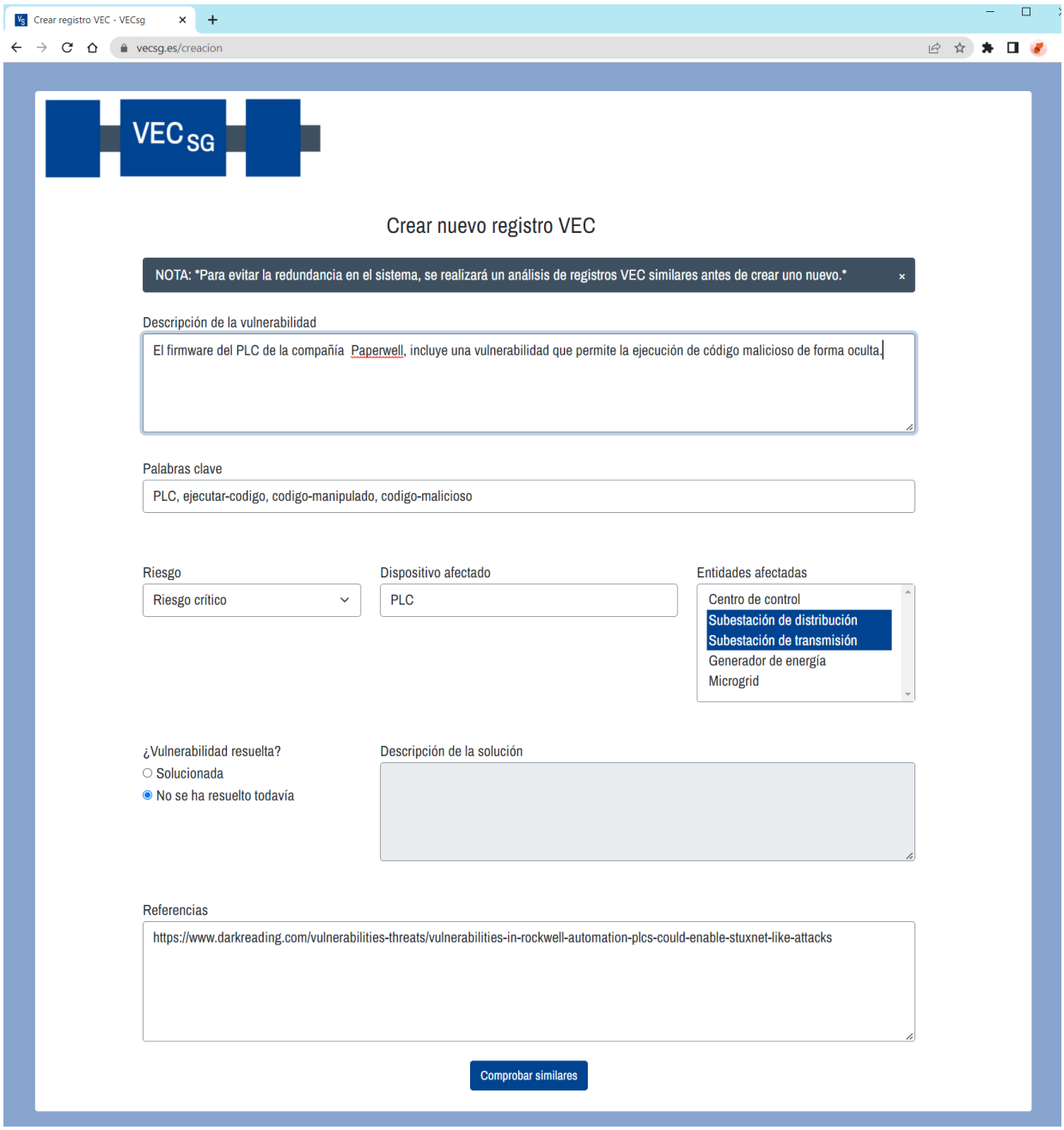
Figura 9: Ejemplo de búsqueda con los parámetros: día 24-05-2022 y no resueltos



B.3. Creación de un nuevo registro VEC

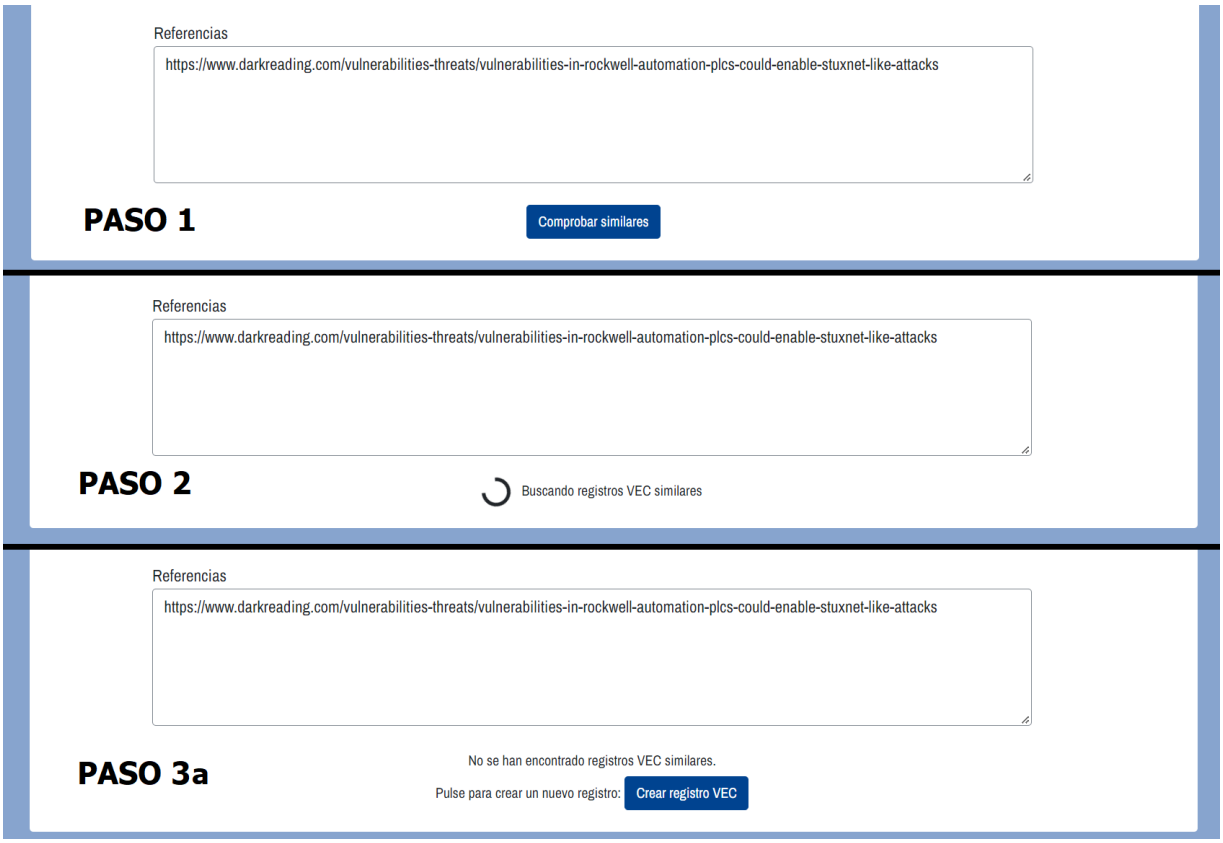
Para crear un nuevo registro VEC, accedemos a la pantalla de *Nuevo registro VEC* (véase la figura 10). En esta pantalla apareceran numeros campos para incluir toda la información que necesitemos sobre la vulnerabilidad que se quiere guardar en VECsg. Entre los valores que podemos introducir se encuentra, la descripción de la vulnerabilidad, entidades de la Smart Grid afectadas, dispositivos o, en caso de que exista, una solución.

Figura 10: Formulario de creación de registros VEC



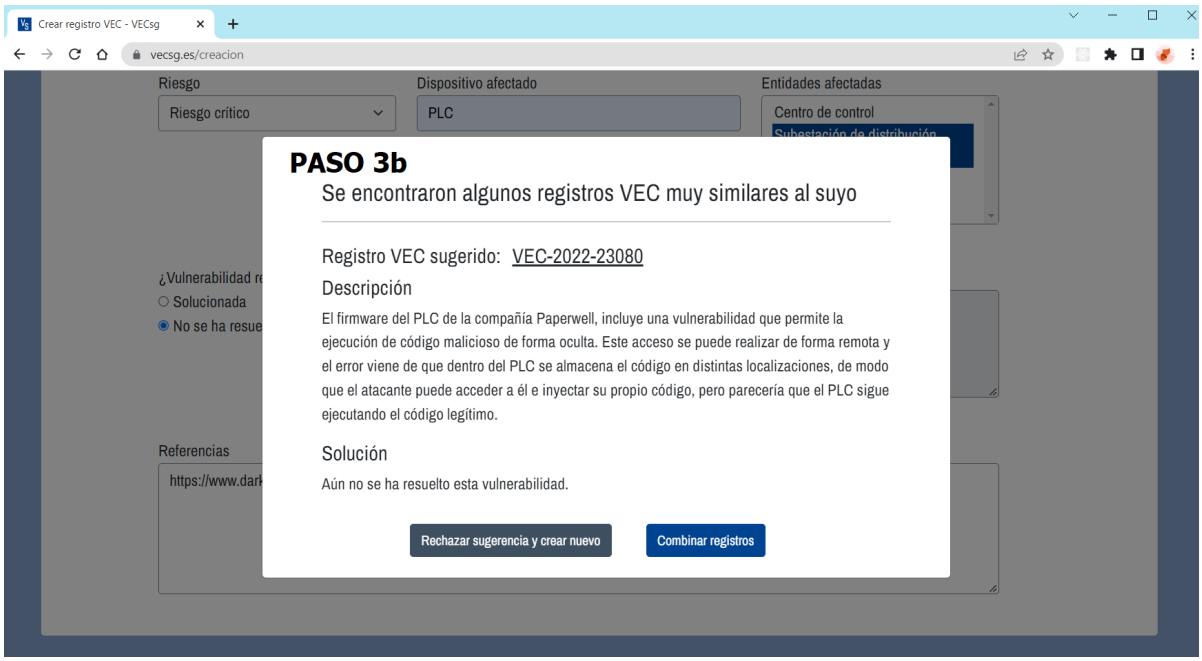
Una vez que se ha completado el formulario de creación de un nuevo registro, el sistema debe realizar un análisis de registros VEC similares para comprobar que no exista uno igual ya en el repositorio (*pasos 1 y 2* de la figura 11). Entonces, aquí pueden darse dos escenarios. Si el registro VEC es único, en VECsg se puede crear uno nuevo con normalidad (*paso 3a* en la propia figura 11).

Figura 11: Escenario de creación cuando el nuevo registro VEC es original



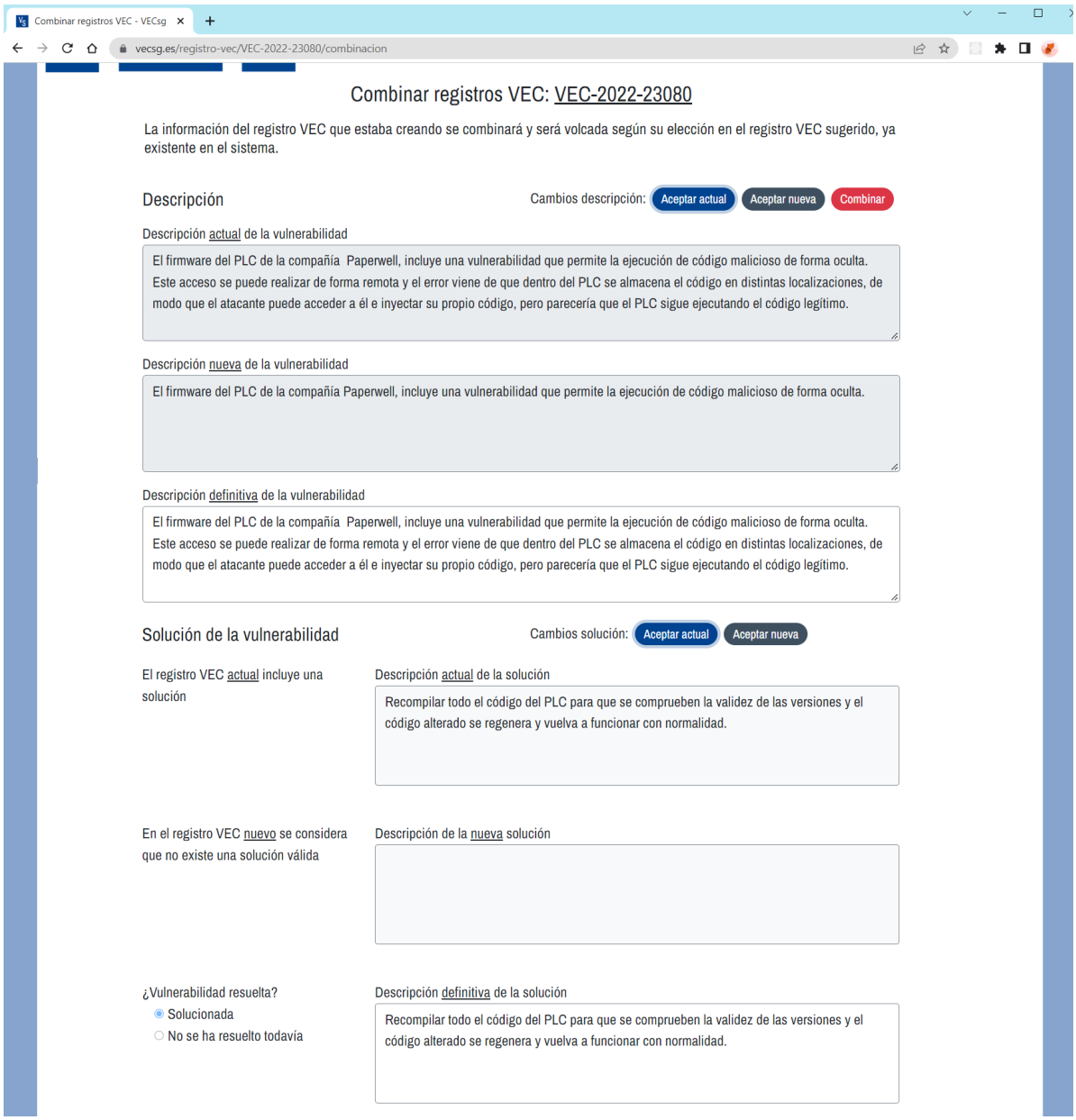
El otro escenario es que se encuentre un registro VEC similar, que se muestra para que el usuario evalúe las similitudes (observar *paso 3b* en la figura 12) y entonces se pueda optar a combinar ambas vulnerabilidades mediante un proceso muy sencillo (observar figura 13). Sin embargo, si se considera que el registro VEC sugerido no es el adecuado y no es exactamante la misma vulnerabilidad que se está intentando registrar, siempre se puede rechazar la sugerencia y crear uno nuevo.

Figura 12: Escenario en el que hay registros VEC con información parecida



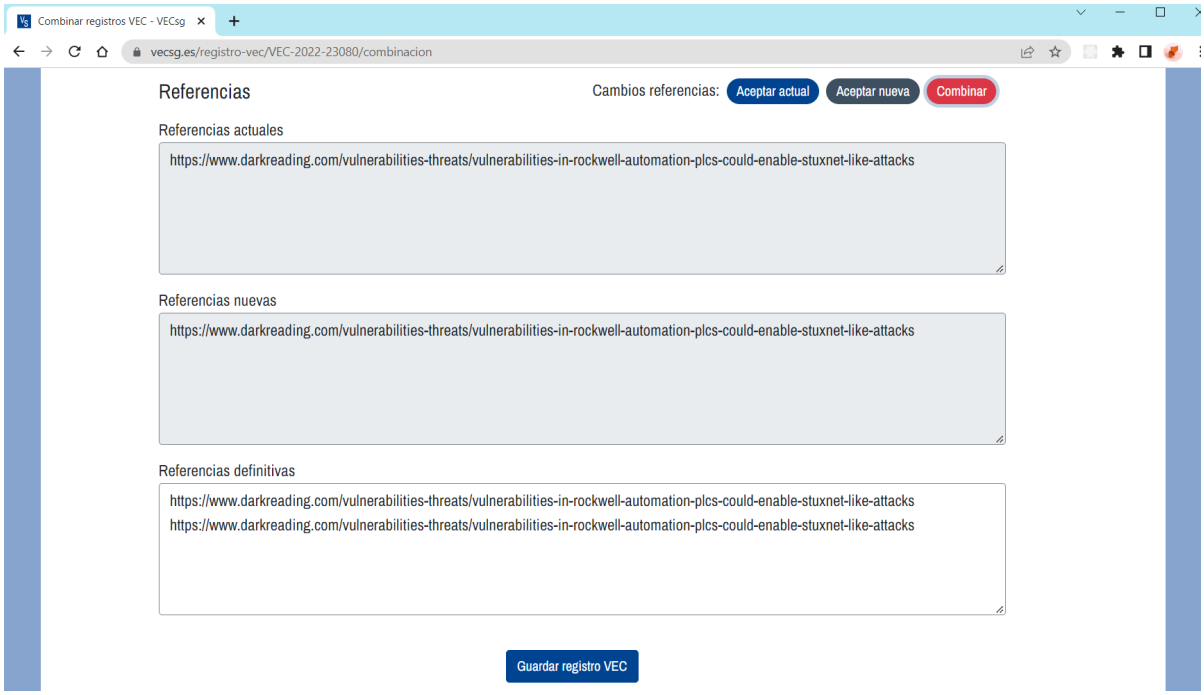
En caso de que sí queramos fusionarlos, al hacer click en el botón de *Combinar registros*, el usuario accede a la pantalla (figura 13) para combinar el registro VEC nuevo que pretendía crear, con el registro VEC que ya existe actualmente en VECsg, tal y como se muestra en la figura 13. Así, el usuario podrá decidir para cada atributo, el valor de qué registro VEC le parece más interesante, para que permanezca en el registro VEC resultado de la fusión.

Figura 13: Vista donde se aprecia cómo combinar la descripción similar de dos registros



Se puede apreciar algunos ejemplos de los cómodos controles que permiten combinar las descripciones, las soluciones o las referencias, en la pantalla de la figura 14. Donde tras elegir que opción se prefiere en cada casa, simplemente guardando los cambios, quedarán fusionados estos dos registros VEC.

Figura 14: Ejemplo de combinación de referencias de los registros VEC

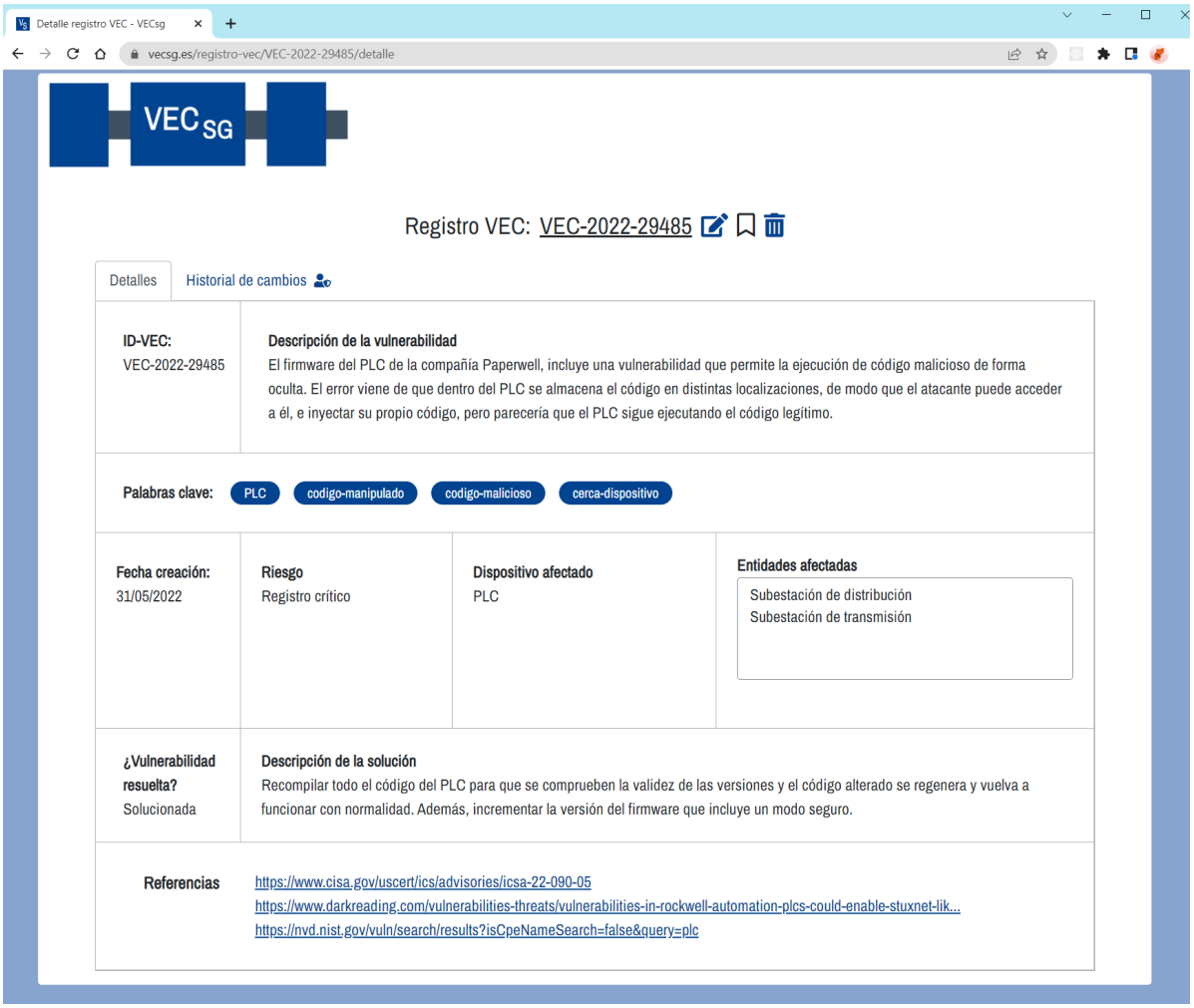


B.4. Detalle y modificación de un registro VEC

Se puede acceder al detalle de un registro VEC, haciendo clic en el icono de detalle de la fila del registro VEC en la tabla donde aparezca. También, se puede acceder directamente al registro VEC desde el cuadro de búsqueda de registros VEC, seleccionando la opción adecuada.

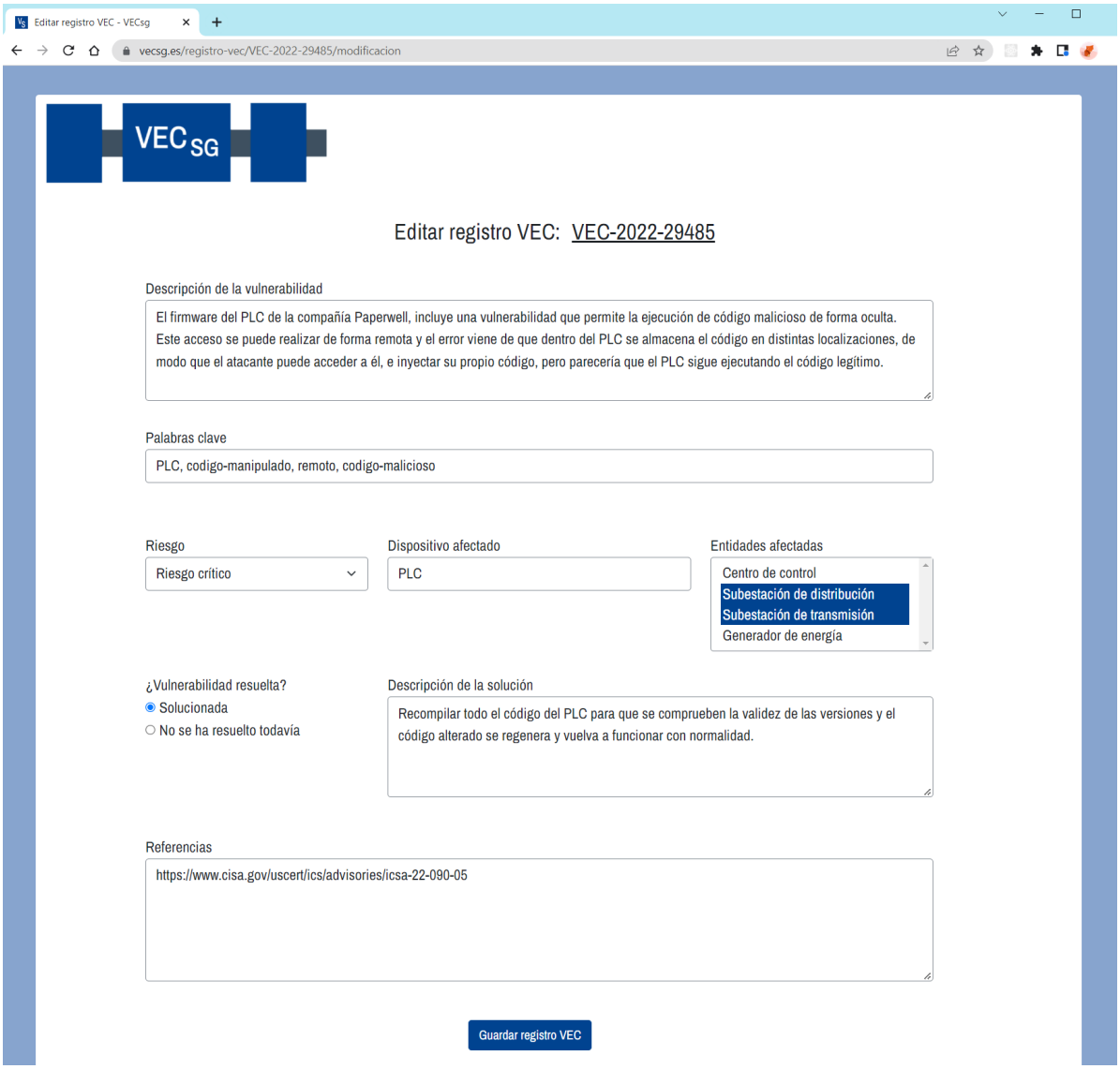
Dentro del detalle de un registro VEC, se puede consultar toda su información y acceder a opciones como marcarlo como favorito, eliminarlo (en caso de que se cuente con permisos de administrador) o modificarlo (véase la figura 15).

Figura 15: Resumen de la vista de detalle de un registro VEC



Si seleccionamos esta última opción, accederemos a un formulario donde se cargarán los datos del registro VEC, en una pantalla como la de la figura 16, para ajustar o corregir cualquier información sobre la vulnerabilidad que se estime conveniente, teniendo en cuenta que nuestros movimientos se registrarán y se consultarán, en caso de que haya algún problema con los registros VEC y se busquen responsables.

Figura 16: Pantalla de *Editar registro VEC*

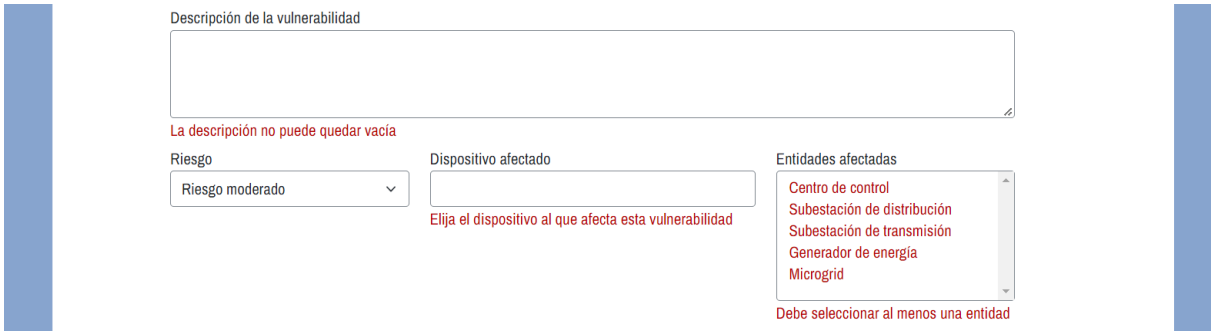


Es importante señalar, que antes de empezar a editar el registro VEC, tendremos que aceptar una serie de condiciones, sobre los permisos que hay que tener dentro de la organización para editar el registro VEC, el aviso de responsabilidad ante los cambios y consentir que todos los movimientos queden guardados, tal y como se ve en la figura 17.

Figura 17: Aviso de condiciones a aceptar para actualizar un registro VEC

Otra cuestión muy importante, tanto al editar como al crear registros VEC, es que se introduzcan correctamente todos los campos obligatorios, si esto no es así, el sistema mostrará un error y señalará cada campo que ha faltado por cumplimentar (figura 18).

Figura 18: Mensajes de error en campos sin rellenar



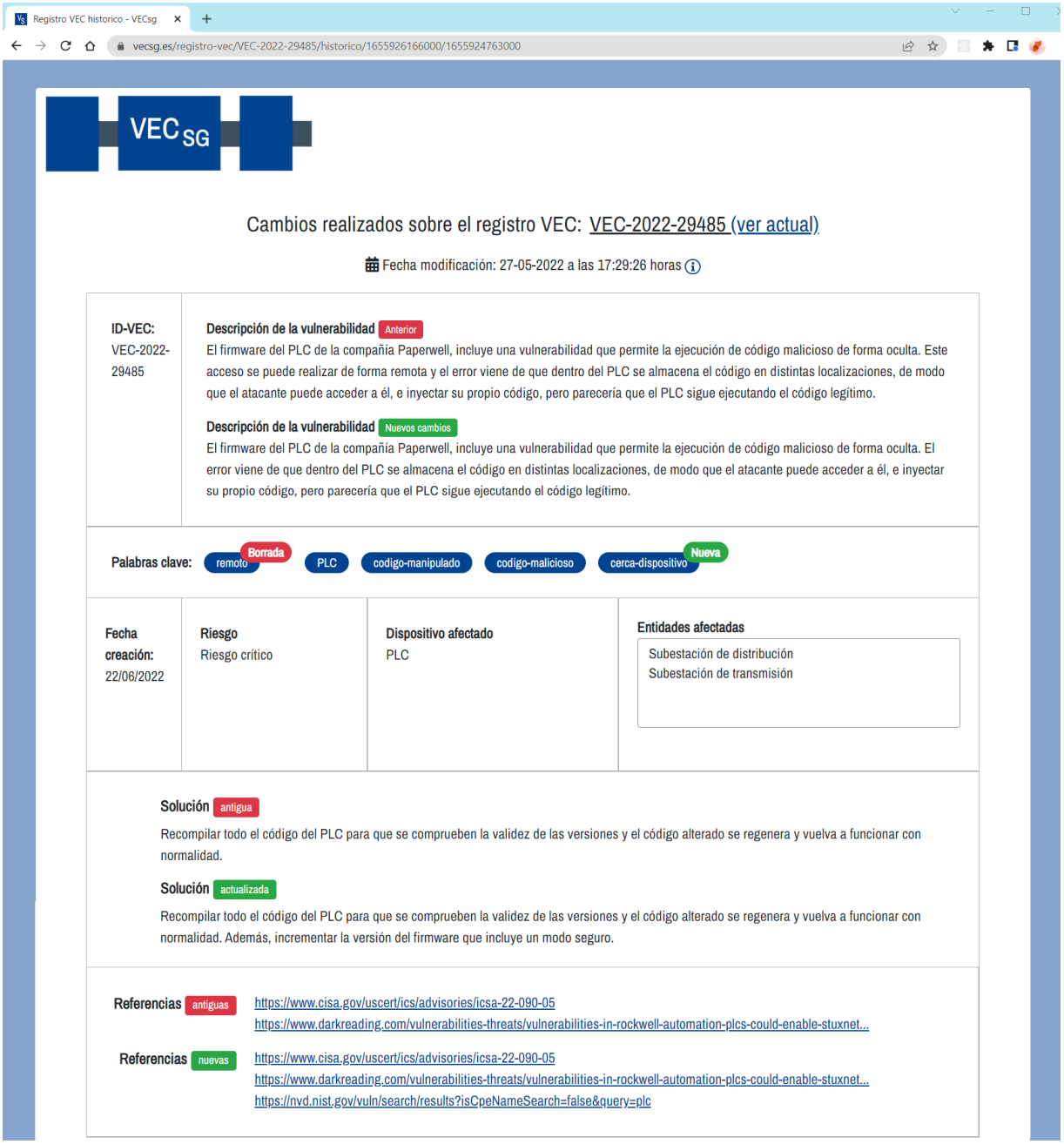
B.5. Trazabilidad y responsabilidad vinculada con el registro VEC

Si se cuentan con permisos de administrador, se puede acceder desde esta pantalla a una pestaña para ver quién es el responsable del registro VEC y consultar un historial de todos los cambios que se han producido en el registro, ordenados por fecha (figura 19). Además, se puede conocer para cada cambio, quién lo ha realizado y en qué ha consistido, accediendo al detalle de cada uno.

Figura 19: Resumen de los cambios y el origen de un registro VEC

Accediendo al detalle de un cambio, podremos conocer cuales son exactamente los campos que han cambiado y de qué manera, véase la figura 20.

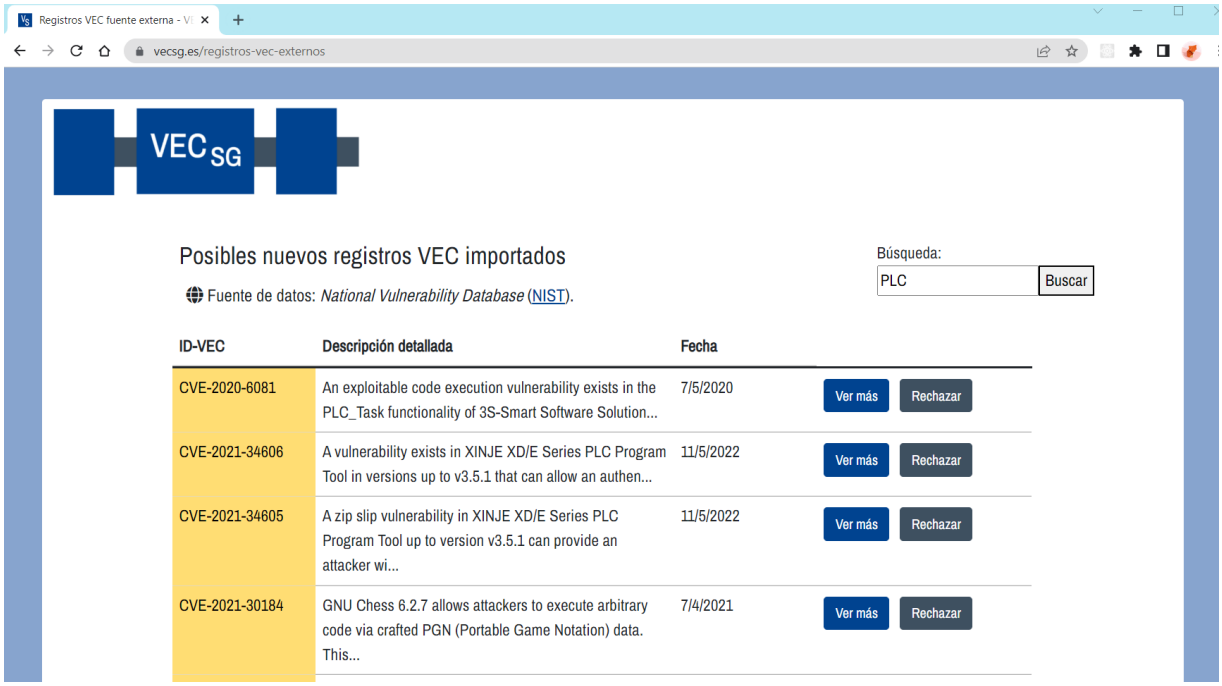
Figura 20: Detalle de los cambios realizados por un usuario sobre un registro VEC



B.6. Importar registros VEC desde fuentes externas

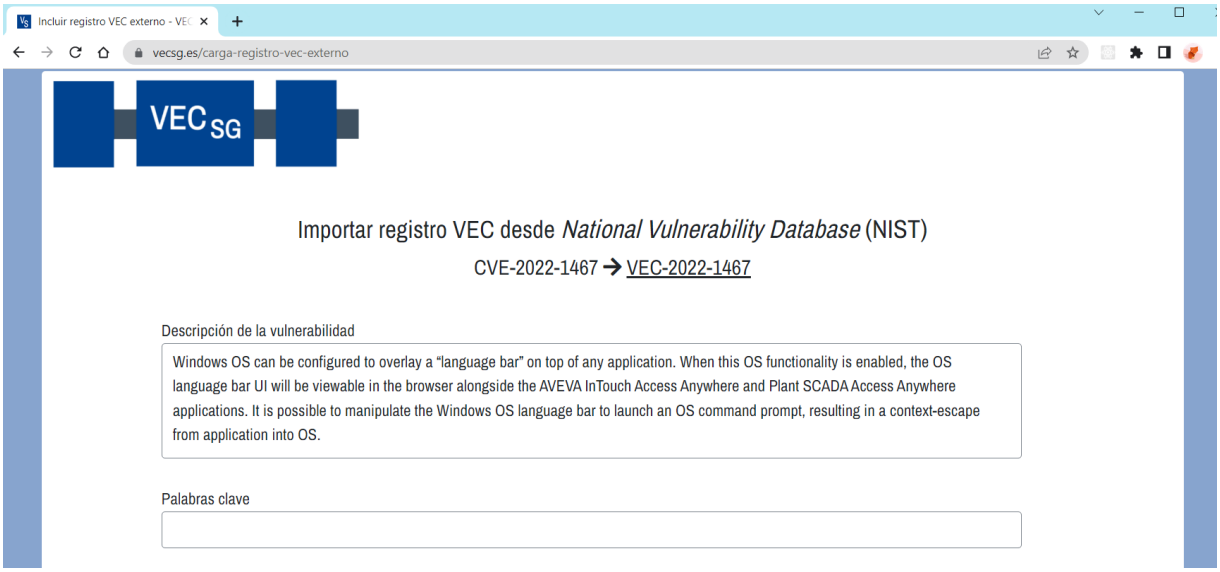
Para obtener registros VEC de otras fuentes, debemos acceder a *Importar registros VEC* en la pantalla principal y aparacerá una vista como la de la figura 21. Entonces, se muestran una serie de registros CVE traidos de la NVD, una base de datos de vulnerabilidades mantenida por el NIST. Podemos elegir entre estos registros para descartar la recomendación o incluirlos en VECsg. Para ello primero hay que adaptar el formato de la vulnerabilidades en CVE al formato particular que tienen los registros VEC, inspirado en el primero.

Figura 21: Apariencia de listado de los registos VEC sugeridos por la NVD del NIST



Si pulsamos en el *Ver más* de un registro CVE, accederemos a una pantalla (figura 22) donde se podrá realizar esta conversión. Los datos del registro CVE se cargarán en un registro VEC nuevo, en él se deben completar las características que tienen que ver con la Smart Grid, para poder importarlo a VECsg.

Figura 22: Pantalla de *Importar registro VEC*, con los datos extraídos



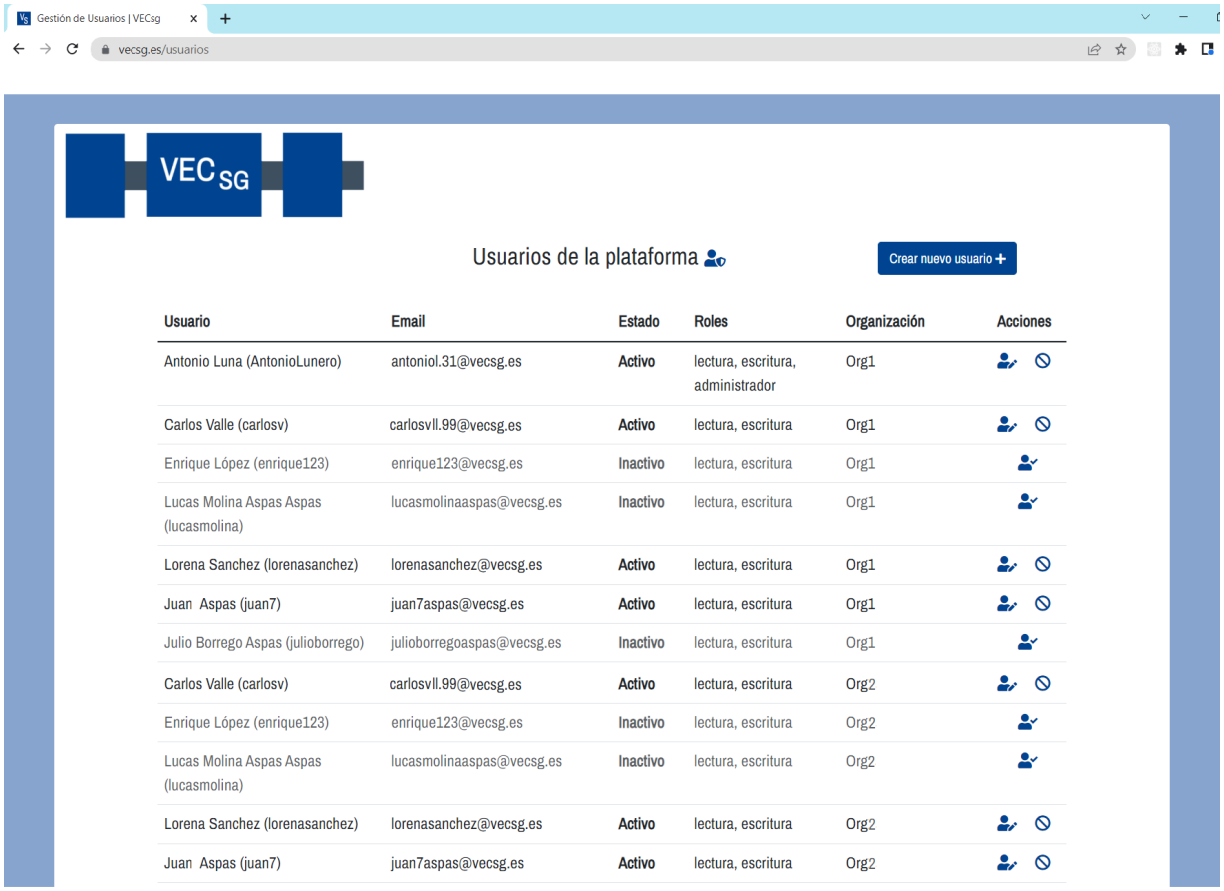
Datos como la descripción, el riesgo y las referencias, son tomadas directamente de la fuente externa de vulnerabilidades, en nuestro caso, la NVD.

Figura 23: Visualización de más información que se obtiene de la fuente externa.

B.7. Gestión de usuarios

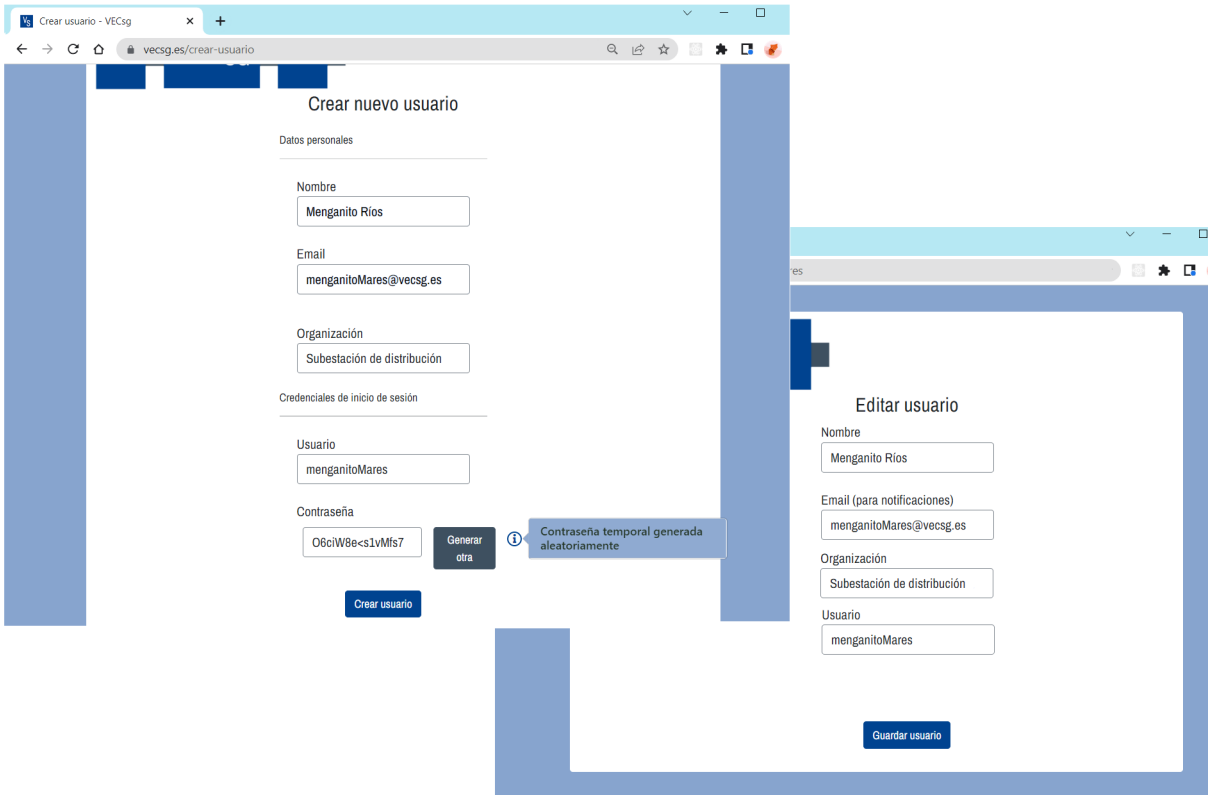
Si se tiene un perfil de usuario administrador, se puede realizar la gestión de usuarios de nuestra organización. Para ello, podemos acceder al listado completo de usuarios en la página *Gestión de usuarios* (figura 24). Desde esta pantalla podemos crear nuevos usuarios pulsando en la opción correspondiente, siempre que se tengan permisos elevados de administrador. También se podrán editar los usuarios ya creados, para modificar alguno de sus datos personales (ver las pantallas correspondientes en la figura 25).

Figura 24: Lista completa de los usuarios de una organización



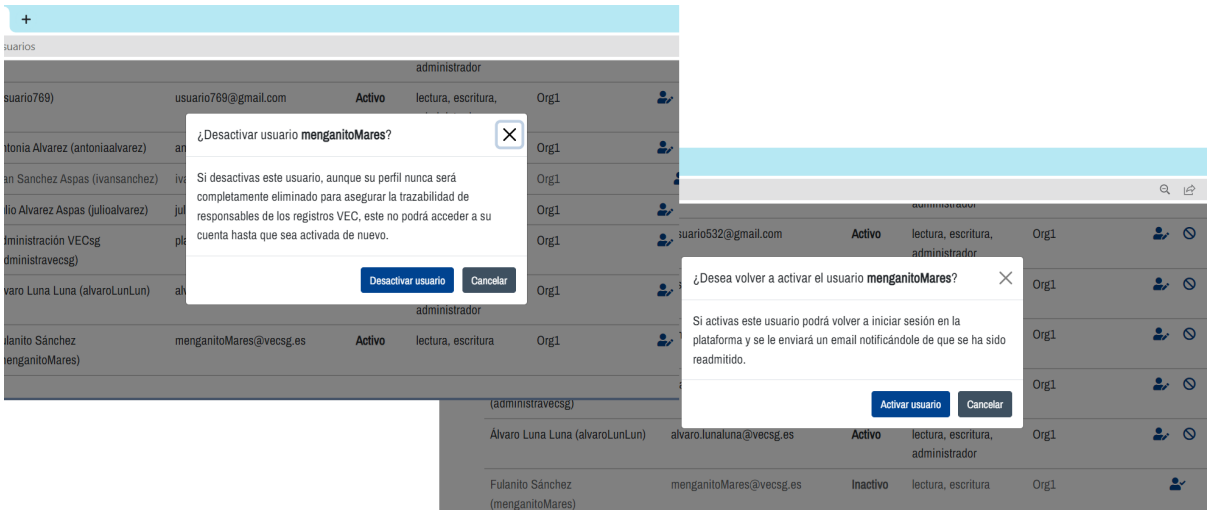
Cabe tener en cuenta que, como toda esta gestión de usuarios, es llevada a cabo por administradores, los usuarios creados de esta manera no necesitan ser aprobados por el consorcio, directamente son disponibles al considerarse que son usuarios bajo condiciones de confianza y legitimidad.

Figura 25: Pantallas de crear y editar usuario



Por último, para gestionar las responsabilidades, se pueden bloquear usuarios quitándoles el acceso, si por ejemplo, se considera que cometieron alguna irregularidad. Del mismo modo, se les puede devolver el acceso a la plataforma, activándolos de nuevo (observar figura 26).

Figura 26: Diálogos de bloqueo y activación de usuarios



Los usuarios que son aceptados de nuevo en la plataforma, reciben una notificación mediante un correo electrónico que tiene la forma de la figura 27.

Figura 27: Correo recibido por el usuario en cuanto es reactivado

B.8. Añadir un registro VEC a la lista de favoritos

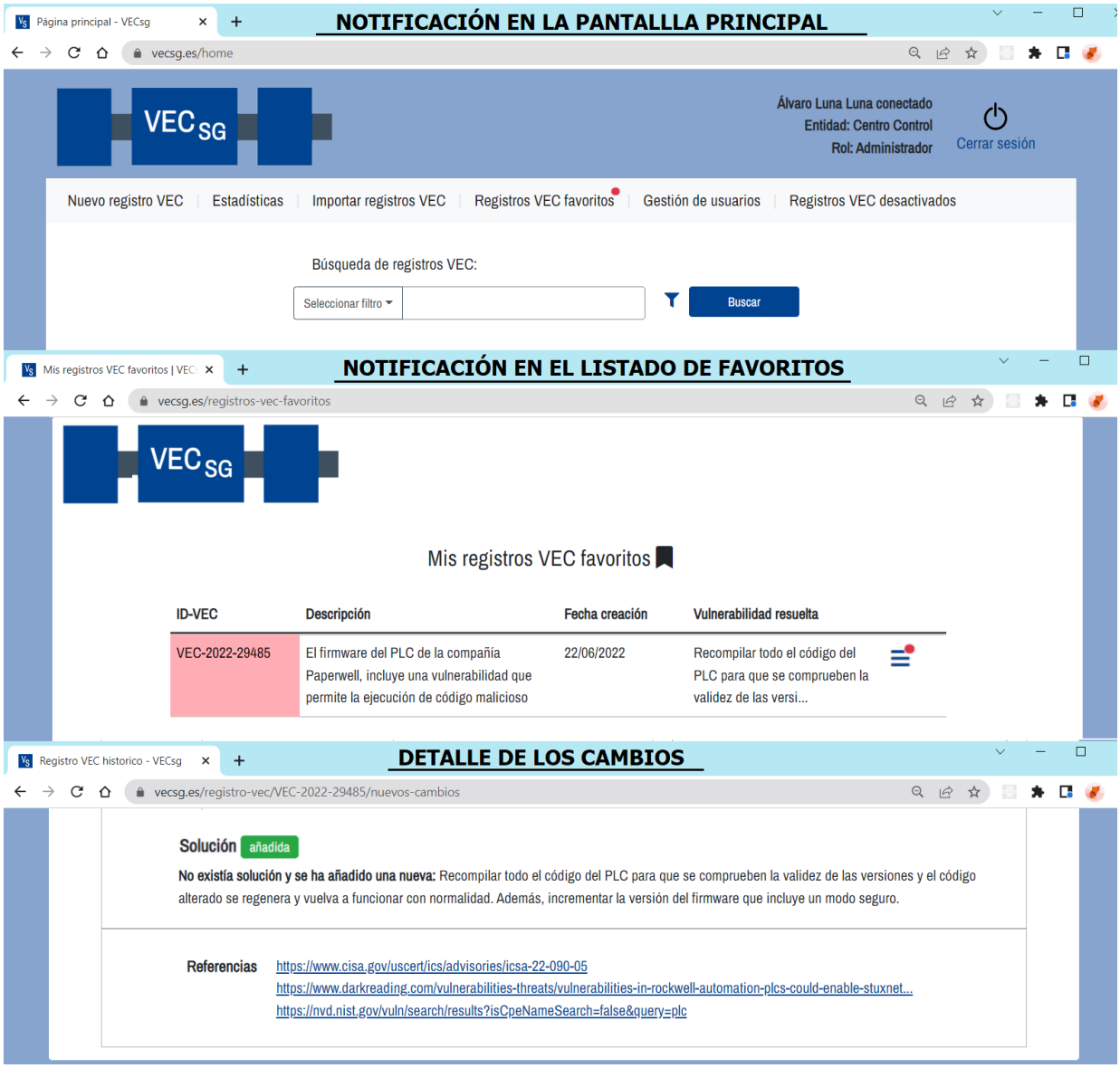
Para seguir un registro VEC y poder conocer todo lo que ocurre con él, se debe añadir como favorito. Para ello se puede acceder a la pantalla de det3lla de un registro VEC y marcar el icono correspondiente si este aún no es uno de nuestros registros favoritos, tal y como se muestra en la interfaz gráfica de la figura 28.

Figura 28: Guardar registro VEC como favorito

Figura 29: Correo de aviso de cambios en uno de los registros VEC favoritos del usuario

Desde el momento en que se empieza a seguir un registro VEC, cada vez que se produzca una cambio en él, recibiremos un correo avisando de ello. Luego, desde la lista favoritos, se puede consultar este registro VEC y ver cual es el cambio que se ha producido. Como este cambio es considerado de especial interés para el usuario, se le notificará mediante pequeñas señales rojas, que se pueden ver en la figura 30.

Figura 30: Despliegue del aviso de cambios en registros VEC favoritos



Apéndice C
Implementación de VECsg

En esta sección se va a explicar con detalle la implementación de las distintas funciones y métodos que permiten el correcto funcionamiento de este sistema distribuido y que los requisitos exigidos por el cliente sean satisfechos.

C.1. Detalle del código del Chaincode

En primer lugar, vamos a empezar mencionando las operaciones implementadas en el *chaincode*, que son las que permiten consultar y actualizar directamente los registros almacenados en la blockchain. Nos referimos en este apartado a los métodos del smart contract de crear, editar o borrar registros VEC.

Para crear un nuevo registro VEC es necesario que la función correspondiente en el *chaincode* (ver figura 1) reciba los parámetros exigidos para el registro con todos sus tipos adecuados. Luego, en esa misma función, se cargan todos estos datos en una instancia de tipo *RegistroVEC* y con la función *Marshal* se codifica al formato correcto, para poder ser incluido en la blockchain. Al final, utilizando la *Fabric Contract API*, concretamente el método *PutState*, se puede proponer una nueva transacción para que el registro quede grabado y disponible para toda la red de blockchain.

De forma similar se procede en el resto de funciones del *chaincode* aunque se adaptan a las particularidades de las funcionalidades requeridas en cada caso. Por ejemplo, si se quieren eliminar registros VEC, entiéndose esta acción como un borrado lógico de registros para que ya no estén activos ni disponibles para los usuarios, se sigue una estrategia similar a la anterior.

```go
//Creacion registro VEC
func (s *SmartContract) Set(ctx contractapi.TransactionContextInterface, idVEC string, descripcion string,
    vulnerabilidadResuelta bool, solucion string, entidades []string, dispositivo string, riesgo int8,
    autor string, fechaCreacion string, referencias string, keywords string, accion string, eliminado bool) error {

    registroVEC := RegistroVEC{
        IdVEC:                idVEC,
        Descripcion:          descripcion,
        VulnerabilidadResuelta: vulnerabilidadResuelta,
        Solucion:             solucion,
        Entidades:            entidades,
        Dispositivo:          dispositivo,
        Riesgo:               convertRiesgo(riesgo),
        Autor:                autor,
        FechaCreacion:        fechaCreacion,
        Referencias:          referencias,
        Keywords:             keywords,
        Accion:               accion,
        Eliminado:            eliminado,
    }

    registroVECBytes, err := json.Marshal(registroVEC)
    if err != nil {
        return fmt.Errorf("Marshal error: %s", err.Error())
    }

    err = ctx.GetStub().PutState(registroVEC.IdVEC, registroVECBytes)
    if err != nil {
        return fmt.Errorf("error al crear nuevo registro VEC: %s", err.Error())
    }

    return nil
}
```

Figura 1: Detalle de la función del *chaincode* para crear un nuevo registro VEC

Como se puede apreciar en la imagen de abajo (véase la figura 2), se está recibiendo como parámetro un valor *ID-VEC*, que es el identificador del registro VEC que se quiere eliminar. Lo que se hace en este método es primero consultar si el registro VEC existe realmente en la cadena de bloques y si es así, decodificar la información leida de la blockchain para convertirlo en un objeto que se puede tratar en el lenguaje *Golang*. Luego, se modifica la propiedad que dice si el registro VEC está activo o no, para marcar este como eliminado. Por último, se vuelve a codificar el registro VEC como *bytes* para incluirlo en la transacción que se realiza en la red de blockchain, al igual que hacíamos al crear uno nuevo.

```go
//Borrado logico de registros VEC
func (s *SmartContract) Delete(ctx contractapi.TransactionContextInterface, idVEC string) error {

    registroVECAsBytes, err := ctx.GetStub().GetState(idVEC)

    if err != nil {
        return fmt.Errorf("Error al obtener el registro VEC del estado actual de la blockchain: %s", err.Error())
    }

    if registroVECAsBytes == nil {
        return fmt.Errorf("%s no existe", idVEC)
    }

    registroVECBorrar := new(RegistroVEC)

    if json.Unmarshal(registroVECAsBytes, registroVECBorrar) != nil {
        return fmt.Errorf("Unmarshal error: %s", err.Error())
    }

    if registroVECBorrar.Eliminado {
        return fmt.Errorf("%s ya ha sido borrado", idVEC)
    } else {
        registroVECBorrar.Eliminado = true
        registroVECBorrar.Accion = Eliminacion

        registroVECDeletedAsBytes, err := json.Marshal(registroVECBorrar)
        if err != nil {
            return fmt.Errorf("Marshal error: %s", err.Error())
        }

        err = ctx.GetStub().PutState(registroVECBorrar.IdVEC, registroVECDeletedAsBytes)
        if err != nil {
            return fmt.Errorf("Error al borrar el registro VEC: %s", err.Error())
        }
    }

    return nil
}
```

Figura 2: Método para borrado lógico de registros VEC presente en el smart contract

De la misma manera, también se ha implementado un método para poder filtrar y obtener un listado de registros VEC deseados, para ello en este método se cuenta con una serie de parámetros por los que buscar. Los registros VEC se filtran utilizando el lenguaje de consultas de CouchDB y por eso se crea una consulta con los diferentes parámetros, tal y como se muestra en la figura 3. Gracias a otro método de la API de smart contracts de Hyperledger, llamado *GetQueryResult*, se ejecuta la consulta y obtenemos un iterador con los registros VEC filtrados. Es necesario iterar los registros VEC uno a uno para hacer la decodificación ya mencionada en los métodos anteriores y que la función pueda devolver la lista de registros VEC buscados, ya que los datos que se obtienen directamente de la blockchain tienen que ser transformados y cargados en tipos de datos interpretables por Golang.

```go
//Consulta: Registros VEC filtrados por unos parametros proporcionados
func (s *SmartContract) FiltrarRegistrosVECOrdenadosPorFecha(ctx contractapi.TransactionContextInterface,descripcion string,
    autor string, fechaCreacion string, entidad string, estado string, riesgo string) ([]QueryResult, error) {

    var filtrarPorEstado string
    if len(estado) > 0 {
        filtrarPorEstado = fmt.Sprintf("'vulnerabilidad-resuelta': %s,", estado)
    }

    var filtrarPorRiesgo string
    if len(riesgo) > 0 {
        filtrarPorRiesgo = fmt.Sprintf("'riesgo': %s,", riesgo)
    }

    consulta := parsearConsulta(
        `{
            'selector':
            {
                'eliminado':false,
                'descripcion': {
                    '$regex': '(?i)%s'
                },
                'autor': {
                    '$regex': '%s'
                },
                ` + filtrarPorEstado + filtrarPorRiesgo + `
                'fecha-creacion': {
                    '$regex': '%s'
                }
            },
            'sort': [{ 'fecha-creacion' : 'desc'}]
        }`)

    ordenarPorFecha := fmt.Sprintf(consulta, descripcion, autor, fechaCreacion, riesgo)
    fmt.Println("[CHAINCODE]: FiltrarRegistrosVECOrdenadosPorFecha - consulta: ", ordenarPorFecha)

    resultsIterator, err := ctx.GetStub().GetQueryResult(ordenarPorFecha)
    if err != nil {
        return nil, err
    }
    defer resultsIterator.Close()

    //Se devuelven los registros VEC decodificados resultado de la iteración
    return iterarRegistrosVEC(resultsIterator)
}
```

Figura 3: Operación de filtrado proporcionada en el *chaincode* de VECsg

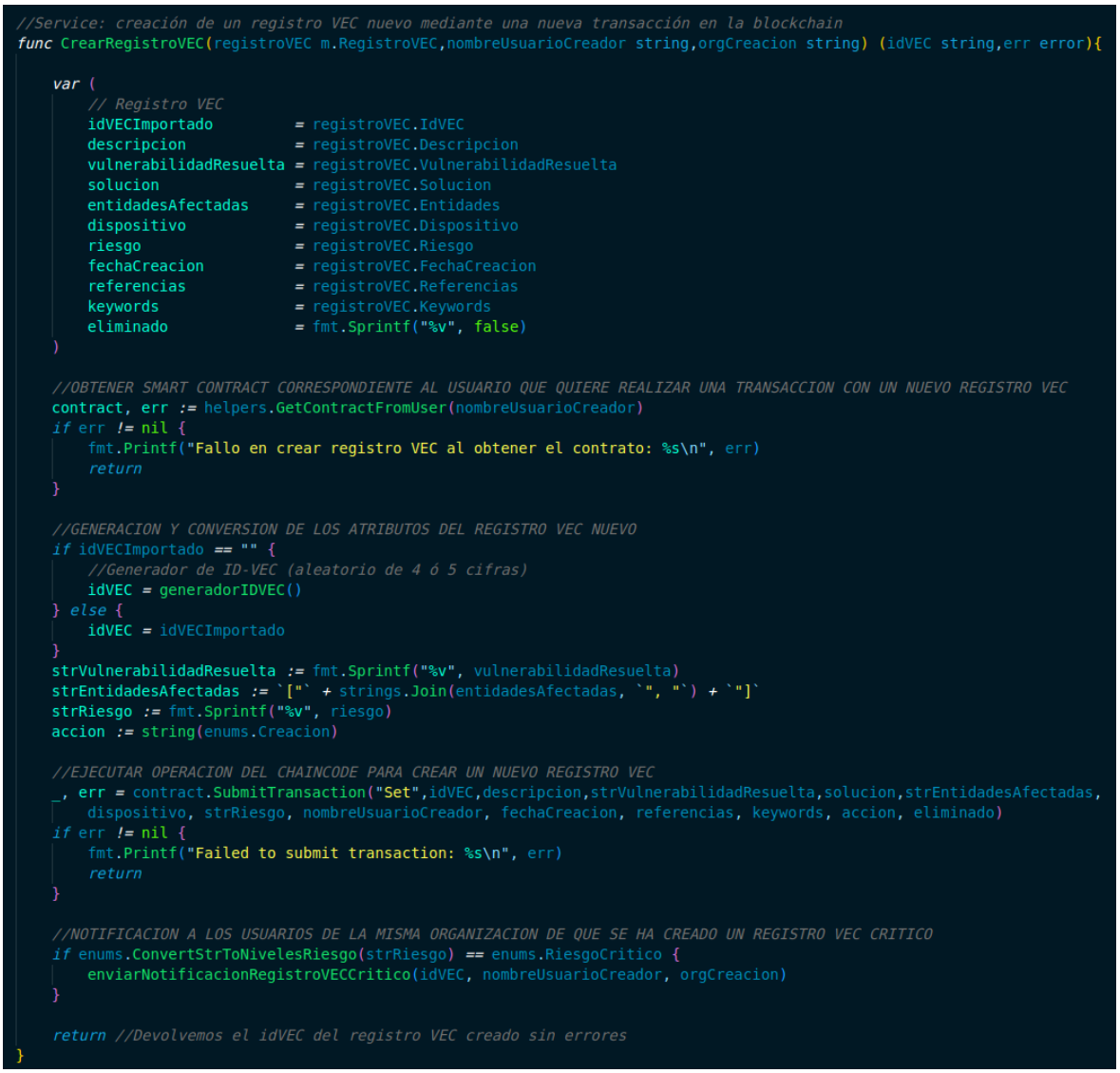
Como ya hemos mencionado anteriormente, el resto de funciones del *chaincode* siguen la misma estructura y sobre todo utilizan la API para contratos de Hyperledger, implementando fácilmente otras operaciones, como el recuento de registros VEC con diferentes características para recabar estadísticas del sistema, la edición de registros VEC, etc.

C.2. Detalle del código del servidor API construido sobre SDK Hyperledger Fabric

Ahora nos desplazamos del smart contract al servidor API creado a partir del SDK de Hyperledger Fabric para Golang. Desde él, usuarios debidamente autenticados y autorizados van a poder realizar llamadas a métodos del *chaincode* para ejecutar las operaciones sobre la blockchain que en él se definen.

Principalmente, el servidor de *Golang* se divide en dos partes clave, los controladores y los servicios. En los controladores se procesan los datos y se gestionan los permisos de quienes realizan peticiones a esta API desde el cliente web. Por otro lado, los servicios son funciones que se encargan de realizar las llamadas al *chaincode* y procesar los resultados que se obtienen de la red de blockchain, que en última instancia, pasan a los controladores para que estos los incluyan en la respuesta a la peticiones recibidas.

Si nos centramos en los servicios como el de la figura 4, podemos observar cómo ejectuar una trasancción en la blockchain gracias al método **SubmitTransaction** ofrecido por el SDK de Hyperledger. Así, se llama a la operación **Set** del contrato inteligente con todos los parametros necesarios para crear un nuevo registro. Antes de ejectuar el *chaincode*, se han tenido que preparar los argumentos que necesita para el método ejecutado y se tiene que obtener el *contract* a partir del usuario que debera coincidir etc. Posteriormente a la ejecución de la transacción, se comprueba si el registro VEC se catalogó de nivel crítico para notificar a los usuarios de que sus sistemas pueden llegar a ser comprometidos.



```go
//Service: creación de un registro VEC nuevo mediante una nueva transacción en la blockchain
func CrearRegistroVEC(registroVEC m.RegistroVEC,nombreUsuarioCreador string,orgCreacion string) (idVEC string,err error){

    var (
        // Registro VEC
        idVECImportado         = registroVEC.IdVEC
        descripcion            = registroVEC.Descripcion
        vulnerabilidadResuelta = registroVEC.VulnerabilidadResuelta
        solucion               = registroVEC.Solucion
        entidadesAfectadas     = registroVEC.Entidades
        dispositivo            = registroVEC.Dispositivo
        riesgo                 = registroVEC.Riesgo
        fechaCreacion          = registroVEC.FechaCreacion
        referencias            = registroVEC.Referencias
        keywords               = registroVEC.Keywords
        eliminado              = fmt.Sprintf("%v", false)
    )

    //OBTENER SMART CONTRACT CORRESPONDIENTE AL USUARIO QUE QUIERE REALIZAR UNA TRANSACCION CON UN NUEVO REGISTRO VEC
    contract, err := helpers.GetContractFromUser(nombreUsuarioCreador)
    if err != nil {
        fmt.Printf("Fallo en crear registro VEC al obtener el contrato: %s\n", err)
        return
    }

    //GENERACION Y CONVERSION DE LOS ATRIBUTOS DEL REGISTRO VEC NUEVO
    if idVECImportado == "" {
        //Generador de ID-VEC (aleatorio de 4 ó 5 cifras)
        idVEC = generadorIDVEC()
    } else {
        idVEC = idVECImportado
    }
    strVulnerabilidadResuelta := fmt.Sprintf("%v", vulnerabilidadResuelta)
    strEntidadesAfectadas := `["` + strings.Join(entidadesAfectadas, `", "`) + `"]`
    strRiesgo := fmt.Sprintf("%v", riesgo)
    accion := string(enums.Creacion)

    //EJECUTAR OPERACION DEL CHAINCODE PARA CREAR UN NUEVO REGISTRO VEC
    _, err = contract.SubmitTransaction("Set",idVEC,descripcion,strVulnerabilidadResuelta,solucion,strEntidadesAfectadas,
        dispositivo, strRiesgo, nombreUsuarioCreador, fechaCreacion, referencias, keywords, accion, eliminado)
    if err != nil {
        fmt.Printf("Failed to submit transaction: %s\n", err)
        return
    }

    //NOTIFICACION A LOS USUARIOS DE LA MISMA ORGANIZACION DE QUE SE HA CREADO UN REGISTRO VEC CRITICO
    if enums.ConvertStrToNivelesRiesgo(strRiesgo) == enums.RiesgoCritico {
        enviarNotificacionRegistroVECCritico(idVEC, nombreUsuarioCreador, orgCreacion)
    }

    return //Devolvemos el idVEC del registro VEC creado sin errores
}
```

Figura 4: Servicio para crear registro VEC

Si prestamos atención a otro de método de los servicios, como el de la figura 5, podemos apreciar como se gestionan los usuarios y las identidades en el servidor que actua como aplicación cliente de la blockchain de Hyperlegder. Al ser un servicio, la función *CrearUsuario* se llama desde un método controlador el cual interpreta el cuerpo de la petición que recibe del cliente web con todos los atributos del nuevo usuario. En primer lugar, es necesario crear el usuario en el servicio de autenticación, que como mencionábamos en secciones anteriores, se ha creado utilizando MongoD para mantener un almacén seguro de credenciales de usuarios (obsérvese el método *GenerarUsuario* del paquete *mongoDBAutenticacion* en la misma figura). Si se han guardado correctamente las credenciales para la autenticación del usuario, pasamos

a la fase de configurar su autorización en el sistema. Para ello tenemos que acceder el servicio de MSP (en inglés *Membership Service Provider*), del que ya hablamos en la memoria ofrecido por el SDK, para poder conectarnos a la Fabric CA de Hyperledger Fabric y poder realizar gestiones con los usuarios. Una vez se cargan los atributos del usuario, haciendo una petición con esos datos a la CA, se registra al usuario en el sistema y se genera el certificado que le dotará de identidad para poder conocer que permisos posee a la hora de acceder a la blockchain mediante el contrato inteligente. Una vez registrado, el usuario se une al sistema o se devuelve error en caso de que haya fallado alguno de los varios pasos realizados.

```go
//Service: crear usuario en el servidor auxiliar de autenticación y replicarlo con el SDK de Hyperledger Fabric
func CrearUsuario(nuevoUsuario m.Usuario) error {

    var (
        identificadorNuevoUsuario   = nuevoUsuario.Username, emailNuevoUsuario = nuevoUsuario.Email
        nombreApellidosNuevoUsuario = nuevoUsuario.NombreApellidos, password = nuevoUsuario.Password
        organizacionNuevoUsuario    = nuevoUsuario.Organizacion, rol = nuevoUsuario.Rol
    )

    //CREACION DE USUARIO EN EL SERVIDOR DE AUTENTICACION ACOPLADO
    err := mongoDBAutenticacion.GenerarUsuario(identificadorNuevoUsuario, password, organizacionNuevoUsuario)
    if err != nil {
        fmt.Printf("Error generando usuario en el servicio de autenticacion: %s\n", err);return err
    }

    //GENERACION DEL USUARIO EN LOS SERVICIOS DE AUTORIZACION DE HYPERLEDGER FABRIC
    msp, err := helpers.GetMSPFromOrganization(organizacionNuevoUsuario)
    if err != nil { return err }

    //CARGA DE LAS CARACTERISTICAS DEL USUARIO
    atributosUsuario := []client.Attribute{{Name: gc.ORGANIZACION, Value: organizacionNuevoUsuario},
        {Name: gc.EMAIL, Value: emailNuevoUsuario},{Name: gc.FAVORITOS, Value: ""},
        {Name: gc.CONFIRMACIONES, Value: gc.SIN_RESTRICCION_CONFIRMACIONES},
        {Name: gc.NOMBRE_APELLIDOS, Value: nombreApellidosNuevoUsuario}}
    atributosUsuario = establecerPermisosUsuario(atributosUsuario, rol)

    //FABRIC CA REGISTRA Y UNE AL USUARIO AL SISTEMA
    secret, err := msp.Register(&client.RegistrationRequest{
        Name: identificadorNuevoUsuario,
        Type: "client", Attributes: atributosUsuario,
        CAName: fmt.Sprintf("ca.%s.acme.com", strings.ToLower(organizacionNuevoUsuario)),
    })
    if err != nil {
        fmt.Printf("error al registrar al usuario: %s", err)
        auths.RevertirCambiosAutenticacion(identificadorNuevoUsuario)
        if strings.Contains(err.Error(), "is already registered") {
            return errors.New(gc.ERROR_USUARIO_YA_EXISTE)
        } else {return err}
    }

    err = msp.Enroll(identificadorNuevoUsuario, client.WithSecret(secret))
    if err != nil {
        fmt.Printf("error al unir al usuario al sistema: %s\n", err);return err
    }
    return nil
}
```

Figura 5: Función dentro de los servicios para crear usuarios

Las funciones controladoras del servidor de la API en VECsg son las que se sitúan entre las rutas y los servicios. Por ello gestionan las peticiones para tomar de ellas los datos y controlan los permisos que tienen los usuarios. Por ejemplo, para comprobar si un usuario tiene el rol con la suficiente autoridad para realizar transacciones que originen nuevos registros VEC en la cadena de bloques.

```go
//Controller: crear registro VEC llamando al service correspondiente con los parametros adecuados
func CrearRegistroVEC(c *gin.Context) {

    var registroVEC m.RegistroVEC

    //GESTION DEL TOKEN JWT RECIBIDO EN LA PETICIÓN PARA OBTENER LAS CLAIMS DEL USUARIO
    jwtKey, err := h.GetPrivateKey()
    if err != nil {
        fmt.Println("Se ha producido un error al obtener llave privada: " + err.Error())
        u.Respond(c.Writer, u.Header(http.StatusInternalServerError)); return
    }

    claims := utils.GetClaimsFromAuthHeader(c, jwtKey)
    if claims == nil {
        fmt.Println("[ERROR]: Se ha producido un error decodificando las claims"); return
    }

    errorBadRequest := json.NewDecoder(c.Request.Body).Decode(&registroVEC)
    if errorBadRequest != nil {
        fmt.Println("Error Bad Request: " + errorBadRequest.Error())
        //HTML error code 400: "Error petición incorrecta"
        u.Respond(c.Writer, u.Header(http.StatusBadRequest)); return
    }

    //COMPROBACION DE QUE EL USUARIO TIENE LOS PERMISOS SUFICIENTES+
    //+ PARA GENERAR TRANSACCIONES (GESTOR O SUPERIOR)
    usuarioAutorizado := comprobarPermisoUsuario(claims.Permisos, gc.ESCRITURA)
    if !usuarioAutorizado {
        //HTML error code 403: "Usuario no autorizado para hacer esta operacion"
        u.Respond(c.Writer, u.Header(http.StatusForbidden))
        fmt.Println("[ERROR]: El usuario no está autorizado a crear registos VEC"); return
    }

    //SE HACE LA LLAMADA AL SERVICIO DE CREAR REGISTRO VEC CON LOS VALORES+
    //+ CORRECTOS UNA VEZ REALIZADAS LAS COMPROBACIONES NECESARIAS
    idVEC, err := v1s.CrearRegistroVEC(registroVEC, claims.Username, claims.OrgName)
    if err != nil {
        //HTML error code 500: "Error inesperado del servidor"
        u.Respond(c.Writer, u.Header(http.StatusInternalServerError))
        fmt.Println("[ERROR]: Error inesperado del servidor al crear registos VEC"); return
    }
    bytesIdVEC, err := json.Marshal(RespuestaCreacion{IdVEC: idVEC})
    if err != nil {
        //HTML error code 500: "Error inesperado del servidor"
        u.Respond(c.Writer, u.Header(http.StatusInternalServerError)); return
    }
    //SE DEVUELVE COMO RESULTADO DE LA CREACIÓN EL ID-VEC QUE SE CREA
    u.Respond(c.Writer, u.Data(200, string(bytesIdVEC)))
}
```

Figura 6: Método controlador para crear registros VEC

El detalle del controlador que se utiliza para manejar en primera instancia las solicitudes para crear un nuevo registro VEC, se puede encontrar en la figura 6. En esta función se puede observar como se tratan valores que vienen en la petición como el token JWT, el cual se valida con la clave privada del servidor y se comprueba su caducidad para poder seguir decodificando los siguiente valores que se han enviado a través de HTTPS (los atributos del nuevo registro VEC). El último paso antes de llamar al servicio correspodiente es comprobar que el usuario tiene los permisos obligatorios, es decir, tiene un rol de administrador o gestor, para poder escribir y no solo ejecutar consultas con el contrato inteligente. Y al igual que en el resto de métodos, la utilidad *Marshal* nos permite tratar los datos para enviarlos en el formato correcto.

Otra función interesante de los controladores es la que se utiliza para hacer login en la plataforma VECsg (figura 8). En ella vemos como tras cargar las credenciales que ha enviado el usuario para iniciar sesión, se pasan esos datos al servicio adecuado, pero en este caso es necesario que el resultado de esa llamada se depure rigurosamente para informar al usuario que quería conectarse en el caso de que ocurra algún error. Si todo sale según lo esperado, se genera un token JWT temporal que abre una especie de sesión durante la cual, el usuario puede hacer llamadas a esta API-REST, y a través del token se pueda controlar su identidad y autoridad en el sistema.

```go
//Controller: login de usuario comprobando sus credenciales e iniciar una sesion con el perfil correcto
func LoginUsuario(c *gin.Context) {

    //CARGA DE LAS CREDENCIALES RECIBIDAS EN LA PETICIÓN
    var credentials m.Credentials

    error := json.NewDecoder(c.Request.Body).Decode(&credentials)
    if error != nil {
        fmt.Println("Error Bad Request: " + error.Error())
        u.Respond(c.Writer, u.Header(http.StatusBadRequest))
        return
    }

    //LLAMADA AL SERVICIO DE LOGIN PARA VERIFICAR CREDENCIALES CORRECTAS Y OBTENER LOS DATOS DEL USUARIO
    permisos, organizacion, error := v1s.LoginUsuario(credentials.Username, credentials.Password)

    if error != nil {
        if error.Error() == gc.ERROR_LOGIN_INCORRECTO {
            //HTML error code 401: "Usuario o contraseña incorrectos"
            u.Respond(c.Writer, u.Header(http.StatusUnauthorized))
            return
        }
        if error.Error() == gc.ERROR_FALTA_CONFIRMACION {
            //HTML error code 425: "El usuario no esta todavia autorizado a iniciar sesion"
            u.Respond(c.Writer, u.Header(http.StatusTooEarly))
            return
        }
        if error.Error() == gc.ERROR_USUARIO_DESACTIVADO {
            //HTML error code 403: "El usuario ha sido desactivado por un administrador"
            u.Respond(c.Writer, u.Header(http.StatusForbidden))
            return
        }
        //HTML error code 500: "Error inesperado del servidor"
        u.Respond(c.Writer, u.Header(http.StatusInternalServerError))
        return
    }

    //GENERACION TOKEN JWT DEL USUARIO QUE HA INICIADO SESIÓN CORRECTAMENTE
    expirationTime := time.Now().Add(time.Minute * 60)

    claims := &m.Claims{Username: credentials.Username,OrgName:  organizacion,
        StandardClaims: jwt.StandardClaims{ExpiresAt: expirationTime.Unix()},Permisos: permisos}
    token := jwt.NewWithClaims(jwt.SigningMethodHS256, claims)
    jwtKey, err := helpers.GetPrivateKey()
    if err != nil {
        fmt.Println("[ERROR]: Se ha producido un error al obtener el secreto o llave privada: " + err.Error())
        u.Respond(c.Writer, u.Header(http.StatusInternalServerError))
        return
    }
    tokenString, error := token.SignedString(jwtKey)
    if error != nil {
        u.Respond(c.Writer, u.Header(http.StatusInternalServerError))
        return
    }

    //ENVIO DEL TOKEN JWT GENERADO PARA PERMITIR AL USUARIO HACER LLAMADAS A LA API
    response := u.Message(200, "Sesión iniciada con éxito.")
    response["tokenJWT"] = tokenString
    u.Respond(c.Writer, response)
}
```

Figura 7: Controlador login usuario

C.3. Detalle del código del cliente web

En el cliente web el código que encontramos es muy sencillo, pues se trata de funciones controladoras que solo tienen que encargarse de llamar hacer peticiones al servidor API REST, incluyendo los parámetros que incluya el usuario y el token JWT y recibir la respuesta para pintarla en pantalla. Haciendo siempre la obligada gestión de errores. A continuación se puede apreciar el detalle de la implementación del método de controlador para obtener un registro VEC y todos su atributos por ID-VEC, escrito en PHP:

```php
class RegistroVECController extends Controller
{
    //REGISTRO VEC POR ID
    public function obtenerRegistroVEC($idVEC){

        $tokenJWT = request()->session()->get("tokenJWT");

        try{
            $responseRegistroVEC = Http::withToken($tokenJWT)->get(env('URL_API_REST_BLOCKCHAIN').env('API_VERSION').'registro-vec/'.$idVEC);
        } catch (\Exception $e)
        {
            Log::error("Error en el método controlador: obtenerRegistroVEC: ".$e->getMessage());
            request()->session()->remove('tokenJWT');
            return view('errors/errorDesconexion', ["sesionIniciada" => true]);
        };
        //BODY: STATUS
        $status = json_decode($responseRegistroVEC->body())->status;
        switch( $status ){
            case 204:
                return redirect('/home')->with('error', 'Registro VEC no encontrado.');
            case 403:
                return redirect('/home')->with('error', 'No se tienen los permisos necesarios para realizar esta operación.');
            case 500:
                return redirect('/home')->with('error', 'Error inesperado del servidor.');
        }

        //BODY: DATA
        $dataRegistroVEC = json_decode($responseRegistroVEC['data'], true);
        $registroVECAdaptado = adaptJSON2Model($dataRegistroVEC);
        $registroVEC = new RegistroVEC($registroVECAdaptado);

        return response()->view('registroVEC/detalle', $registroVEC);
    }
```

Figura 8: Controlador para obtener registro VEC por ID-VEC

En resumen, lo que se intenta es que todas la funciones sigan las misma estructura respectivamente en el contrato inteligente, los servicios, los controladores del servidor e incluso en el frontend. Esto es así, para asegurar unas buenas prácticas que permitan que los flujos de datos se mantengan ordenados y controlados y sea fácil depurar el gran número de funciones implementadas, estableciéndose de este modo una guía o convenio de cómo implementar código en cada una de las partes del proyecto. Así, sí se quieren añadir más funcionalidades es posible mantener el código limpio y mantenible en el tiempo.

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga

E.T.S. DE INGENIERÍA INFORMÁTICA