Modelo de autenticación mediante blockchain Ethereum para mitigar las brechas de seguridad en las cámaras IoT del sector masivo

FRAY ESNEIDER OSPINA RODRIGUEZ

Instituto Tecnológico Metropolitano

Facultad de Ingeniería

Medellín, Colombia

Año

2024

Modelo de autenticación mediante blockchain Ethereum para mitigar las brechas de seguridad en las cámaras IoT del sector masivo

FRAY ESNEIDER OSPINA RODRIGUEZ

Tesis o trabajo de investigación presentada(o) como requisito parcial para optar al título de:

Magister en Seguridad informática

Director (a):

Juliver Gil Herrera

Línea de Investigación:

Ciencias Computacionales

Instituto Tecnológico Metropolitano

Facultad de Ingeniería

Medellín, Colombia

Año

2024

Dedicatoria

Este trabajo está dedicado especialmente a mi familia, por el apoyo, el tiempo, la comprensión, por los espacios que dejamos de compartir y aquellos momentos que tocó dejar de lado, este esfuerzo va dedicado a ellos, son el motor de mi vida y los que me alientan a ser más fuerte cada día. Es un sacrificio por un mejor mañana para ellos y para mí. Dedicado también especialmente a mi novia por su tiempo y paciencia, no me logró ver graduar, partió de este mundo demasiado pronto.

Agradecimientos

El agradecimiento es para los asesores, profesores de las asignaturas y al director de la maestría ya que sin el apoyo, orientación, paciencia y aporte no hubiera logrado sacar esto adelante. Hubo momentos grises que con la ayuda de todos siempre hubo esperanza y optimismo sobre el trabajo, adicional agradecido con todos los compañeros de clase por su aporte directo e indirecto en la parte de conocimiento y profesión.

Resumen

El trabajo de grado se enfoca en el diseño y aplicación de un modelo de autenticación utilizando la tecnología blockchain de Ethereum para mejorar la seguridad de los dispositivos IoT hogares como lo son las cámaras de seguridad, teniendo en cuanta que son dispositivos inteligentes con conexión a internet que permiten un control y monitoreo del hogar. Día a día estos dispositivos van en un auge de crecimiento exponencial ampliando el monitoreo y visualización del hogar, compartiendo más y más conexiones sobre el internet, en contraste, los ciberdelincuentes están constantemente ocupados, y cada día surgen nuevas amenazas. Esto significa que una cámara de seguridad adquirida hace cinco años o una comprada apenas hace seis meses podrían presentar una brecha de seguridad. Esto conlleva a que incluso aquellos hackers con poca experiencia puedan identificar exploits en la web y aprovecharlos para infiltrarse en tu red.

Mediante el desarrollo del modelo se logró realizar una investigación sobre tres dispositivos de fabricantes diferentes, seleccionando uno de los tres para la prueba de concepto, bajo la ejecución de un objetivo general definido por un modelo de autenticación mediante blockchain Ethereum, y cuatro objetivos específicos orientados a la caracterización de información, identificación de amenazas, ejecución de configuraciones y finalmente validación sobre un escenario de pruebas sobre el dispositivo seleccionado.

Palabras clave: Autenticación, blockchain, cámaras de seguridad, Ethereum. IoT, vulnerabilidad, SDK, CVSS, CVE.

Abstract

The thesis focuses on the design and implementation of an authentication model using Ethereum blockchain technology to enhance the security of IoT devices in households, such as security cameras. It considers these devices as smart gadgets with internet connectivity, allowing for home control and monitoring. Day by day, these devices experience exponential growth, expanding home monitoring and visualization by sharing more and more connections over the internet. In contrast, cybercriminals are consistently busy, and new threats emerge daily. This implies that a security camera purchased five years ago, or one bought just six months ago could exhibit a well-known vulnerability. Consequently, even hackers with limited experience can identify exploits on the web and utilize them to infiltrate your network.

Through the development of the methodology, an investigation was conducted on three devices from different manufacturers, with one selected for the proof of concept. This was done under the pursuit of a general objective defined by an-Ethereum blockchain-based authentication model, along with four specific objectives focused on information characterization, threat identification, configuration execution, and ultimately validation in a test scenario using the selected device.

Keywords: Authentication, blockchain, security cameras, Ethereum. IoT, vulnerability, SDK, CVSS, CVE

Contenido

Lista de figuras

Lista de tablas

Pág.

Lista de Abreviaturas

Abreviatura	termino
IoT	Internet de las cosas
HK	Hikvision
SDK	Kit Desarrollo de Software
LAN	Red de Área Local
WAN	Red de Área Extensa
CCTV	Circuito Cerrado de Televisión
IEEE	Instituto de Ingenieros Eléctricos y Electrónicos
PTZ	Zoom de inclinación panorámica
SOL	Solidity
NVD	Base de datos nacional de vulnerabilidad
RUT	Registro Único Tributario
CVSS	Sistema de puntuación de vulnerabilidad común
CVE	Vulnerabilidades y exposiciones comunes

Abbreviations list

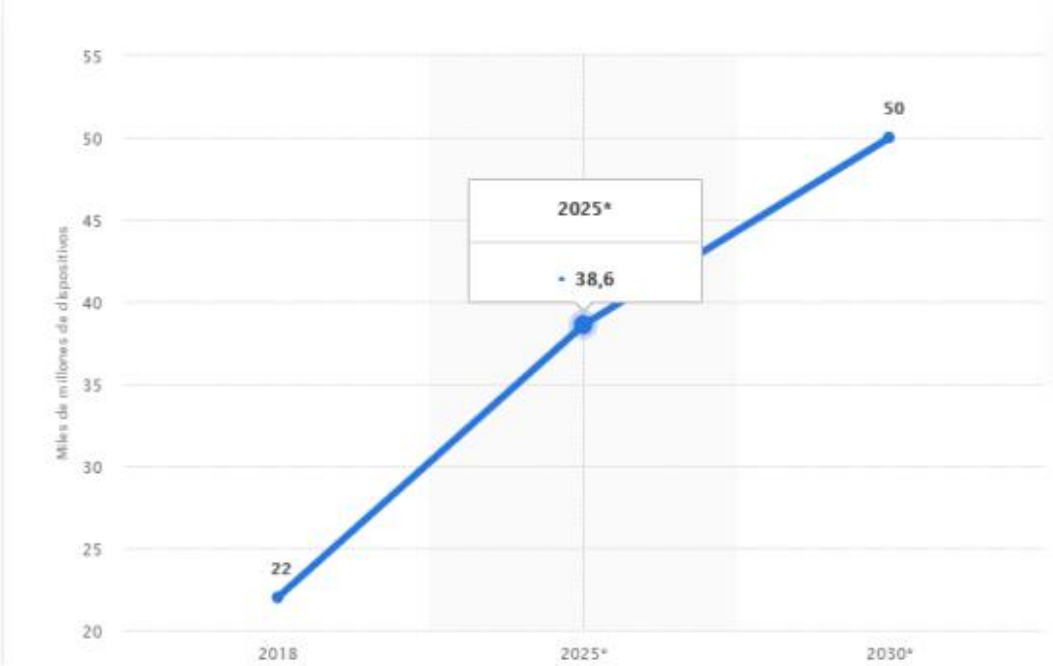
Abbreviation	Term
IoT	Internet of Things
HK	Hikvision
SDK	Software Development Kit
LAN	Local Area Network
WAN	Wide Area Network
CCTV	Closed Circuit Television
IEEE	Institute of Electrical and Electronics Engineers.
PTZ	Pan Tilt Zoom
SOL	Solidity
NVD	National Vulnerability Database
RUT	Single Tax Registry
CVSS	Common Vulnerability Scoring System
CVE	Common Vulnerabilities and Exposures

Introducción

El término "Internet de las cosas" (IoT) proviene de la conexión masiva de dispositivos automatizados, una idea relacionada por Kevin Ashton [1]. Esta interconexión implica un alejamiento de los sistemas manuales, permitiendo una mayor interacción con las personas, la razón detrás de esto radica en el control ejercido por los usuarios sobre las máquinas o dispositivos conectados a Internet, Sin embargo, se han identificado debilidades en términos de precisión y tiempos de respuesta. Aunque estas limitaciones están experimentando cambios, el objetivo del IoT es facilitar la rápida interconexión de nuevos dispositivos, especialmente en entornos domésticos, para generar comportamientos automatizados en respuesta a eventos específicos. En este contexto, los dispositivos son los actores principales de esta tecnología, y sus aplicaciones se extienden a la industria, el medio ambiente y la sociedad. [2].

Las cámaras de Internet de las Cosas (IoT) se han vuelto extendidas en los hogares modernos, ofreciendo una variedad de funciones, desde seguridad hasta monitoreo remoto. Sin embargo, con su proliferación, surgen preocupaciones de seguridad significativas, especialmente en lo que respecta a la autenticación. La autenticación débil o inadecuada puede dejar estas cámaras vulnerables a intrusiones maliciosas, comprometiendo la privacidad y la seguridad de los residentes. La siguiente imagen, representa el aumento de dispositivos IoT actual y una proyección hacia el 2030, donde cada día son mucho más los dispositivos dentro del entorno hogar para múltiples funciones, entre ellas el monitoreo de objetos, personas, mercancía o la misma vivienda a través de cámaras de vigilancia IoT hogares. La grafica ilustra tres puntos, uno en la parte inferior izquierda representando la cantidad de dispositivos en 2018 correspondiente a 22 mil millones de dispositivos conectados a internet, uno central proyectando un incremento de 38,6 mil millones de dispositivos conectados a internet en 2025 y un tercero en la parte superior derecha que proyecta para el 2030 un incremento de 50 mil millones de dispositivos conectados a internet.

Fig. 1: Aumento de dispositivos IoT conectados en internet.



Nota. La grafica ilustra el crecimiento de los dispositivos IoT conectados cada vez más en internet. Fuente: [3].

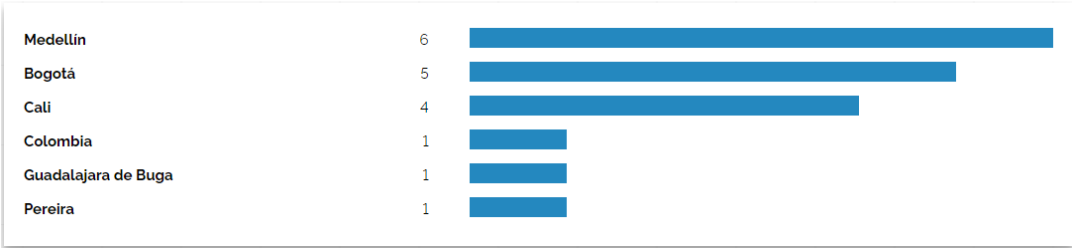
Al buscar cámaras web IP por algunas marcas, es fácil obtener las contraseñas de fábrica con las que vienen preconfiguradas. Estas situaciones son más comunes de lo que se podría pensar, ya que no es raro encontrar informes periódicos de extorsiones derivadas de ataques en los cuales los perpetradores logran acceder a las cámaras para obtener imágenes o información sensible de las víctimas [4]. El vector de ataque en estos casos no se limita únicamente a las contraseñas; con frecuencia, el ataque va acompañado de una configuración deficiente de autenticación o nula en la red doméstica.

De la misma forma las brechas de seguridad y vulnerabilidad han ido creciendo de acuerdo con el incremento de dispositivos IoT, especialmente en las cámaras de vigilancia en hogares, estas han ido creciendo en el mercado en todas las marcas entre ellas (TP-link, D-link, Hikvision, Dahua, V-supra, Ezviz, Hilook, Tenda, y otras marcas genéricas chinas), debido al consumo y auge residencial, lo que siendo más los casos de acoso cibernético, las amenazas mediante el audio de las mismas cámaras, los chantajes para pedir dinero por medio del robo de información sensible, los saqueos de casas y apartamentos en ausencia de personas debido al acceso y monitoreo que logran tener los ciberdelincuentes.

En este contexto, surge la necesidad de implementar modelos de autenticación más robustos y seguros que puedan hacer frente a las crecientes amenazas cibernéticas. Una solución prometedora es la autenticación mediante blockchain Ethereum, que ofrece una serie de características que pueden abordar eficazmente los desafíos de seguridad en las cámaras IoT para hogares.

En las dos imágenes siguientes se puede apreciar las ciudades y puertos más vulnerables en los dispositivos IoT. En la fig. 2 (Ciudades más vulnerables de Colombia), se identificaron registros sobre shodan.io con el filtro IPcamera country:CO el cual registró la cantidad de cámaras vulnerables o con anomalías por ciudades.

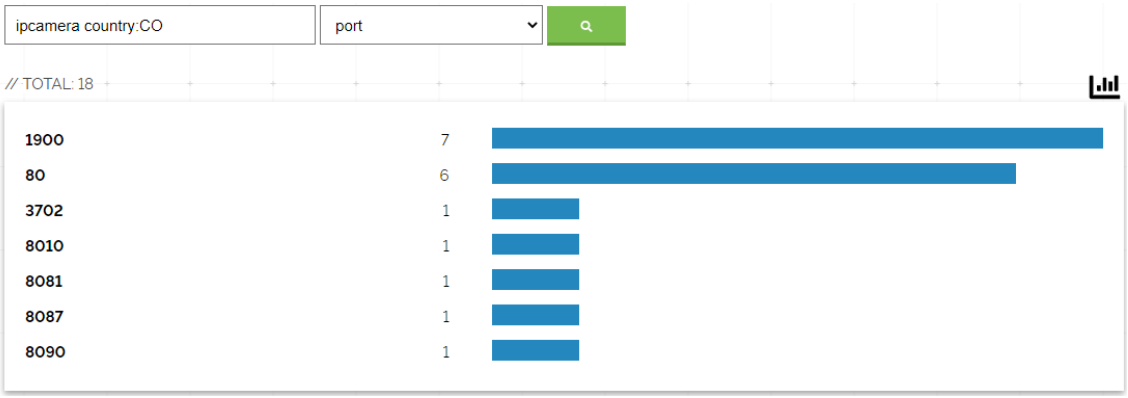
Fig. 2: Ciudades más vulnerables de Colombia.



Nota. Ciudades más vulnerables por ciberdelincuentes en Colombia. Fuente: [5].

En la fig. 3 (Puertos más vulnerados de las cámaras en Colombia.), bajo la misma plataforma shodan.io, y con el filtro IPcamera conuntry:CO filtrando por puerto, se identificaron el número de puerto y la cantidad de puertos vulnerables, expuestos, o con novedades en Colombia.

Fig. 3: Puertos más vulnerados de las cámaras en Colombia.



Nota. Puertos de cámaras de vigilancia más vulnerables por ciberdelincuentes en Colombia. Fuente: [5].

Se plantea la implementación de un modelo de autenticación basado en blockchain Ethereum que pueda mejorar significativamente la seguridad de las cámaras IoT en hogares al abordar las deficiencias de los métodos de autenticación tradicionales. Se espera que esta solución proporcione una autenticación más segura y resistente a los ataques cibernéticos al utilizar claves criptográficas, firmas digitales y un registro inmutable de transacciones. Además, se espera que la descentralización y la resistencia a la manipulación inherentes a la tecnología blockchain brinden una capa adicional de protección contra intrusiones maliciosas. En conjunto, se anticipa que la implementación de este modelo de autenticación contribuirá a mitigar las vulnerabilidades de seguridad en las cámaras IoT para hogares y garantizar la privacidad y la seguridad de los usuarios.

Para la realización de esta investigación se cuenta con el siguiente objetivo general y objetivos específicos:

Objetivo General

Diseñar un modelo de autenticación mediante blockchain Ethereum para mejorar la seguridad en las cámaras IoT enfocado en hogares, con el fin de minimizar las brechas de seguridad en acceso, monitoreo, interceptación de contenido e información.

Objetivos Específicos

- Caracterizar información de las diferentes autenticaciones y protocolos de las cámaras de vigilancia.
- Identificar amenazas y riesgos de las cámaras de vigilancia en hogares.
- Ejecutar las configuraciones, códigos de autenticación basado en las amenazas y riesgos identificados.
- Validar el modelo aplicado en un escenario de pruebas para la configuración de la autenticación a través de blockchain para las cámaras de vigilancia.

1. Marco Teórico, y Estado del Arte

Este apartado cuanta con el marco teórico y estado del arte que abarcan los estándares de comunicaciones inalámbricas 802.11, y alámbricas 802.3, conceptos de los diferentes tipos de cámaras de seguridad, protocolos de autenticación de seguridad web. El marco teórico contiene transcendencia de las redes de comunicación respectivamente, dando un acercamiento sobre el inicio de la interconexión y el avance con la que cuenta actualmente en los diferentes escenarios, veremos como Colombia en medio de estas tecnologías emergentes las va adaptando y evolucionando.

1.1 Marco teórico

Las redes de datos han estado en uso durante mucho tiempo, surgiendo inicialmente para satisfacer la necesidad de compartir recursos y servicios a través de sistemas informáticos con el objetivo de procesar la información de manera más eficiente. Esto se logra mediante la implementación de protocolos establecidos como el modelo TCP/IP y el modelo OSI. A partir de estas consideraciones, se establece que cuando las computadoras comparten un espacio local de extensión limitada, se denomina LAN (Red de Área Local), que es una de las más utilizadas en la actualidad [6], Otros tipos de redes que han ganado relevancia y forman parte del funcionamiento de diversas redes en un área geográfica un poco más amplia, sin exceder una extensión mayor, se conocen como PAN (Red de Área Personal). Cuando la extensión es mayor y conecta varias áreas geográficas, se denomina WAN (Red de Área Extensa), como es el caso de Internet [7].

Viendo como nacen las redes de datos vamos ahora a ver como nacen los dispositivos de cámaras de vigilancia, su inicio impacto una revolución de las cámaras, fueron inicialmente en el ámbito cinematográfico, sus primeras apariciones ocurrieron en el ámbito cinematográfico en 1880. No fue sino hasta 1942 que las cámaras se utilizaron como elementos de seguridad en sistemas de circuito cerrado de televisión (CCTV). En sus inicios, estos sistemas consistían únicamente en cámaras en blanco y negro conectadas a monitores, diseñados originalmente para supervisar el lanzamiento de cohetes. Este evento marca el comienzo de la videovigilancia, que a lo largo de la historia ha experimentado un crecimiento significativo. [8].

A medida que avanzan la electrónica y la tecnología en sistemas de cámaras de vigilancia, aumenta la demanda y la cantidad de usos, siendo un lujo en su inicio, terminando con una necesidad, de la misma forma generan un incremento en los dispositivos IoT conectados a la red, lo que ha atraído una atención considerable para mejorar la seguridad bajo este incremento, ya que se aumenta en equipos e interconexiones pero no en seguridad, al estar estos equipos interconectados y accesibles desde internet, generan un ambiente hogar desprotegido, expuestos a delitos, robo de información, suplantación, robo de contenido y todo lo relacionado a ataques informáticos [9].

La Internet de las cosas (IoT) está posibilitando la inclusión de nuevas variables previamente no consideradas, como la temperatura, la luz y la humedad. Esta capacidad de expansión se extiende incluso a ubicaciones antes percibidas como inaccesibles, generando un volumen significativo de datos dispersos en la red. Estos datos provienen de la creciente interconexión de millones de dispositivos, que incluyen módulos con sensores para validar comportamientos específicos. Los dispositivos tienen funciones de control, ubicación y emisión de alertas, y las casas inteligentes cuentan con una variedad de dispositivos y sensores que colaboran en la toma de decisiones para ejecutar operaciones programadas, conocido como automatización. Este desarrollo ha dado lugar a la creación de numerosas aplicaciones asociadas a esta red de dispositivos interconectados.

De manera gradual, ha ocurrido un aumento constante en la proliferación de dispositivos, lo cual impulsa la expansión del Internet de las cosas (IoT). Este fenómeno representa la primera revolución de Internet, transformando la naturaleza de la red para hacerla más sensorial. Esta evolución busca atraer la atención de los consumidores a través de experiencias basadas en tecnologías inteligentes, que incluyen aspectos relacionados con la seguridad y la privacidad [10]. Este crecimiento se fundamenta en la comunicación entre dispositivos, permitiéndoles determinar comportamientos y ejercer control sin requerir la intervención directa del usuario. Se busca alcanzar lugares previamente inexplorados mediante esta comunicación. No obstante, este progreso aún requiere parámetros claros y

estandarizados en términos de seguridad, privacidad y arquitectura. La IEEE es una de las organizaciones que trabaja activamente para garantizar que los paquetes IP sean comprensibles en diversos tipos de redes, evitando interrupciones en esta comunicación [11].

En la imagen a continuación se ilustra dos círculos encerrando la nube publica de conexión a internet, donde en el primer círculo interno representa sensores conectados a internet y el segundo circulo exterior representa otro tipo de tecnología incorporada a la red IoT, mediante diferentes dispositivos y con fines más enfatizados en hogares, como bombillos, smartphone, relojes inteligentes entre otros.

Fig. 4: Crecimiento de internet de las cosas IoT

Nota. Dispositivos IoT interconectados entre ellos mediante una arquitectura de red. Fuente: [12]

Sabemos que la innovación es bastante importante, busca volver un mundo más digital y automatizado, pero de la mano va el riesgo y las vulnerabilidades como lo indica [13], a medida que aumenta la conectividad en todas las industrias, los actores de amenazas se dirigen a la superficie de red en expansión y cada vez más interconectada de las empresas en todo el mundo; si bien los entornos de IoT se han vuelto cada vez más complejos, las

soluciones de seguridad de TI se han quedado atrás, con visibilidad y control limitados sobre los dispositivos de IoT y sus riesgos asociados.

1.1.1 Estándar 802.11 b/g/n

Uno de los estándares que se empleará es el 802.11 b/g/n, donde 802.11 representa la designación del IEEE para las redes inalámbricas. En este contexto, ha habido dos variaciones con respecto al estándar inalámbrico 802.11 inicial, que ofrecían velocidades de transmisión de 1 o 2 Mbps y utilizaban la misma frecuencia de radio de 2.4 GHz. La distinción entre ellos radicaba en la forma en que los datos viajaban a través de los medios de radiofrecuencia (RF), ya que uno utilizaba FHSS y el otro DSSS [14]. Es importante señalar que los estándares originales 802.11 resultan ser demasiado lentos para las exigencias actuales de las redes y ya no se implementan. Cada uno de estos estándares presenta un comportamiento específico en cuanto a velocidad de transmisión, dependiendo del dispositivo y su hardware, ya que pueden variar en términos de velocidad, alcance de transmisión y frecuencia utilizada. Sin embargo, en términos de implementación, son similares. Todos los estándares pueden utilizar una infraestructura o un diseño de red inalámbrica, incluso admiten una arquitectura ad hoc (redes temporales sin necesidad de un enrutador). Además, todos pueden hacer uso de los mismos protocolos de seguridad [15].

Cuando se aborda la seguridad inalámbrica en Wi-Fi, existen solo algunas opciones disponibles para el público en general, especialmente al configurar una red doméstica inalámbrica. Los principales protocolos de seguridad en uso actualmente son WEP, WPA y WPA2, junto con los algoritmos TKIP y AES [16]. En esencia, la evolución de los protocolos de seguridad inalámbrica ha sido gradual debido a los estándares y la necesidad de mantener la estabilidad. Los mencionados anteriormente ya han demostrado vulnerabilidades, y afortunadamente, en 2018 se presentó WPA3 como un nuevo protocolo de seguridad estándar, que se consolidó en 2020. Sin embargo, la adopción generalizada puede llevar tiempo. La mayoría de los hogares y empresas aún utilizan WPA2, ya que el hardware compatible con WPA3 puede resultar costoso. Con WPA3, el cifrado entre el

dispositivo del usuario y la red es específico e individualizado, y los usuarios incluso pueden prescindir de introducir una contraseña. [17].

La tabla 1. (Detalles de generaciones wifi bajo 802.11) describe los parámetros de operación del protocolo wifi 802.11, relacionando por cada generación de wifi, la tecnología, le frecuencia de operación y la velocidad de datos.

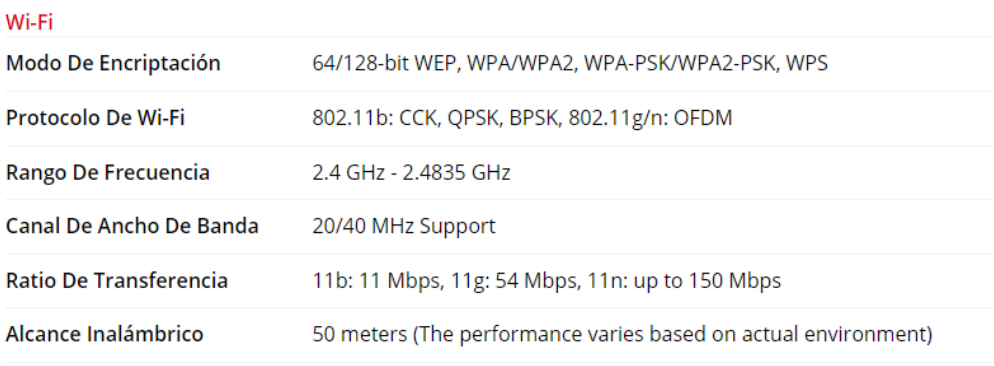
Tabla 1. Detalles de generaciones wifi bajo 802.11

GENERACIÓN	TECNOLOGÍA	FRECUENCIA	VELOCIDAD
-	802.11b	2.4 GHz	1 - 11 Mbps
-	802.11a	5 GHz	Up to 54 Mbps
-	802.11g	2.4 GHz	Up to 54 Mbps
Wi-Fi 4	802.11n	2.4 and 5 GHz	Up to 600 Mbps
Wi-Fi 5	802.11ac	2.4 and 5 GHz	Up to 3.5 Gbps
Wi-Fi 6	802.11ax	2.4 and 5 GHz	Up to 9.6 Gbps

Nota. Especificaciones técnicas de operación del protocolo 802.11. Fuente: [18]

La imagen siguiente Fig. 5 (Parámetros del protocolo wifi cámara IoT) describe los parámetros del dispositivo IoT cámara de seguridad sobre el cual trabaja en 802.11, adicional relaciona el modo de encriptación, rango de frecuencia relacionada al protocolo y el canal de ancho de banda.

Fig. 5: Parámetros del protocolo wifi cámara IoT



Nota. Especificaciones técnicas de red cámara IoT en protocolo 802.11. Fuente: [19]

1.1.2 Estándar 802.3

En comparación con el protocolo anterior este se centra en conexiones LAN 802.3, redes de área local en estrella o en anillo, no maneja protocolo de autenticación, maneja un límite de distancia dependiendo del medio, siendo este al inicio por cable coaxial, pasando luego cable UTP y posteriormente a fibra óptica, a través de estos medios físicos tiende hacer un tipo de conexión más estable, a pesar de que tiene también sus versiones, varían en el tipo de conexión, compatibilidad de equipos, alcance del mismo, categoría, referencia y versión del cable utilizado, no todos aptos para las mismas funciones, unos tienen una mayor cobertura y eficiencia basado en los equipos y necesidades de cada escenario [20].

1.1.3 Autenticación Blockchain

La autenticación mediante blockchain, como conocemos, tuvo sus inicios con el surgimiento del bitcoin [21]. El 1 de noviembre de 2008, bajo el nombre de Satoshi Nakamoto, se compartió un enlace a un documento en un sitio web sobre criptografía, que trataba sobre un sistema de dinero digital de tipo P2P. Este sistema, denominado Bitcoin, introdujo una innovadora estructura de datos llamada cadena de bloques o Blockchain, junto con todas las características y pasos necesarios para el funcionamiento de esta red distribuida. Aproximadamente dos meses después, el 3 de enero de 2009, se puso en marcha la primera red P2P basada en el protocolo Bitcoin. Esta red utiliza varias formas de autenticación, entre las cuales se incluyen la identidad digital y modelos de IdM.

La tecnología Blockchain en el Internet de las cosas (IoT) destaca por su capacidad para abordar los desafíos relacionados con la escalabilidad, confiabilidad y privacidad. Facilita la coordinación entre dispositivos, permite el rastreo de millones de dispositivos conectados y procesa transacciones, todo mediante un enfoque descentralizado que emplea algoritmos criptográficos para mejorar la privacidad de los datos de los usuarios. Este

enfoque elimina las fallas y proporciona un ecosistema resiliente. Su conectividad es extraordinaria, ya que el Blockchain se sustenta en una plataforma óptima que resuelve la necesidad de coordinación en el IoT, permitiendo que los dispositivos trabajen de manera integrada sin causar problemas. Es una infraestructura modelada hacia la segura, alejada del modelo centralizado [22], Esta innovadora tecnología ha logrado eliminar intermediarios y descentralizar completamente la gestión de manera segura, gracias al uso de cifrado, lleva años reinventando la manera en la que se gestionan las transacciones, de hecho, se estima que esta descentralización de procesos transforme todavía más el sector de las finanzas. Existen numerosas plataformas con tecnología blockchain, sin embargo, entre las más idóneas para las aplicaciones con autenticación.

1.1.4 Blockchain Ethereum

Ethereum, que inició en el año 2015 con el propósito de crear y desarrollar aplicaciones descentralizadas, opera bajo la tecnología bitcoin con algunas distinciones [23], Ambas plataformas permiten el uso de dinero digital sin depender de proveedores de pago o bancos. Sin embargo, Ethereum es programable, lo que implica que también puede ser empleado para construir y mantener aplicaciones descentralizadas en su infraestructura. La capacidad de ser programable significa que se pueden crear aplicaciones que utilizan la cadena de bloques para almacenar datos o controlar las funciones de su aplicación. Esto resulta en una cadena de bloques de propósito general que puede ser programada para realizar diversas funciones. La falta de límites en las capacidades de Ethereum representa una notable innovación en la red Ethereum, sin embargo como todo desarrollo requiere inversión, así mismo operan las transacciones sobre las plataformas de blockchain, donde cada una de ellas tiene un costo (gas) dependiendo del tipo de cadena, pero para fines de pruebas y desarrollos continuos, existen plataformas y aplicaciones que permiten llevar el escenario a un entorno de pruebas antes de llevar el sistema a transacciones Ethereum.

1.1.5 Contratos Inteligentes

Los contratos inteligentes en la blockchain Ethereum son programas informáticos autónomos que ejecutan automáticamente acciones cuando se cumplen ciertas condiciones predefinidas. Estos contratos están escritos en código y se almacenan en la blockchain, lo que les permite funcionar de manera descentralizada y sin necesidad de intermediarios.

Los contratos inteligentes Ethereum son especialmente conocidos por su capacidad para automatizar y garantizar la ejecución de acuerdos sin la necesidad de confiar en terceros. Utilizan la tecnología blockchain para mantener un registro seguro e inmutable de todas las transacciones y ejecuciones de contrato, lo que garantiza la transparencia y la integridad del proceso.

Estos contratos pueden utilizarse en una variedad de casos de uso, desde transacciones financieras hasta votaciones descentralizadas y registros de propiedad. Su implementación en la blockchain Ethereum ha abierto un amplio abanico de posibilidades para la innovación en diversos campos, aprovechando las ventajas de la descentralización y la seguridad inherentes a la tecnología blockchain.

Para llevar a cabo la ejecución de la blockchain Ethereum, se requiere de Hardhad, un entorno de desarrollo para la creación de contratos inteligentes en Ethereum. Se utiliza principalmente para compilar, probar y desplegar contratos inteligentes de manera eficiente y segura, de igual forma debe hacerse uso del lenguaje de programación específico para este tipo de Blockchain como es solidity.

1.1.6 Lenguaje de programación

Solidity representa un lenguaje de programación de tipado estático que ha sido creado con la finalidad de facilitar la creación de contratos inteligentes ejecutados en la máquina virtual de Ethereum. Este lenguaje de programación, diseñado específicamente para la escritura de contratos inteligentes en la blockchain de Ethereum, otorga a los

desarrolladores la capacidad de definir las reglas y el funcionamiento de las aplicaciones descentralizadas (DApps). Los contratos inteligentes son programas que operan en una red peer-to-peer, donde no existe una autoridad central con control exclusivo sobre la ejecución, lo que posibilita la implementación de diversas lógicas, como tokens de valor, sistemas de propiedad, sistemas de votación y otras funcionalidades [24], es uno de los lenguajes usados, pero de igual forma requiere una compilación y comprobación de funcionamiento la cual se puede trabajar bajo Hardhad. Se presenta como un entorno de desarrollo de Ethereum diseñado para usuarios experimentados y profesionales. Simplifica la realización de tareas comunes, como la ejecución de pruebas, la detección automática de errores en el código y la interacción con contratos inteligentes. Puede explorar la variedad de complementos disponibles para integrar con sus herramientas actuales. Fue creado por la Fundación Nomic en beneficio de la comunidad Ethereum [25].

1.1.7 Integración

Hardhad se integra fácilmente con otras herramientas y servicios comunes en el ecosistema de Ethereum, como: SDK, Metamask, Infura y Ganache, facilitando el desarrollo de aplicaciones descentralizadas (dApps), es altamente personalizable y extensible, lo que permite a los desarrolladores integrar fácilmente nuevas funcionalidades y herramientas según sea necesario para sus proyectos específicos.

Como consiguiente a nivel de integración y para llevar a cabo la interacción de la blockchain con el dispositivo IoT se hace uso de la herramienta del proveedor SDK de hikvision: conjunto de herramientas de desarrollo de software que permite a los desarrolladores integrar funcionalidades de los productos de Hikvision en sus propias aplicaciones o sistemas. Estas herramientas proporcionan acceso a diversas funciones, como la transmisión de video en tiempo real, el control de dispositivos, la gestión de eventos y la reproducción de video grabado.

1.2 Estado del arte

En Colombia, existe una diversidad de empresas dedicadas a ofrecer soluciones en el ámbito del Internet de las Cosas (IoT), y una de ellas es E-Security, que cuenta con más de 20 años de experiencia y ha sido pionera en el país. A través de un artículo, el Gerente de E-Security detalla las ventajas del IoT, que van más allá de la comodidad, abarcando aspectos como la eficiencia en el uso de servicios como el agua y la energía, especialmente en el entorno doméstico. Aunque en los hogares inteligentes se busca optimizar la seguridad, es esencial que las personas, al considerar la implementación de un hogar inteligente y vigilado, busquen asesoramiento previo de empresas con reconocido prestigio en el mercado. Esta asesoría es crucial ya que permite minimizar y gestionar de manera adecuada los diversos riesgos asociados al uso del IoT [26].

Cabe destacar en medio del auge que día a día surge en Colombia, los diferentes trabajos, tesis, investigaciones que van de la mano con este proyecto, entre ellos se han creado artículos de identificación de riesgos como nos informa [27], en la misma relacionan la evaluación de los riesgos asociados a las cámaras IP en entornos residenciales con el propósito de desarrollar una guía de buenas prácticas destinada a usuarios que deseen incorporar este tipo de dispositivos en sus hogares o apartamentos. La relevancia del riesgo puede variar según la criticidad del equipo, sus funciones específicas o la dependencia que se tenga de él. Además, se dispone de un prototipo de un sistema de monitoreo y videovigilancia para el hogar, con un enfoque en la Internet de las Cosas (IoT), compuesto por sensores y cámara de video adaptados una Rasberry, armando un entorno de varios actuadores por medio de un software que notificará los registros a el usuario por medio del celular [28].

La rápida implementación de sistemas de vigilancia en áreas residenciales, metropolitanas y servicios urbanos ha sido crucial para atender las demandas de la población en busca de una mejor calidad de vida. De manera significativa, hemos presenciado una evolución notable en los dispositivos digitales, como teléfonos inteligentes, sensores, aplicaciones inteligentes, actuadores y máquinas inteligentes. Esta evolución ha impulsado

objetivos comerciales claros para la industria de Internet de las cosas (IoT). Ahora, es posible interconectar todos estos nodos y establecer conexiones entre ellos a través de Internet. De este modo las ciudades inteligentes están avanzando en su nivel de inteligencia debido al desarrollo de tecnologías asistidas por computadora. Estas ciudades incorporan una amplia gama de aplicaciones electrónicas, como cámaras de vigilancia y sensores para mejorar la gestión del transporte y otros servicios. El concepto de una ciudad inteligente potenciada por la innovación de IoT se presenta como una idea revolucionaria, pero también surge la preocupación por la seguridad de la información. Las cámaras de televisión de circuito cerrado (CCTV) se han vuelto un componente esencial en estas ciudades inteligentes.

De esta forma es esencial establecer mecanismos robustos de autenticación y autorización para estos dispositivos, garantizando la confianza dentro de las redes de IoT. Dado que los dispositivos IoT generan y transmiten datos sensibles en términos de seguridad, la preservación de la confidencialidad, así como una autenticación y autorización adecuadas, se convierten en prioridades fundamentales para cualquier sistema IoT. La implementación de soluciones basadas en certificados digitales, respaldadas por autoridades de certificación, puede resultar costosa debido a la gran cantidad de dispositivos que participan en estas redes de IoT. En este contexto, la tecnología blockchain emerge como una alternativa viable y rentable para abordar los desafíos de confidencialidad, autenticación y autorización en entornos de IoT. Se propone un sistema basado en blockchain para el registro, autenticación, autorización y preservación de la confidencialidad de los datos generados por los dispositivos IoT [29].

De acuerdo con lo anterior, al ver el avance y crecimiento de estos dispositivos al igual que la inseguridad y fraudes cibernéticos [30], vemos que ya hay empresas y fabricantes que dan un paso e iniciativa a implementar tecnologías que ayuden e impulsen la seguridad de estos dispositivos, entre ellos IoTeX, una plataforma orientada a la privacidad para el Internet de las Cosas, ha colaborado con el fabricante de cámaras Tenvis Technology para lanzar Ucam, una cámara de seguridad para interiores impulsada por tecnología blockchain. Según el director de desarrollo comercial, los usuarios de Ucam pueden acceder a los datos de su cámara a través de un sistema descentralizado y autenticarse

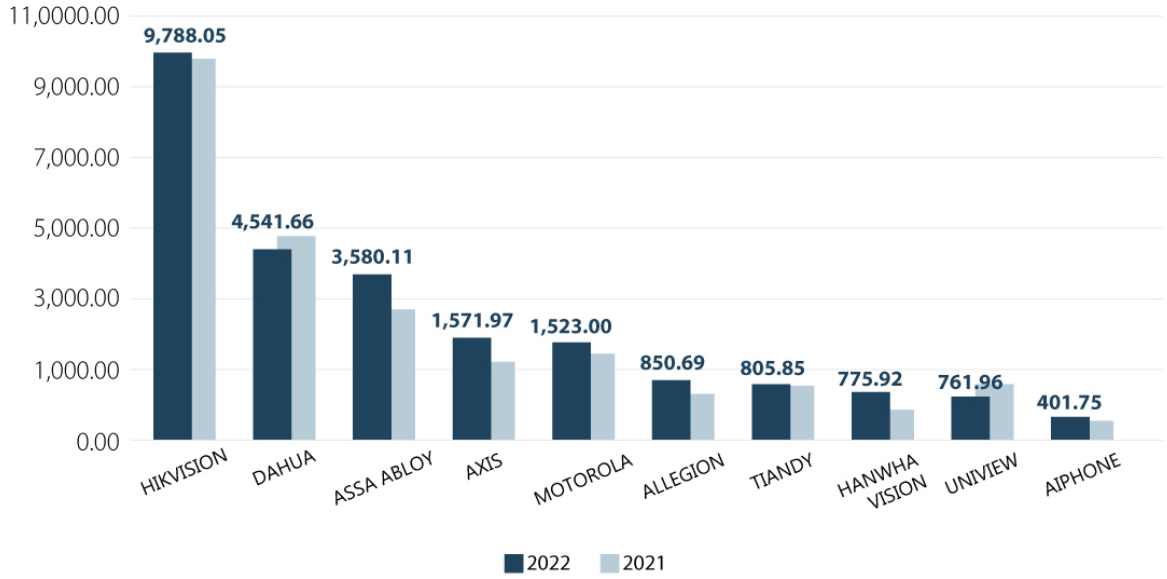
con una contraseña "indescifrable". La computación se realiza en la cámara o el dispositivo móvil del usuario, lo que significa que el proceso de descifrado ocurre localmente, permitiendo a los usuarios tener control total sobre sus propios datos. "Se utiliza una clave privada para cifrar todos los datos de extremo a extremo. Necesitamos tecnología que asegure nuestra privacidad y propiedad, en lugar de depender de términos, condiciones y políticas" [31].

Siguiendo esta línea de investigación, vemos como **Sánchez** [32], Propone en su investigación la unión de la tecnología Blockchain con la tecnología IoT, logrando crear infraestructura segura, descentralizada, y confiable garantizando la integridad de los datos, sobre el crecimiento en dispositivos y de una era de industria 4.0.

Adicionalmente no solo los estudiantes mediante tesis y trabajos de investigación se incorporan a estos avances, también las empresas de tecnología y desarrollo se encuentran dedicando su tiempo y esfuerzos a este avance, debido a que las tecnologías Blockchain han llegado a un nivel de madurez que permite, si se aplican correctamente, gestionar en tiempo real una gran cantidad de datos provenientes de dispositivos IoT. Esto abre nuevas oportunidades para casos de uso donde las redes Blockchain no solo sirven para la certificación, sino también como una base segura para las comunicaciones entre personas y máquinas, dispositivos [33].

En la fig. 6 (Fabricante de seguridad IoT con más ventas), se describe las 10 mejores empresas en seguridad física, representando la cantidad de ventas tanto en 2021 como en 2022, expresando el valor en millones de dólares, de los cuales hace relevancia el proveedor hikvision ubicado en el primer lugar con ventas de 9,788.05 millones de dólares, seguido a la derecha de Dahua con 4.541.66 millones de dólares en ventas en el periodo de 2021 y 2022.

Fig. 6: Fabricante de seguridad IoT con más ventas



Nota. Cantidad de ventas en millones de dólares por fabricante en cámaras de seguridad. Fuente: [34]

2. Metodología y resultados

El desarrollo de esta investigación tiene como resultado previsto la disminución de brechas de vulnerabilidades existentes en IoT para cámaras de vigilancia en ambiente masivo definido como hogares, proponiendo para este proyecto una metodología cualitativa [35]. Toda vez que está orientada a recopilar y analizar datos y otros detalles, de tipo inductivo, puesto que permite explorar y describir diferentes escenarios [35].

De acuerdo con el objetivo general y los 4 objetivos específicos, para el cumplimiento y el desarrollo del proyecto se planteó en 4 fases, en donde cada una de estas corresponde a un objetivo específico y de las cuales se derivan una serie de actividades.

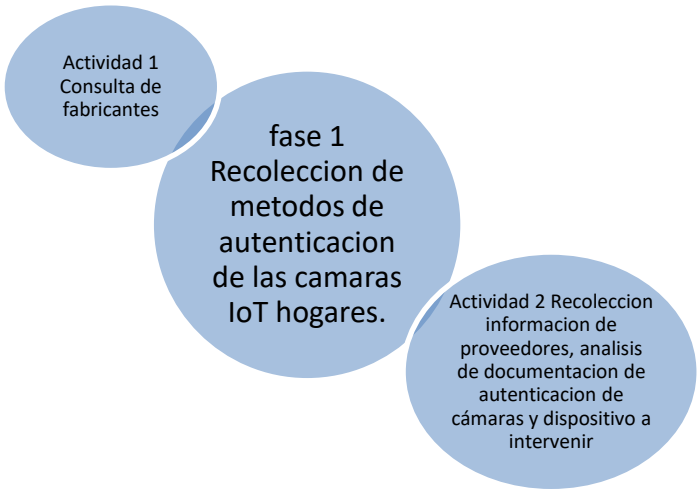
Tabla 2. Descripción de fases y actividades por objetivos

OBJETIVO	FASE	ACTIVIDAD	ENTREGABLE
1: Caracterizar información de las diferentes autenticaciones y protocolos de las cámaras de vigilancia	1: Recolección de métodos de autenticación de las cámaras de vigilancia y protocolos de comunicación.	1 -Consulta de fabricantes. 2- Recolección de información de proveedores, análisis de documentación de autenticación de cámaras y selección de dispositivo a intervenir.	información de las autenticaciones y protocolos de las diferentes cámaras identificadas, y cámara a intervenir
2: Identificar amenazas y riesgos de las cámaras de vigilancia seleccionadas enfocadas en hogares.	2: identificación de amenazas y riesgos	1 - análisis de las situaciones que dan origen a las vulnerabilidades o riesgos. 2 - Captura de resultados. 3- Riesgos consolidados.	Informe de vulnerabilidades y resultados.
3: Planear las configuraciones de autenticación basado en las amenazas y riesgos identificados.	3: Planeación de la configuración.	1 – Definición de la arquitectura base 2 - Construcción modelo de autenticación.	Pautas, código y plan de desarrollo.
4: Validar el modelo aplicado para la configuración de la autenticación a través de blockchain para las cámaras de vigilancia.	4: Validación.	1-Pruebas de funcionamiento y validación	Registro de pruebas de funcionamiento.

Nota. Descripción de las fases con las actividades correspondientes para cumplir el objetivo. Fuente: diseño propio.

A continuación, se describen cada una de las fases para dar cumplimiento a los objetivos.

Fig. 7: Esquema de actividades referente a la fase 1



Nota. Actividades relacionadas en ejecución a la fase 1. Fuente: diseño propio.

2.1 Fase 1 Recolección de métodos de autenticación de las cámaras de vigilancia y protocolos de comunicación.

En esta fase se realizó una consulta de información sobre las autenticaciones de algunas marcas de cámaras de vigilancia y protocolos de comunicación de los dispositivos IoT hogares.

2.1.1 Consulta de fabricantes

Se consultó en diferentes fuentes de información sobre fabricantes de cámaras IoT cuyo foco es la vigilancia o monitoreos de hogares, consulta realizada en foros técnicos, páginas relacionadas en la fabricación y distribución de estos dispositivos.

Al consultar en los sitios web se obtuvieron fabricantes de la marca Geovisión, hikvision, Dahua, axis, Bosch, Panasonic, Nest, ring, Arlo, Blink, Wyze y TP-link, sus países de orígenes son respectivamente Taiwán, China, Suecia, Alemania, Japón y estados unidos, en estas se identifica que la mayoría fabrican dispositivos para el sector industrial y pymes, solo algunos se centran en el sector hogares y de fácil acceso al dispositivo (compra, disponibilidad) como lo son TP-link, Dahua y hikvision, donde realizan distribuciones a los países latinoamericanos para ser distribuidos por ciudades principales, como Bogotá, Medellín y Cali en lo que relaciona a Colombia.

En las búsquedas realizadas a nivel web sobre distribuidores de la marca Dahua, se encontraron los siguientes distribuidores.

De igual forma se identifican algunos de los proveedores de cámaras Dahua en Medellín que aportan al sector pyme y sector hogares, el primero que se identificó mejor calificado es DYF seguridad, este cuenta con gran portafolio en cámaras, pero no cuenta con área especializada de soporte, el área más relevante es la comercial, para obtener equipos, e interactuar con la empresa se requería del RUT (registro único tributario) para ser inscrito como integrador y así poder tener acceso a la lista de referencias y precios de los equipos.

El segundo mejor calificado se encontró GVS con una calificación de 4.7 sobre 5.0, a este se visitó y se identificó que cuenta con gran portafolio de referencias de cámaras, áreas de soporte y una amplia área comercial y proyectos. para acceder como cliente o integrador de la empresa se requería RUT (registro único tributario) donde indicara la actividad de comercio de equipos electrónicos y telecomunicaciones.

El tercer distribuidor calificado es tácticas en seguridad, el cual no cuenta con un establecimiento comercial, se encuentra ubicado en laureles en una unidad residencial, por lo cual no dispone de áreas comerciales, áreas de soporte y áreas de proyectos, de la misma manera este requiere RUT (registro único tributario), con este distribuidor no se realizó proceso de inscripción.

De la misma forma se Consulta distribuidores de la marca Hikvision, obtenido el resultado de calificación de la siguiente manera.

Se valida el resultado encontrado para las cámaras de la marca Hikvision, donde esta vez registra la empresa fénix como mejor calificado, pero los precios que manejaban eran más caros para personas con Rut, para un buen precio debía ser con cámara de comercio.

El segundo mejor calificado fue Grupo control el cual contaba con una gran cantidad de referencias en stock, área de proyectos, área de soporte especializado, área comercial y demás áreas administrativas. Este permitió el registro a través de RUT (registro único tributario) dando buenos precios y soporte técnico.

En la consulta del tercer distribuidor, era similar al segundo, pero sin área de soporte especializado, por ende, se selecciona el proveedor segundo mejor calificado con 4.7 sobre 5.0.

Al visitarlo se logró el registro de la misma manera que con los anteriores, mediante RUT (registro único tributario), el distribuidor contaba con área comercial y área de soporte al igual que las áreas administrativas, lo que indicaba y daba buena confiabilidad.

En la tabla a continuación tabla 3. (Calificación consulta proveedores), se resumen las consultas realizadas sobre diferentes navegadores y calificaciones realizadas por usuarios y clientes de los proveedores, de acuerdo con la marca del dispositivo, la puntuación está entre 1 y 5, siendo cinco la calificación más alta y uno la más baja.

Tabla 3. Calificación consulta proveedores

PROVEEDOR	MARCA	PUNTUACION	REFERENCIA PUNTUACION
DYF SEGURIDAD	DAHUA	4.9	5.0
GVS COLOMBIA	DAHUA	4.7	5.0
FENIX	HIKVISION	4.8	5.0
GRUPO CONTROL	HIKVISION	4.7	5.0
SEGURPROF	V-SUPRA, V380	4.8	5.0

Nota. Descripción de resultados de marcas de cámaras con los respectivos proveedores y calificación.
Fuente: diseño propio.

2.1.2 Recolección de información de proveedores, análisis de documentación de autenticación de cámaras y selección de dispositivo a intervenir.

Luego de ser realizada la consulta y haber identificado algunos proveedores a través de la web, varios distribuidores se encontraron ubicados en Medellín, Bogotá y Cali, siendo los proveedores de Medellín un sector más cerca. se realizó un trabajo de campo ejecutando un desplazamiento a 3 de los distribuidores ubicados en la misma zona donde se expuso el caso de estudio y el alcance del trabajo de investigación, generando así un ambiente de confianza y colaboración para tener acceso a la información necesaria.

Después del acercamiento con algunos de los distribuidores de las cámaras de seguridad IoT hogares y con los contactos del personal de soporte, se hace solicitud de la documentación necesaria, y uno de los dispositivos recomendados a intervenir junto con ello ficha técnica, manuales de configuración y manual de usuario (importante para la relación de los equipos con la red y el usuario final), y sus características generales, donde se identifican El modo de uso, Configuración inicial, , direcciones IP default, métodos de autenticación, protocolos de comunicación y tipo de aplicaciones disponibles para visualizaciones remotas.

Estando ya registrado con los tres proveedores (Gvs, grupo control y Segurprof), se realiza visita técnica donde se habla con el personal técnico de cada empresa, recibiendo las recomendaciones de acuerdo a la necesidad expuesta y alcance del proyecto, para lo cual se determinó la cámara más vendida para el sector hogares, acceso web, precio y disponibilidad, por lo que compartieron las referencias a trabajar por cada proveedor respectivamente (DH-IPC-HDW1239T1-LED-S5, DS-2CV2Q21FD-IW(2.8mm) y Cámara wifi V380 3.6mm) las cuales tuvieran las mismas características en mayor porcentaje. En la

documentación se identifican los métodos de autenticación, los protocoles de conexión y cifrado.

Como resultado obtenido de la consulta de los dispositivos de diferente fabricante, se relacionan las tres referencias de cámaras consultadas de acuerdo con la información de cada proveedor, con los métodos de autenticación, protocolos de comunicación y cifrado por cada referencia junto con su fabricante, detallando la información en la tabla 4. Se logró identificar que las tres referencias seleccionadas tienen similitudes en métodos de autenticación, protocolos de comunicación y cifrado, de los cuales se tomaron como base para la continuación y desarrollo de la tesis, el método de acceso web, el protocolo de comunicación HTTP mediante el estándar 802.11g/n y el cifrado WPA/WPA2.

Para llevar a cabo un análisis riguroso de la autenticación de dispositivos IoT, especialmente en cámaras hogar, es fundamental definir claramente los elementos clave a considerar. En ese orden de ideas los atributos no solo proporcionan una base sólida para la investigación, sino que también asegura que se aborden todos los aspectos críticos pertinentes. A continuación, se presenta una propuesta de atributos que deben ser tenidos en cuenta y en los cuales se pone especial énfasis.

Método de acceso: Este atributo se refiere a cómo se accede al dispositivo IoT, siempre y cuando se tenga enlazada alguna de las interfaces de red (como Wi-Fi, Ethernet), permitiendo el acceso web, y/o acceso con la APP. La seguridad del método de acceso puede determinar la resistencia del dispositivo frente a ataques físicos y la facilidad con la que se puede acceder remotamente.

Protocolos de comunicación: Los protocolos de comunicación utilizados por el dispositivo IoT son esenciales para garantizar la integridad y confidencialidad de los datos transmitidos. Analizar estos protocolos permite identificar posibles vulnerabilidades en la capa de comunicación que podrían ser explotadas por atacantes.

Cifrado: La utilización de técnicas de cifrado adecuadas es fundamental para proteger la confidencialidad de la información transmitida entre el dispositivo IoT y otros

sistemas. Evaluar los algoritmos de cifrado utilizados y su implementación es crucial para determinar la robustez del sistema frente a ataques de interceptación de datos.

Configuraciones por defecto: Muchos dispositivos IoT vienen con configuraciones predefinidas que pueden ser inseguras o fácilmente explotables. Evaluar las configuraciones por defecto del dispositivo permite identificar posibles puntos débiles que podrían ser utilizados por atacantes para comprometer la seguridad del sistema.

Al justificar la inclusión de estos atributos en el análisis de la autenticación de dispositivos IoT, se puede demostrar cómo cada uno de ellos contribuye a la identificación de vulnerabilidades en los procesos de autenticación. Además, esta definición sienta las bases para el desarrollo de la tabla #4, que refleja los resultados obtenidos en el estudio y proporciona una visión clara de las áreas de mejora en la seguridad de los dispositivos analizados.

2.1.2.1 Métodos de acceso

Acceso web: La mayoría de las cámaras de hogares venían con solo acceso web, lo que ha ido reduciendo a través del tiempo por múltiples factores entre ellos seguridad y configuración. Este consta de un navegador web ya sea en dispositivo celular, Tablet o equipo de cómputo, el cual apunta a una dirección IP e indicando un puerto de acceso, de fabrica viene el 80 por ende no se requiere especificarlo. Mediante este acceso la cámara responde con una página solicitando un usuario y una contraseña para autenticarse al dispositivo, en su mayoría el usuario por defecto es admin y la contraseña es 1234, en algunos firmwares la contraseña previamente debe ser configurada para poder iniciar la cámara.

Acceso APP: Este acceso se centra en una aplicación móvil y desktop desarrollada por el fabricante, la cual contiene 4 formas de configurar o enlazar la cámara hacia el dispositivo (IP/Dominio, DDNS, código QR o dispositivo en línea). Para la configuración IP/Dominio se requiere publicar la dirección IP y puertos de la cámara, un usuario y una contraseña. Para el acceso DDNS muy poco ya utilizado, se requiere de un registro en una

página de un tercero donde relacione una dirección IP y este se encarga de traducir a un nombre de dominio, utilizado anteriormente con servicios de IP publicas dinámicas. Código QR, esta configuración solo requiere de tener la cámara conectada a internet, luego en la configuración se lee el código QR etiquetado por debajo o en un lado de las cámaras, la mayoría los trae en los manuales o etiquetado en la caja del producto, al escanear este código, la cámara se direcciona hacia un servidor del fabricante (nube) el cual administra la conexión y le permite acceder al recurso, actuando como un puente, es decir, la cámara va hacia el servidor y retorna hacia el dispositivo móvil del usuario, este es el único método que se usa actualmente para configurar la cámara desde cero, sin configuración previa, ya que la misma aplicación va indicando paso a paso e incluso asignar la contraseña porque el usuario siempre prevalece el mismo en todas las cámaras(admin), de no configurar la cámaras desde cero con el QR se debe acceder mediante un navegador web.

La mayoría de las cámaras hogares cuando se encienden, irradian su propio SSID el cual es reconocido por la misma aplicación del fabricante, con el fin de generar una red local entre la cámara y el dispositivo, creando un ambiente de confianza para iniciar la configuración y vinculación.

Las aplicaciones móviles para cada marca, disponibles para Android y IOS son las siguientes:

Dahua: gDMSS Lite
Hikvision: IVMS 4500 antes, actualmente Hik-Connect, para equipos de cómputo IVMS 4200.
Shenzhen Vstarcam Technology: V380

2.1.2.2 Protocolo de comunicación

HTTP: Estos dispositivos IoT, tienen una interfaz web a la que se puede acceder a través de un navegador. Utilizando HTTP, los usuarios pueden acceder a esta interfaz para ver el flujo de video en tiempo real, configurar opciones de la cámara, como la resolución de video, activar o desactivar funciones como el movimiento o el sonido, y acceder al

historial de grabaciones, modificar los parámetros de seguridad. A su vez HTTP opera sobre el puerto 80 por defecto y es un protocolo no seguro, lo que significa que la información transmitida no está encriptada y puede ser interceptada y leída por terceros

802.11b (CCK, QPSK, BPSK): Compatibilidad heredada. Si se requiere soporte para dispositivos más antiguos que solo son compatibles con el estándar 802.11b, como algunos dispositivos más antiguos como impresoras, cámaras IP o equipos de red, entonces el uso de este protocolo es necesario para asegurar la interoperabilidad. En situaciones donde la velocidad de transmisión no es crítica y la prioridad está en la compatibilidad y el alcance, como en aplicaciones de Internet de las cosas (IoT) donde se transmiten datos pequeños de manera intermitente, el estándar 802.11b puede ser suficiente.

802.11g/n (OFDM): Mayor velocidad de datos: Si se requiere una mayor velocidad de transmisión, como en aplicaciones de transmisión de video de alta definición o transferencia de archivos grandes, entonces los estándares 802.11g/n son preferibles debido a su capacidad para ofrecer tasas de transferencia de datos más altas. En entornos donde hay una alta densidad de redes inalámbricas o dispositivos que generan interferencia, como en áreas urbanas densamente pobladas o entornos empresariales, OFDM (utilizado en 802.11g/n) puede ser más robusto y resistente a la interferencia que los protocolos más antiguos como 802.11b.

Alcance mejorado: OFDM también puede proporcionar un mejor rendimiento en términos de alcance y cobertura en comparación con 802.11b en ciertos escenarios, lo que lo hace más adecuado para despliegues de redes inalámbricas en áreas grandes o entornos con obstáculos físicos.

2.1.2.3 Cifrado

El protocolo WEB se refiere a HTTP (Hypertext Transfer Protocol), que es el protocolo de comunicación utilizado para transferir datos en la web. HTTP no proporciona cifrado de datos de forma nativa, lo que significa que la información transmitida a través de HTTP es vulnerable a ser interceptada y leída por terceros. Para mejorar la seguridad de las comunicaciones web, es recomendable utilizar HTTPS en lugar de HTTP. HTTPS utiliza

SSL/TLS para cifrar los datos transmitidos entre el cliente y el servidor, proporcionando así una capa adicional de seguridad.

WPA2/WPA3:WPA2 (Wi-Fi Protected Access 2) y WPA3 son protocolos de seguridad utilizados en redes Wi-Fi para proteger la integridad y confidencialidad de los datos transmitidos. Ambos protocolos utilizan cifrado AES (Advanced Encryption Standard) para proteger las comunicaciones inalámbricas entre dispositivos y puntos de acceso.

WPA2 utiliza TKIP (Temporal Key Integrity Protocol) o AES-CCMP (Counter Mode with Cipher Block Chaining Message Authentication Code Protocol) para cifrar los datos, mientras que WPA3 utiliza exclusivamente AES-CCMP, que es más seguro que TKIP.

WPA3 también introduce mejoras en la autenticación y en la protección contra ataques de fuerza bruta, lo que lo hace más resistente a las vulnerabilidades conocidas en WPA2.

SSL (Secure Sockets Layer) y TLS (Transport Layer Security): son protocolos de seguridad utilizados para cifrar la comunicación entre un cliente y un servidor en internet. Utilizan algoritmos de cifrado simétricos y asimétricos para proteger los datos transmitidos, así como para proporcionar autenticación del servidor y, opcionalmente, del cliente, siendo TLS una versión más segura y actualizada de SSL, y se recomienda su uso sobre SSL debido a las vulnerabilidades conocidas en las versiones anteriores de SSL.

Estos protocolos se utilizan comúnmente para asegurar conexiones HTTPS, pero también se pueden utilizar para proteger otros servicios de red, como correo electrónico, transferencia de archivos y mensajería instantánea.

Los protocolos antes mencionados como tal cuentan con un buen uso cotidiano y extendido, 802.11b (que utiliza modulación CCK, QPSK, y BPSK) y 802.11g/n (que utiliza OFDM, y HTTP, siendo estos los principales usados para los dispositivos tratados en este trabajo de grado, aunque como se ha mencionado son ampliamente utilizados, presentan varias fallas de seguridad debido a su naturaleza no cifrada y a la falta de autenticación

robusta, susceptibles a diversas fallas de seguridad, algunas de las cuales se han explotado en el pasado y siguen representando riesgos para las redes de estos dispositivos IoT.

Tabla 4. Relación de referencias, protocolos y fabricantes de cámaras

Referencia / Tipo	**Fabricante**	**Distribuidor**	**Método de acceso**	**Protocolo de comunicación**	**Cifrado**
Cámara wifi V380 3.6mm	Shenzhen Vstarcam Technology	Segurprof	-Acceso web -Acceso APP	RTSP, HTTP, ONVIF, 802.11/b/g	WPA2/WPA3 SSL/TLS
DS-2CV2Q21FD-IW(2.8mm)	Hikvision	Grupo Control de Colombia	-Acceso web -Acceso APP	HTTP, RSTP, HTTPS, SERVER PORT, IEEE 802.3, 802.11b: CCK, QPSK, BPSK, 802.11g/n: OFDM	64/128-bit WEP, WPA/WPA2, WPA-PSK/WPA2-PSK, WPS
Cámara wifi V380 3.6mm	Shenzhen Vstarcam Technology	Segurprof	-Acceso web -Acceso APP	RTSP, HTTP, ONVIF, 802.11/b/g	WPA2/WPA3 SSL/TLS

Nota. Relación de métodos de autenticación, protocoles de comunicación y fabricante. Fuente: Diseño propio con información de cada manual.

De acuerdo con la información consolidada en la tabla 4. (Relación de referencias, protocolos y fabricantes de cámaras), se consulta en shodan.io sobre las dos marcas más comunes obteniendo un resultado mayor sobre la marca hikvision por ende es uno de los puntos que llevó a seleccionar el dispositivo para seguir el desarrollo del trabajo de grado.

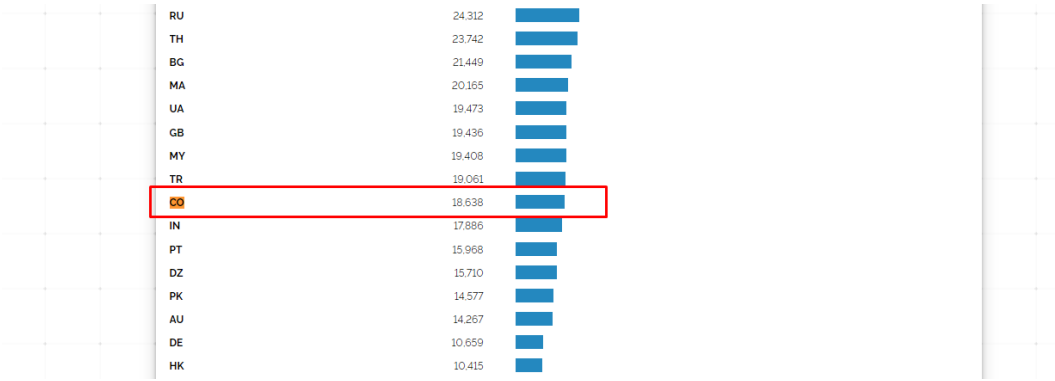
En la imagen a continuación se relaciona el resultado obtenido en la consulta realizada en shodan.io sobre la cantidad de cámaras hikvision en Colombia, el cual registró 37.736 dispositivos.

Fig. 8: Cantidad de Cámaras hikvision en Colombia

Nota. Resultado de búsqueda de distribuidores de cámaras hikvision en Medellín. Fuente: [5]

De igual forma se procedió con la misma consulta, pero en la marca Dahua la cual relaciona en la imagen a continuación Fig. 9 (Cantidad de Cámaras Dahua en Colombia), el resultado de 18.638 dispositivos, de la marca en Colombia.

Fig. 9: Cantidad de Cámaras Dahua en Colombia



Nota. Resultado de búsqueda de distribuidores de cámaras Dahua en Medellín. Fuente: [5]

Teniendo en cuenta las investigaciones anteriores y la información recopilada el equipo que se seleccionó y se compró es la cámara **DS-2CV2Q21FD-IW(2.8mm).**

A continuación, se describen las características básicas de la Cámara seleccionada, entre las cuales se destacan el video, los parámetros de red, resolución y audio.

Tabla 5. Características resaltadas cámara IoT hogares

Ítems	**Especificación**
Sensor de Imagen	1/2.8" Progressive Scan CMOS
Max. Resolución	1920 × 1080
Iluminación Mínima	0.01 Lux @ (F1.2, AGC ON), 0 Lux with IR
Tiempo de Obturación	1/3 s to 1/100,000 s
Dia y Noche	Filtro de corte IR con interruptor magnético
Ajuste Angulo	Pan: 0° to 355°; Tilt: -10° to 90°
flujo Principal	50 Hz: 25 fps (1920 × 1080, 1280 ×720),60Hz: 30fps (1920 × 1080, 1280 ×720)
Flujo Secundario	50 Hz: 25 fps (704 × 576, 352 × 288, 640 × 480, 320 × 240),60 Hz: 30 fps (704 × 480, 352 × 240, 640 × 480, 320 × 240
Tasa de Bits	32 kbps – 8 Mbps

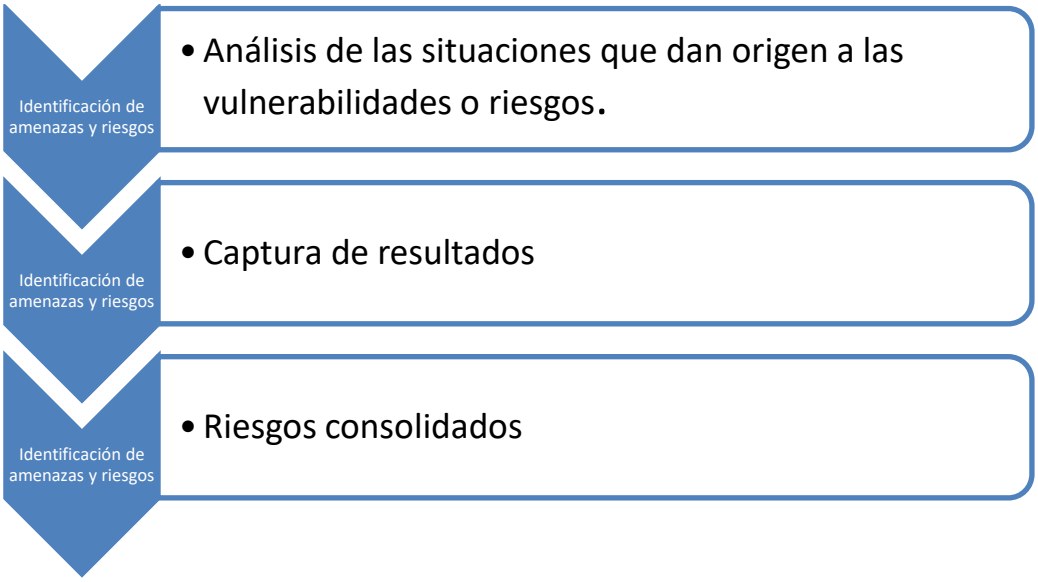
Compresión de Video	Flujo Ppal: H.264, Secundario: H.264
Audio	Soporta banda sonora mono
Compresión Audio	G.711/G.722.1/G.726/MP2L2/PCM
Bits de Audio	64Kbps (G.711) /16Kbps (G.722.1) /16Kbps(G.726)/32-160Kbps(MP2L2)
Frecuencia de Audio	Max. 16 kHz
Modo de Cifrado	64/128-bit WEP, WPA/WPA2, WPA-PSK/WPA2-PSK, WPS
Protocolo 802.11	802.11b: CCK, QPSK, BPSK, 802.11g/n: OFDM
Rango Frecuencia	2.4 GHz - 2.4835 GHz
Canal Ancho Banda	20/40 MHz
Radio de Transferencia	11b: 11 Mbps, 11g: 54 Mbps, 11n: up to 150 Mbps
Alcance Inalámbrico	50 mts

Nota. Características de operación del dispositivo seleccionado. Fuente: [36]

2.2 Fase 2 identificación de amenazas y riesgos.

De acuerdo con la información recopilada y material seleccionado previamente, se procede con un análisis de vulnerabilidades y riesgos al dispositivo seleccionado previamente bajo el costo, características y disponibilidad.

Fig. 10: Esquema de actividades referente a la fase 2



Nota. Actividades relacionadas en ejecución a la fase 2. Fuente: diseño propio.

En las consultas realizadas sobre vulnerabilidades en la marca, se lograron identificar en una consulta básica, alrededor de 19 vulnerabilidades entre 2013 y 2023 sobre algunos de sus productos relacionadas con control de acceso, Cross-site scripting, inyección de código, enumeración de usuarios, desbordamiento de búfer, Denegación de servicio, generación códigos de recuperación de contraseña, escalada de privilegios entre otros, en los cuales se vieron dispositivos afectados como Clúster, San (Storage Área Network), DVR(Digital Video Recorder), Cámaras IP, NVR (Network Video Recorder), y software [37].

2.2.1 Análisis de las situaciones que dan origen a las vulnerabilidades o riesgos.

La marca seleccionada ha tenido grandes impactos a nivel de seguridad a través del tiempo, debido a su auge, parecido a lo que se vive hoy en día con los sistemas operativos Windows, las cuales tienen más eventos de seguridad que otras marcas, entre ellas las

relacionadas en la tabla 1, de acuerdo con los incidentes que se han venido presentando en la marca en varios de sus productos, y desde el 2013. Al tener vulnerabilidades masivas han generado firmware que contrarrestan ciertas fallas de seguridad, lo que indica que un técnico, integrador o usuario con conocimiento debe ingresar a la página de Hikvision, descargar el firmware apropiado y aplicarlo, siendo esto un proceso que no cualquier persona va a saber hacerlo y no todos están enterados del evento, para lo cual genera problemas ya que si el instalador los contacta y les explica la situación el usuario pensaría que solo es interés económico en un servicio, lo que conlleva que no todos los usuarios ejecuten y corrijan las vulnerabilidades, lo que indica que las vulnerabilidades seguirán estando presentes en muchos dispositivos. Teniendo en cuenta lo anterior y otros tipos de eventos, la marca decide adicionar en su manual una exoneración de seguridad, traducido del inglés, “descargo de responsabilidad: en la medida máxima permitida por la ley aplicable, este manual y el producto descritos, con su hardware, software y firmware, se proporcionan "tal cual" y "con todas las fallas y errores" minúscula intencional [36].

De la misma forma aclara en la segunda hoja que, “en ningún caso Hikvision será responsable ante usted por cualquier daño especial, consecuente, incidental o indirecto, incluyendo, entre otros, daños por pérdida de ganancias empresariales, interrupción o pérdida de datos, corrupción de sistemas o pérdida de documentación, ya sea basado en incumplimiento de contrato, agravio (incluyendo negligencia), responsabilidad del producto, o de otra manera, en relación con el uso del producto, incluso si hikvision ha sido informado de la posibilidad de tales daños o pérdidas” [36].

Siguiendo con el desarrollo de este objetivo, y en el análisis se encontró:

CVE-2017-7921: En esta vulnerabilidad encontrada, con calificación 10 en la versión 3.0 y 7.5 en la versión 2.0, se identificó un problema de autenticación inadecuada en varios productos Hikvision, que incluyen las series (desde la versión 5.2.0 build 140721 hasta la hasta la 5.3.5 build 160106).

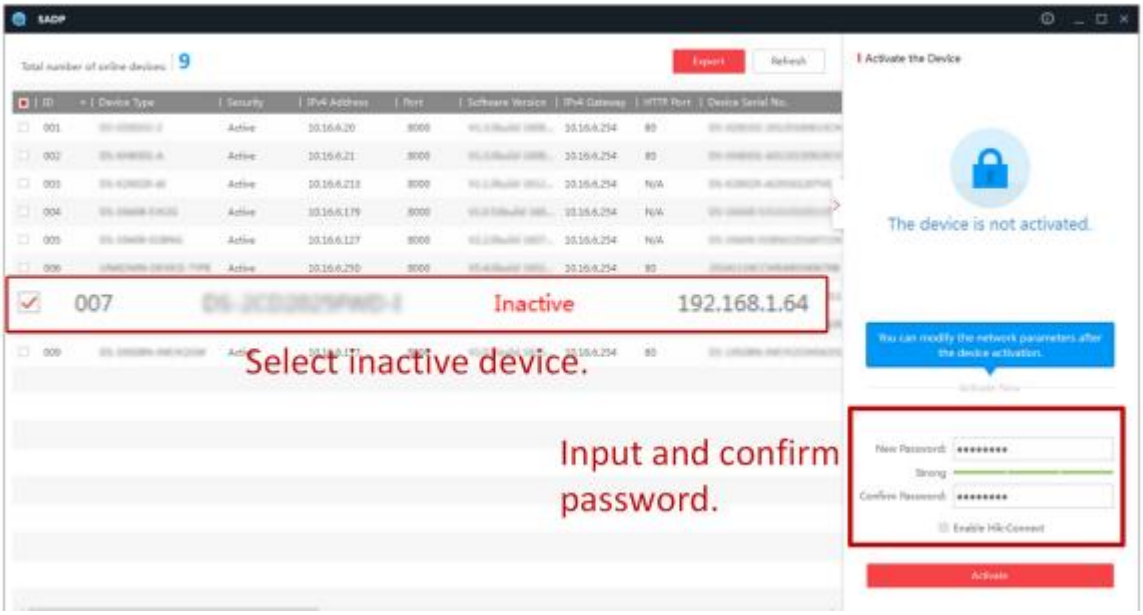
La vulnerabilidad de autenticación inadecuada se presenta cuando una aplicación no verifica de manera adecuada la identidad de los usuarios. Este fallo podría posibilitar que un usuario malintencionado aumente sus privilegios en el sistema y obtenga acceso a información confidencial [38].

CVE-2017-7923: En esta se encontró una calificación v3.0 de 8.8 (alta) y en la versión 2.0 4.0 (media), en esta vulnerabilidad se encontró un problema relacionado con contraseñas en archivos de configuración en varios dispositivos Hikvision, (desde la versión 5.2.0 build 140805 hasta la 5.4.5 build 160928) y dispositivos DS-2CD63xx (desde la versión 5.0.9 build 140305 hasta la 5.3.5 build 160106).

Esta vulnerabilidad relacionada con contraseñas en archivos de configuración podría permitir a un usuario malicioso aumentar sus privilegios o asumir la identidad de otro usuario y acceder a información confidencial [39].

El manual para este equipo corresponde al UD28967B-A_Network-Camera_User-Manual_5.7.20_20221215, en la página 2 de este manual se encontró la IP por defecto y la aplicación con la cual se configura los parámetros de red y contraseña inicialmente, debido a que el usuario siempre permanece por defecto(admin), este no permitió editarlo o deshabilitarlo, lo que genera el primer dato de autenticación para los ciberdelincuentes.

Fig. 11: Dirección IP por defecto.



Nota. Identificación dirección IP por defecto cámara hikvision. Fuente: [36]

Con el conocimiento de la dirección IP, es posible ingresar al navegador web mediante el protocolo HTTP, acceder al entorno login del dispositivo, lo que indica que contine directorios, al tener el acceso a los directorios, estos permiten que el atacante ingrese al directorio que aloja los registros de autenticación, como lo son usuario y contraseña, y de la misma forma editarlos a beneficio.

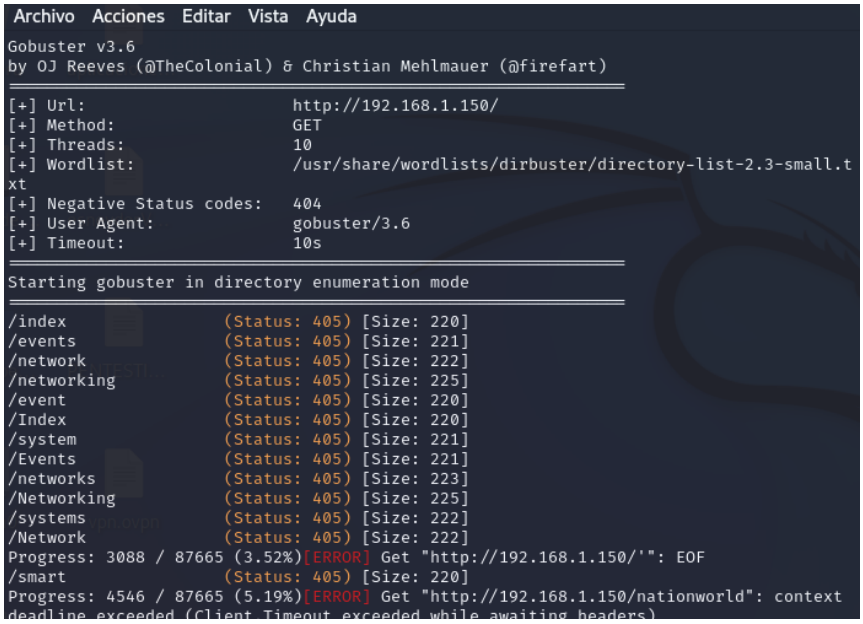
Para determinar que directorios hay disponibles en un servidor, existen varias formas, sin embargo, la más efectiva se concentra en realizar peticiones HTTP de directorios y extensiones de archivo comunes o bien conocidos, por ejemplo, el directorio home o el archivo Home.PHP Por consiguiente, las aplicaciones de fuerza bruta de directorios toman de una lista los directorios conocidos y ensamblan las peticiones HTTP correspondientes. Existen muchas herramientas de este tipo, sin embargo, Gobuster es una de las más efectivas. Para determinar, que directorios hay disponibles en el servidor se puede usar el siguiente comando:

Gobuster dir -u http://192.168.1.150/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt

- -u: Recibe como argumento la URL a escanear
- -w: Recibe como argumento la lista de directorios a escanear.
- dir: Especifica que se va a usar el módulo de directorios de Gobuster.

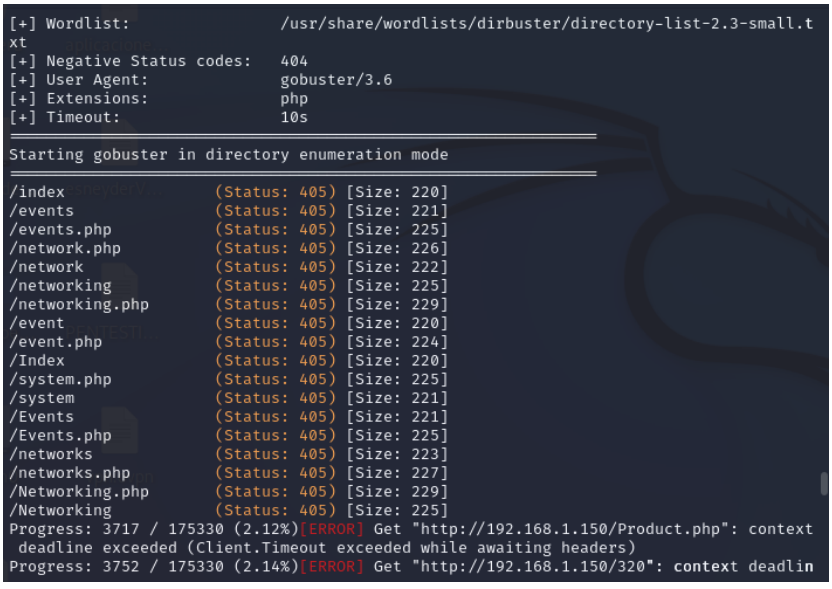
En la fig.11 (Directorio raíz) y fig. 12 (Directorio raíz .PHP), se puede verificar los directorios que componen toda la configuración del dispositivo, página de inicio, parámetros de red, sistema, eventos.

Fig. 12: Directorio raíz



Nota. Identificación de directorios. Fuente: Prueba con Kali Linux

Fig. 13: Directorio raíz .PHP



Nota. Identificación de directorios de configuración .PHP. Fuente: Prueba con Kali Linux

Configuraciones por defecto: esta es una de las situaciones que están aprovechando los ciberdelincuentes para autenticarse de forma no autorizada en los dispositivos, abarcan

usuario, contraseña, dirección IP, puertos, protocolos, SSID predeterminado sin cifrado. En algunos casos solo los usuarios cambian los parámetros de red, dejando el servicio en DHCP con el fin de vincularlos a la red local, y dejan las demás configuraciones por defecto.

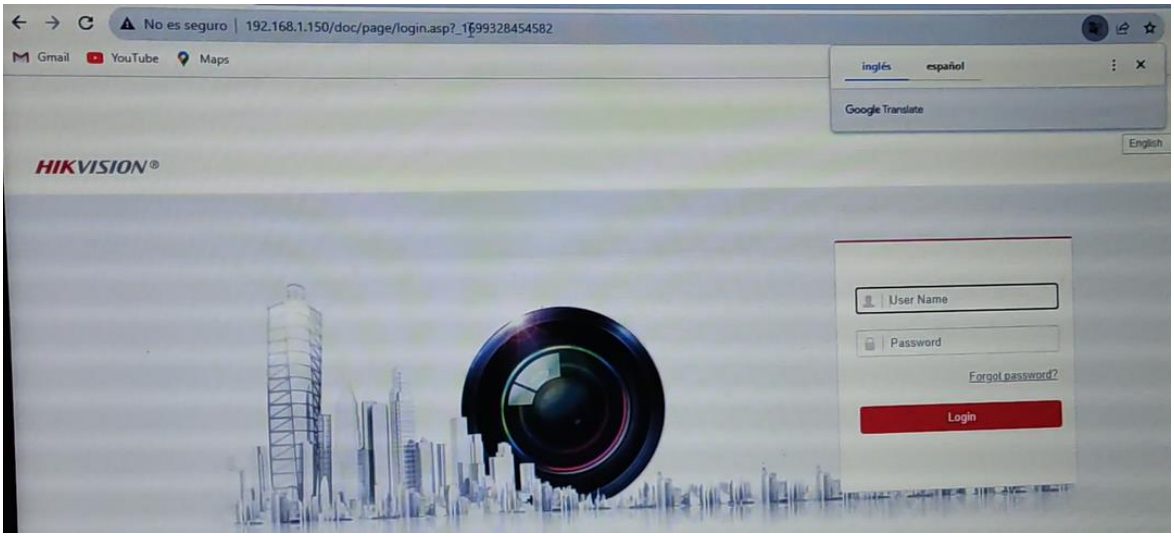
Autogestión y conocimiento: Al ser un dispositivo IoT, los usuarios cada vez más adquieren estos equipos mediante tiendas virtuales, como cualquier otro dispositivo. Consideran un dispositivo más, con un manual y un video en YouTube de cómo ponerlo a funcionar. Pero de tras de esto, está el conocimiento técnico que se debe tener para editar las diferentes configuraciones que traen los equipos como tal, que permiten fortalecer la seguridad, habilitación de cifrado tanto credenciales como de contenido, periodo de inactividad por intentos fallidos, cantidad de intentos disponible antes de bloqueo, etc.

2.2.2 Captura de resultados.

Al terminar el análisis en el apartado anterior, se tomó registro de capturas de pantalla, almacenando los resultados de manera digital, en un archivo de Word, y PDF, nombrado Registro Evidencias.

En la fig. 14 (Directorio web de la ruta de acceso login), se identifica el acceso a la página de inicio de sesión del dispositivo, mediante la dirección IP, y el directorio del acceso web, logrando llegar hasta el login de acceso.

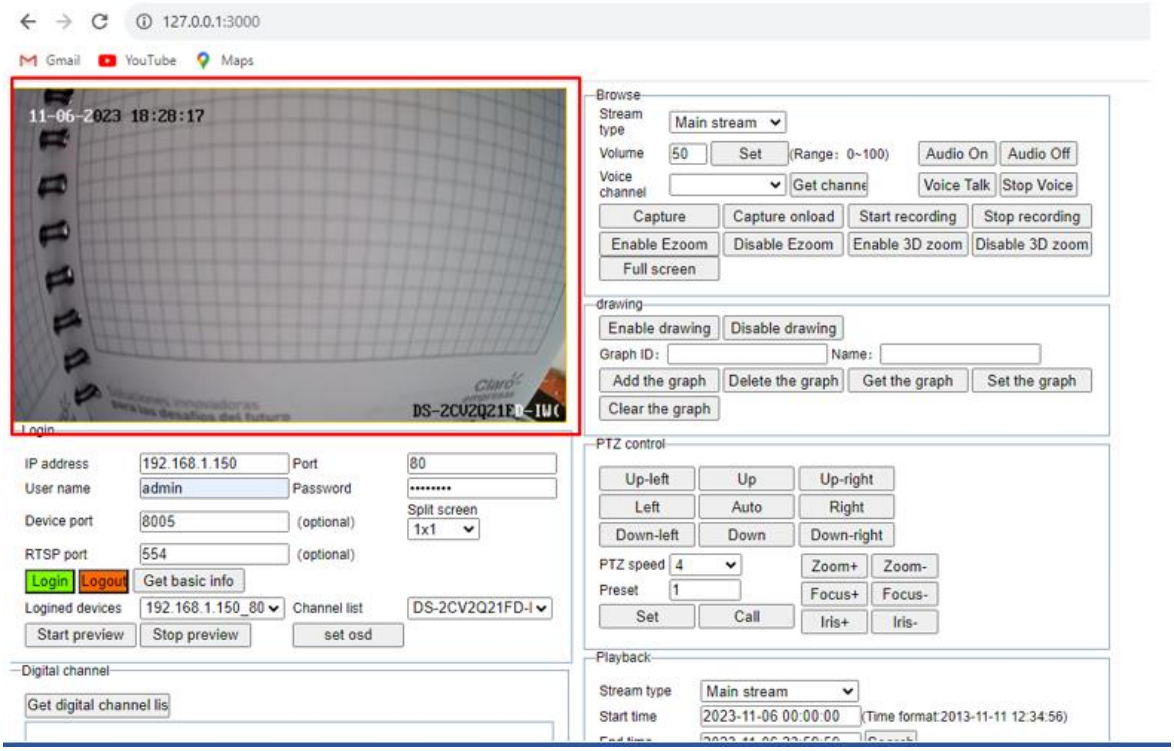
Fig. 14: Directorio web de la ruta de acceso login



Nota. Dirección de directorio correspondiente al acceso de credenciales. Fuente: Acceso directo local

En la fig. 15 (Acceso login a través del SDK), se identifica el acceso a la página de inicio de sesión del dispositivo, mediante el acceso blockchain, sin directorios, solo el puerto específico sobre el cual corre la aplicación.

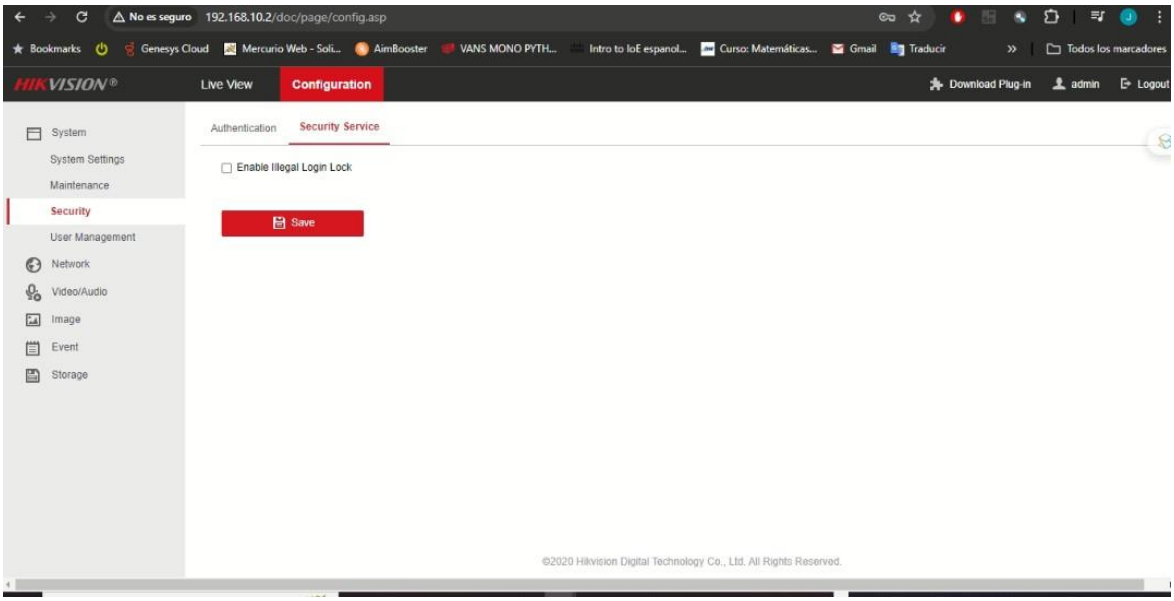
Fig. 15: Acceso login a través del SDK



Nota. Acceso al login del dispositivo mediante el SDK. Fuente: Acceso directo al equipo localmente.

En la fig. 16 siguiente (Parámetro de bloqueo de sesión), se identifica una de las configuraciones de seguridad por defecto, la cual permite bloquear múltiples intentos de sesión fallidos.

Fig. 16: Parámetro de bloqueo de sesión



Nota. Parámetro de bloqueo de múltiples intentos de sesión fallidos. Fuente: propia

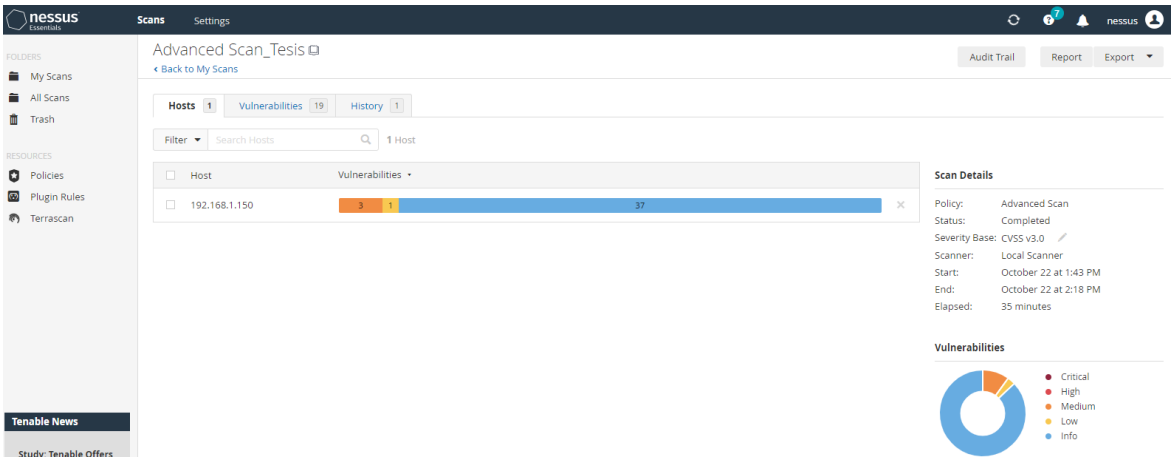
Entre las vulnerabilidades encontradas en la CVE se relaciona **CVE-2021-36260,** esta cuenta con una calificación de 9.8, relaciona una detección de vulnerabilidad de inyección de comandos en el servidor web de un producto de Hikvision. Debido a la falta de una adecuada validación de entrada, un atacante podría explotar esta debilidad para llevar a cabo un ataque de inyección de comandos mediante el envío de mensajes que contengan instrucciones maliciosas [40].

Con el equipo obtenido se creó un laboratorio práctico en un ambiente de hogar (local, lan), interconectando los equipos a la red, este ambiente local es debido a tener un entorno controlado, accesible y sin intervenir o afectar a un usuario final. De forma opcional y curiosa en los escenarios creados se iniciaron varias pruebas con Nessus esencial, Nmap, validando que otras vulnerabilidades se identificaban.

Mediante la herramienta de Nessus, instalada sobre Kali Linux versión 2023, se ejecuta escaneo avanzado sobre el dispositivo, el cual arrojó como resultado 19

vulnerabilidades 3 con calificación media y una baja, el restante son informativas. Entre las encontradas se encuentran evidencias sobre HTTP, SSL, TLS, enumeración de plataforma y datos expuestos del dispositivo, a continuación, se ilustran resultados en la imagen, para mayor detalle dirigirse al informe anexo (Advanced Scan_Tesis_hucx2i).

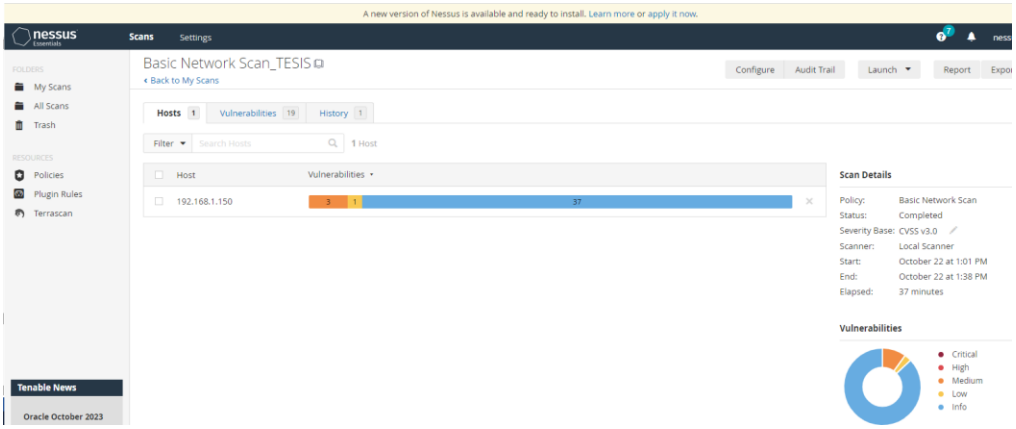
Fig. 17: Escaneo Avanzado sobre dispositivo IoT



Nota. Escaneo avanzado a dispositivo IoT cámara hikvision. Fuente: [41]

Adicional al escaneo anterior se realizó un escaneo de red básico el cual arrojó los mismos resultados del primer escaneo, como se identifica en la imagen a continuación.

Fig. 18: Escaneo de red básico sobre dispositivo IoT



Nota. Escaneo de red básico a dispositivo IoT cámara hikvision. Fuente: [41]

A través de Nmap se realizaron escaneos sobre el dispositivo, donde se obtuvieron resultados adicionales.

Fig. 19: Escaneo puertos abiertos mediante

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sS -T4 -p- 192.168.1.150
[sudo] contraseña para kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-11-06 01:53 CET
Nmap scan report for 192.168.1.150
Host is up (0.031s latency).
Not shown: 65527 closed tcp ports (reset)
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https
554/tcp   open  rtsp
8005/tcp  open  mxi
9010/tcp  open  sdr
9020/tcp  open  tambora
33925/tcp open  unknown
49152/tcp open  unknown
MAC Address: 30:1B:97:97:FA:A1 (Lierda Science & Technology Group)

Nmap done: 1 IP address (1 host up) scanned in 48.62 seconds
```

Nota. Comando de Nmap para revisar estado y puertos abiertos. Fuente: Captura propia.

2.2.3 Riesgos consolidados.

Esta actividad relacionó las dos actividades anteriores, pero de forma más complementaria, en la cual se incluyen riesgos adicionales y su calificación arrojada por CVSS.

El Sistema de Puntuación de Vulnerabilidad Común (CVSS) recoge las principales características técnicas de las vulnerabilidades en software, hardware y firmware. Proporciona puntuaciones numéricas que indican la gravedad de una vulnerabilidad en comparación con otras. CVSS consta de tres conjuntos de métricas: base, temporal y ambiental. La puntuación base muestra la gravedad intrínseca de una vulnerabilidad, mientras que las métricas temporales y ambientales ajustan esta según factores que varían con el tiempo y el entorno informático específico. Generalmente, las puntuaciones base las emite la entidad responsable del producto vulnerable o un tercero en su nombre, ya que son estables a lo largo del tiempo. Sin embargo, los usuarios de CVSS deben complementar estas puntuaciones con las temporales y ambientales específicas de su entorno para una evaluación más precisa. CVSS se utiliza como parte de la gestión de vulnerabilidades organizacionales, aunque otros factores como el número de usuarios afectados, pérdidas monetarias o percepción pública también influyen en las decisiones de remediación. Los beneficios de CVSS incluyen la estandarización y transparencia en la evaluación de vulnerabilidades, independientemente del proveedor o la plataforma [42]

Como se ha explicado previamente, las métricas de explotabilidad describen las características de lo que puede ser vulnerado, formalmente denominado componente vulnerable. Por ende, cada una de las métricas de explotabilidad detalladas a continuación debe evaluarse en relación con dicho componente vulnerable y debe reflejar las propiedades de la vulnerabilidad que facilitan un ataque exitoso.

Al evaluar las métricas base, se parte del supuesto de que el atacante posee un conocimiento avanzado sobre las debilidades del sistema objetivo, lo que incluye la configuración general y los mecanismos de defensa predefinidos (como los firewalls integrados, los límites de velocidad y la supervisión del tráfico). Por ejemplo, aunque la explotación de una vulnerabilidad pueda resultar en un éxito predecible y repetible, se

consideraría de baja complejidad para el ataque, sin importar el conocimiento o las habilidades del atacante. Además, cualquier medida de mitigación de ataques específicos dirigidos a un objetivo particular (como los filtros de firewall personalizados o las listas de acceso) debería ser reflejada en el conjunto de métricas ambientales de puntuación.

A continuación, se describen los Parámetros de la métrica base.

2.2.3.1 Vector de ataque (AV)

- Red (norte): El componente vulnerable está vinculado a la pila de red y el conjunto de posibles atacantes se extiende más allá de las otras opciones enumeradas a continuación, hasta incluir todo Internet. Esta vulnerabilidad suele denominarse "explotable remotamente" y puede considerarse como un ataque que puede explotarse *a nivel de protocolo* a uno o más saltos de red (por ejemplo, a través de uno o más enrutadores). Un ejemplo de ataque de red es un atacante que provoca una denegación de servicio (DoS) al enviar un paquete TCP especialmente diseñado a través de una red de área amplia (por ejemplo, CVE-2004-0230).
- Adyacente (A): El componente vulnerable está vinculado a la pila de red, pero el ataque se limita *a nivel de protocolo* a una topología lógicamente adyacente. Esto puede significar que un ataque debe lanzarse desde la misma red física compartida (p. ej., Bluetooth o IEEE 802.11) o lógica (p. ej., subred IP local), o desde dentro de un dominio administrativo seguro o limitado (p. ej., MPLS, VPN segura para una zona de red administrativa). Un ejemplo de un ataque adyacente sería una inundación de ARP (IPv4) o de descubrimiento de vecinos (IPv6) que provocaría una denegación de servicio en el segmento de LAN local (por ejemplo, CVE-2013-6014).
- Locales (L): El componente vulnerable no está vinculado a la pila de red y la ruta del atacante es a través de capacidades de lectura/escritura/ejecución. Cualquiera:

  el atacante explota la vulnerabilidad accediendo al sistema de destino localmente (por ejemplo, teclado, consola) o de forma remota (por ejemplo, SSH); *o*

el atacante depende de la interacción del usuario por parte de otra persona para realizar las acciones necesarias para explotar la vulnerabilidad (por ejemplo, utilizar técnicas de ingeniería social para engañar a un usuario legítimo para que abra un documento malicioso).

- Físico (P): El ataque requiere que el atacante toque o manipule físicamente el componente vulnerable. La interacción física puede ser breve (p. ej., ataque de sirvienta malvada [1]) o persistente. Un ejemplo de este tipo de ataque es un ataque de arranque en frío en el que un atacante obtiene acceso a las claves de cifrado del disco después de acceder físicamente al sistema de destino. Otros ejemplos incluyen ataques a periféricos a través de FireWire/USB Direct Memory Access (DMA).

2.2.3.2 Complejidad del ataque (AC)

- Bajo (L): No existen condiciones de acceso especializado ni circunstancias atenuantes. Un atacante puede esperar un éxito repetible al atacar el componente vulnerable.
- Alto (Alto): Un ataque exitoso depende de condiciones fuera del control del atacante. Es decir, un ataque exitoso no se puede lograr a voluntad, sino que requiere que el atacante invierta una cantidad mensurable de esfuerzo en preparación o ejecución contra el componente vulnerable antes de que se pueda esperar un ataque exitoso. [2] Por ejemplo, un ataque exitoso puede depender de que un atacante supere cualquiera de las siguientes condiciones:

  El atacante debe recopilar conocimientos sobre el entorno en el que existe el objetivo/componente vulnerable. Por ejemplo, un requisito para recopilar detalles sobre los ajustes de configuración de destino, números de secuencia o secretos compartidos.

  El atacante debe preparar el entorno objetivo para mejorar la confiabilidad del exploit. Por ejemplo, explotación repetida para ganar una condición de carrera o superar técnicas avanzadas de mitigación de exploits.

El atacante debe insertarse en la ruta lógica de la red entre el objetivo y el recurso solicitado por la víctima para poder leer y/o modificar las comunicaciones de la red (por ejemplo, un ataque de hombre en el medio).

2.2.3.3 Privilegios requeridos (PR)

- Ninguno (N): El atacante no está autorizado antes del ataque y, por lo tanto, no requiere ningún acceso a la configuración o archivos del sistema vulnerable para llevar a cabo un ataque
- Bajo (L): El atacante requiere privilegios que proporcionen capacidades básicas de usuario que normalmente podrían afectar solo a la configuración y los archivos propiedad de un usuario. Alternativamente, un atacante con privilegios bajos tiene la capacidad de acceder sólo a recursos no confidenciales.
- Alto (Alto): El atacante requiere privilegios que proporcionen un control significativo (por ejemplo, administrativo) sobre el componente vulnerable, permitiendo el acceso a configuraciones y archivos de todo el componente.

2.2.3.4 Interacción del usuario (UI)

- Ninguno (N): El sistema vulnerable puede explotarse sin interacción por parte de ningún usuario.
- Requerido (R): La explotación exitosa de esta vulnerabilidad requiere que el usuario realice alguna acción antes de que se pueda explotar la vulnerabilidad. Por ejemplo, es posible que un exploit exitoso solo sea posible durante la instalación de una aplicación por parte de un administrador del sistema.

2.2.3.5 Alcance (S)

- Sin cambios (U): Una vulnerabilidad explotada sólo puede afectar a los recursos gestionados por la misma autoridad de seguridad. En este caso, el componente

vulnerable y el componente afectado son el mismo o ambos están administrados por la misma autoridad de seguridad.

- Cambiado (C): Una vulnerabilidad explotada puede afectar recursos más allá del alcance de seguridad administrado por la autoridad de seguridad del componente vulnerable. En este caso, el componente vulnerable y el componente impactado son diferentes y gestionados por diferentes autoridades de seguridad.

2.2.3.6 Confidencialidad (C)

- Alto (Alto): Se produce una pérdida total de confidencialidad, lo que da como resultado que todos los recursos del componente afectado se divulguen al atacante. Alternativamente, se obtiene acceso sólo a cierta información restringida, pero la información divulgada presenta un impacto directo y grave. Por ejemplo, un atacante roba la contraseña del administrador o las claves de cifrado privadas de un servidor web.
- Bajo (L): Hay cierta pérdida de confidencialidad. Se obtiene acceso a cierta información restringida, pero el atacante no tiene control sobre qué información se obtiene, o la cantidad o tipo de pérdida es limitada. La divulgación de información no causa una pérdida directa y grave al componente afectado.
- Ninguno (N): No hay pérdida de confidencialidad dentro del componente afectado.

2.2.3.7 Integridad

- Alto (Alto): Hay una pérdida total de integridad o una pérdida completa de protección. Por ejemplo, el atacante puede modificar cualquiera o todos los archivos protegidos por el componente afectado. Alternativamente, sólo se pueden modificar algunos archivos, pero una modificación maliciosa presentaría una consecuencia directa y grave para el componente afectado.
- Bajo (L): La modificación de datos es posible, pero el atacante no tiene control sobre las consecuencias de una modificación o la cantidad de modificación es limitada. La

modificación de datos no tiene un impacto directo y grave en el componente afectado.

- Ninguno (N): No hay pérdida de integridad dentro del componente afectado.

2.2.3.8 Disponibilidad (A)

- Alto (Alto): Hay una pérdida total de disponibilidad, lo que hace que el atacante pueda denegar por completo el acceso a los recursos del componente afectado; esta pérdida es sostenida (mientras el atacante continúa lanzando el ataque) o persistente (la condición persiste incluso después de que el ataque se haya completado). Alternativamente, el atacante tiene la capacidad de negar cierta disponibilidad, pero la pérdida de disponibilidad presenta una consecuencia directa y grave para el componente afectado (por ejemplo, el atacante no puede interrumpir las conexiones existentes, pero puede impedir nuevas conexiones; el atacante puede explotar repetidamente una vulnerabilidad que, en cada caso de un ataque exitoso, pierde sólo una pequeña cantidad de memoria, pero después de una explotación repetida hace que un servicio deje de estar disponible por completo).
- Bajo (L): El rendimiento se reduce o hay interrupciones en la disponibilidad de recursos. Incluso si es posible explotar repetidamente la vulnerabilidad, el atacante no tiene la capacidad de negar completamente el servicio a los usuarios legítimos. Los recursos en el componente afectado están parcialmente disponibles todo el tiempo o completamente disponibles solo una parte del tiempo, pero en general no hay consecuencias directas y graves para el componente afectado.
- Ninguno (N): No hay ningún impacto en la disponibilidad dentro del componente afectado.

En la siguiente tabla #6 (escala de calificación cualitativa CVSS) se representa una escala de valores cualitativos, tomados de la CVSS, con rangos que van desde 0,0 hasta 10,0. En ciertos contextos, resulta práctico contar con una descripción en texto de las puntuaciones

numéricas obtenidas en las categorías de Base, Amenaza y Entorno del CVSS. Todas las puntuaciones CVSS, sin importar su etiqueta específica, pueden ser asociadas con evaluaciones cualitativas, a continuación, se describe la clasificación:

Nulo: Se refiere a un riesgo que tiene una probabilidad muy baja de ocurrencia y un impacto insignificante. Este nivel de riesgo generalmente se considera aceptable y no requiere acciones de mitigación adicionales.

Bajo: Indica un riesgo con una probabilidad baja de ocurrencia y un impacto limitado. Aunque puede no ser una preocupación inmediata, se recomienda monitorear este tipo de riesgo y considerar medidas preventivas para reducir su probabilidad de ocurrencia o mitigar sus posibles impactos.

Medio: Se refiere a un riesgo con una probabilidad moderada de ocurrencia y un impacto significativo. Estos riesgos requieren una atención especial y la implementación de medidas de mitigación adecuadas para reducir tanto su probabilidad como su impacto.

Alto: Indica un riesgo con una probabilidad alta de ocurrencia y un impacto considerable. Estos riesgos representan una amenaza significativa y requieren una acción inmediata para mitigar su impacto o reducir su probabilidad de ocurrencia.

Crítico: Se refiere a un riesgo con una probabilidad muy alta de ocurrencia y un impacto catastrófico. Estos riesgos representan una amenaza existencial y requieren una acción inmediata y exhaustiva para su mitigación.

Tabla 6. Escala de calificación cualitativa CVSS

Puntuación	clasificación
0,0	Nulo - Ninguno
0,1 – 3,9	Bajo
4,0 – 6,9	Medio
7,0 – 8,9	Alto
9,0 – 10,0	Critico

Nota. Relación de la escala de gravedad cualitativa de acuerdo con el CVSS. Fuente: [42]

En la tabla 7. (Riesgos consolidados) se relaciona el activo, las vulnerabilidades y riesgos del activo, una columna que relaciona el scoring resultante con numeración CVE, y

riesgos sin asignación de una numeración CVE, las cuales se realizaron con un cálculo a través de la herramienta de CVSS V3.1, obteniendo de igual forma un scoring. Esta valoración fue realizada por el autor, siguiendo los pasos secuenciales de la herramienta antes mencionada, con el fin de lograr tener la métrica base del grado de criticidad desde lo que el CVSS establece, generando un mecanismo de visualización de la gravedad de las vulnerabilidades.

En un análisis de riesgo generalmente se parte de analizar tanto la probabilidad como el impacto de manera cuantitativa para determinar el nivel de riesgo. Así mismo el CVSS ya lleva implícito una valoración de distintos atributos de la vulnerabilidad para lograr un scoring.

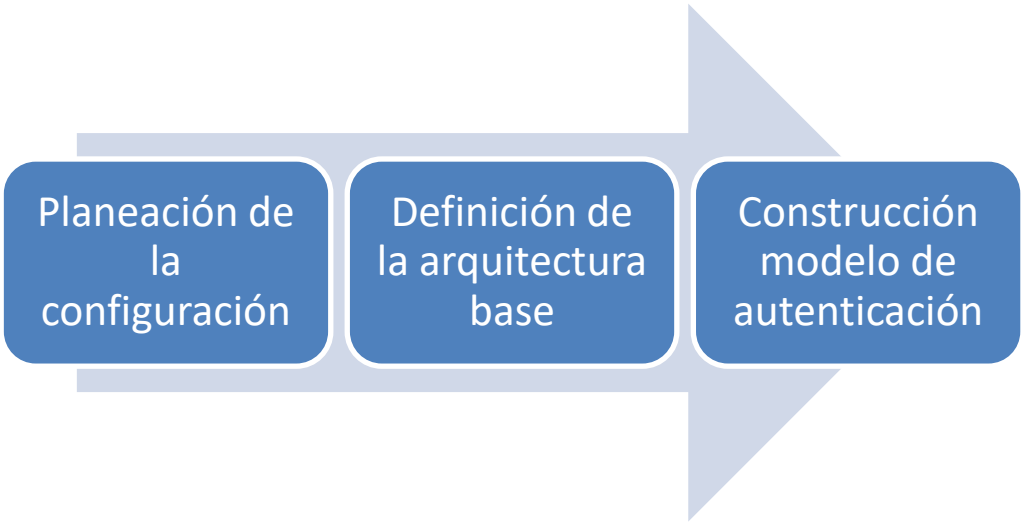
Tabla 7. Riesgos consolidados

ACTIVO	VULNERABILIDAD	BASE	RIESGO
Cámara IoT	CVE-2017-7921 Autenticación incorrecta	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H	Critico
	CVE-2017-7923 Contraseña en el archivo de configuración.	CVSS:3.0/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	Alto
	Acceso no autorizado	AV:N/AC:H/PR:H/UI:R/S:U/C:H/I:L/A:L	Medio
	Ataque de fuerza bruta	AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	Alto
	Ataque de Reenvío de Contraseña	AV:A/AC:H/PR:L/UI:R/S:U/C:H/I:L/A:L	Medio

Nota. Tabla de relación de riesgos consolidados, algunos de ellos con vulnerabilidad asociada con asignación de la CVE, el restante cuenta con scoring de acuerdo con el CVSS. Fuente: Diseño propio.

2.3 Fase 3 Planeación de la configuración.

Fig. 20: Esquema de actividades referente a la fase 3



Nota. Actividades relacionadas en ejecución a la fase 3. Fuente: diseño propio

Esta fase incluyó la definición de la arquitectura, y un modelo de autenticación, que llevan a cabo la conexión del dispositivo. El diseño del modelo de blockchain ofrece una serie de atributos que pueden contribuir a mitigar las brechas de seguridad en la autenticación de las cámaras de seguridad IoT al garantizar la integridad, confiabilidad y transparencia de los registros de autenticación.

Inicialmente la descentralización de la información genera ya un nivel de seguridad, implica que los registros de autenticación se almacenan y verifican en múltiples nodos de la red, esto reduce el riesgo de un único punto de falla y hace que sea más difícil para los atacantes comprometer o falsificar los registros de autenticación. Una vez que se registra la transacción de autenticación en la cadena de bloques, esta información es inmutable y no puede ser alterada ni eliminada. Esto garantiza que los registros de autenticación de las cámaras IoT sean confiables y a prueba de manipulaciones.

Con base en los objetivos anteriores y análisis realizados, se procedió con el esquema de configuración, que permiten la mitigación de los riesgos y vulnerabilidades registradas,

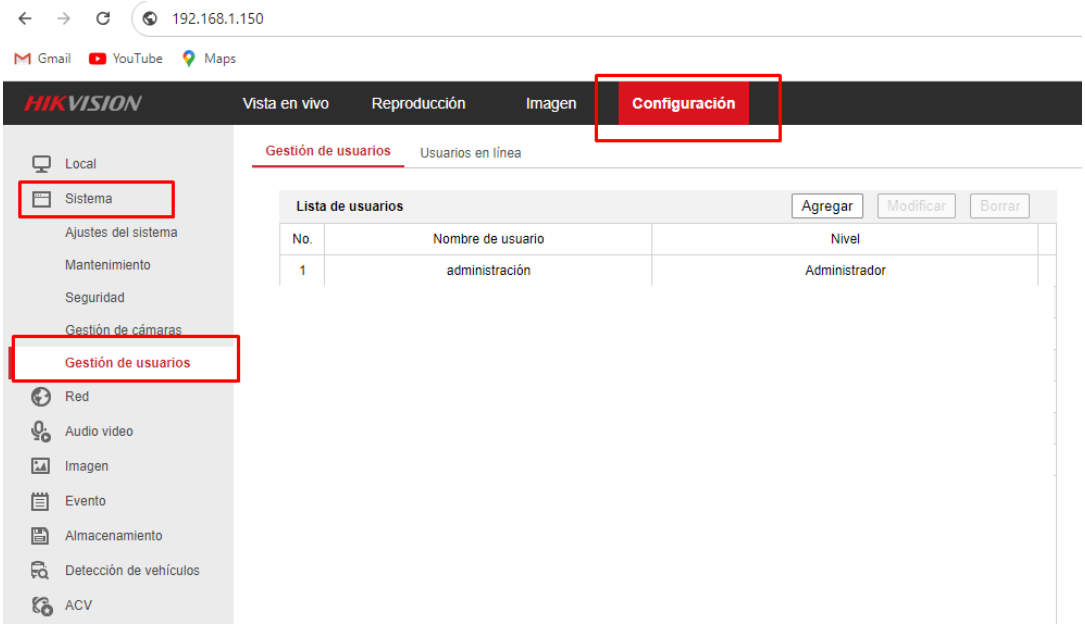
siendo trabajados en el modelo de autenticación mediante blockchain Ethereum, este más robusto directamente proporcional a más cantidad de nodos, es decir, entre más nodos contenga una cadena de bloques, mayor es la dificultad de ser vulnerable, lo que genera mayor seguridad a la transacción y a la conexión del dispositivo IoT. Siguiendo esta línea y para seguir con el desarrollo, se debe tener en cuenta el funcionamiento de la Blockchain Ethereum con el dispositivo IoT y que gracias a su lenguaje de programación permite gestionar y automatizar resultados:

Registro de usuario:

Teniendo en cuenta en las pruebas realizadas, no se logró crear el usuario en el dispositivo de forma automática, este se debe registrar siempre al igual que la contraseña directa y manualmente en el dispositivo IoT, antes de pasar y de hacer las solicitudes mediante la blockchain. Se ingresa al dispositivo directamente sobre la dirección IP asignada, con el usuario administrador previamente activado, en el menú configuración, sistema, administración de cuentas (gestión de usuarios), se crea el usuario o se edita el predeterminado (admin).

La imagen a continuación Fig.21 (registro de usuario en dispositivo IoT) ilustra el menú y ruta donde se crea el usuario que tendrá acceso a la cámara.

Fig. 21: Registro de usuario en dispositivo IoT



Nota. Inicio de sesión y creación de usuario directo en el dispositivo IoT a través del navegador web. Fuente: captura propia.

Autenticación de usuario en la Blockchain

Después de contar con el usuario y contraseña creado sobre el dispositivo IoT, el usuario mediante un navegador web y con el plugin previamente instalado (HCWebSDKPlugin.exe), hace el ingreso al servidor web a través de la dirección IP donde se encuentra instalada la blockchain, seguido del puerto designado (3000). al cargar el entorno gráfico, estos salen en blanco, sin ningún parámetro por seguridad. El usuario cuenta con los campos del dispositivo IoT: dirección IP, puerto, usuario y contraseña, después de diligenciar los campos el usuario debe presionar el botón login, en ese instante las credenciales pasan por el SDK mediante la ayuda de express y node.js, enviando posteriormente los datos a hardhat.js que es el que registra y trata los datos del usuario, a través de la ayuda de Ethers, para poder mantenerlo en la Blockchain, este finalmente entrega los datos al dispositivo IoT, permitiendo la interacción del video sobre el entorno grafico dispuesto por el SDK.

Los datos usados por el usuario, ya se encuentran creados previamente en la Blockchain, mediante los códigos de programación, lo que indica que si se crea un nuevo usuario y contraseña en el dispositivo IoT se deben de registrar también en la Blockchain, de esta manera un usuario que pase por la Blockchain y no esté previamente registrado, no se procesará la información, de esta manera se pretende trabajar entre los riesgos consolidados:

Autenticación incorrecta: Un modelo de autenticación basado en blockchain puede utilizar firmas digitales o claves criptográficas para verificar la identidad de los usuarios de manera más segura, en la interacción con el dispositivo se trabaja un cifrado SHA256, si el usuario intenta iniciar sesión de forma errada durante los primeros cisco minutos, la integración cuenta con un bloqueo de múltiples intentos de sesión. Si el usuario intenta iniciar sesión después de los cinco minutos y las credenciales son erradas, la blockchain no genera la transacción, desde la entrada descarta el paquete.

Contraseña en el archivo de configuración: En un modelo de autenticación basado en blockchain, no hay necesidad de almacenar contraseñas en archivos de configuración sobre directorios, las contraseñas no se almacenan en texto plano. En su lugar, se almacenan como hashes. Un hash es una representación única de la contraseña original que no se puede revertir fácilmente para obtener la contraseña original.

Acceso no autorizado: Utilizando un blockchain, las transacciones de autenticación podrían ser registradas y verificadas de manera transparente en la cadena de bloques, lo que dificultaría el acceso no autorizado a recursos protegidos.

Ataque de fuerza bruta: La naturaleza descentralizada y resistente a la modificación de un blockchain como Ethereum hace que sea difícil realizar estos ataques, inicialmente si el usuario no existe en la cadena de bloques este no genera ninguna transacción, omite el paquete, adicional con la ayuda del SDK, se establece un bloqueo de tiempo por múltiples intentos de sesión, teniendo en cuenta el funcionamiento anterior mencionado a los 5 minutos de ser iniciado el dispositivo.

Ataque de Reenvío de Contraseña: En un sistema de autenticación basado en blockchain, las transacciones se registran de forma inmutable en la cadena de bloques, lo que dificulta la posibilidad de reenviar contraseñas de manera malintencionada sin el conocimiento del usuario.

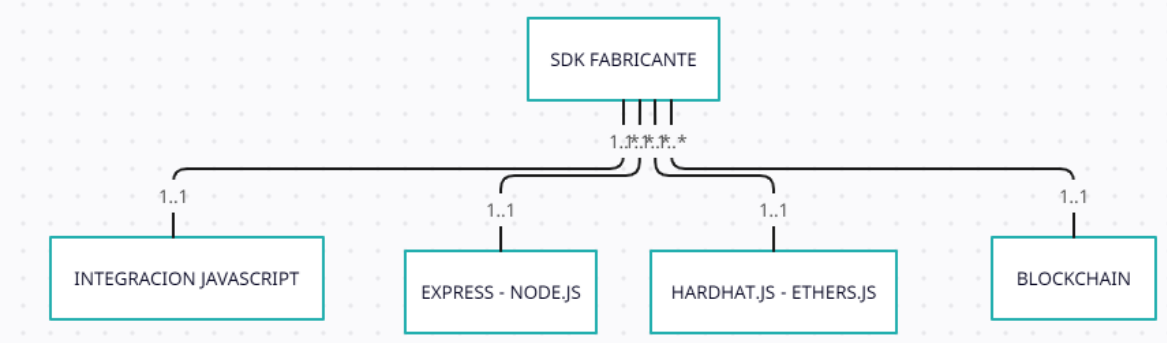
2.3.1 Definición de la arquitectura base

En esta actividad se planteó y dispuso de una arquitectura base sobre la cual se centra el desarrollo de las actividades siguientes, conformadas por la cadena de bloques y herramientas necesarias para la ejecución de este.

Esta arquitectura se centra en la integración de un sistema compuesto por el dispositivo seleccionado, router o modem de conexión, blockchain, desarrollo JavaScript para integración y usuario de consulta que consume el servicio.

Es importante resaltar que, durante la investigación, no fue posible crear el usuario directamente desde la blockchain, este ya debe estar creado en el dispositivo IoT.

Fig. 22: Arquitectura base de integración

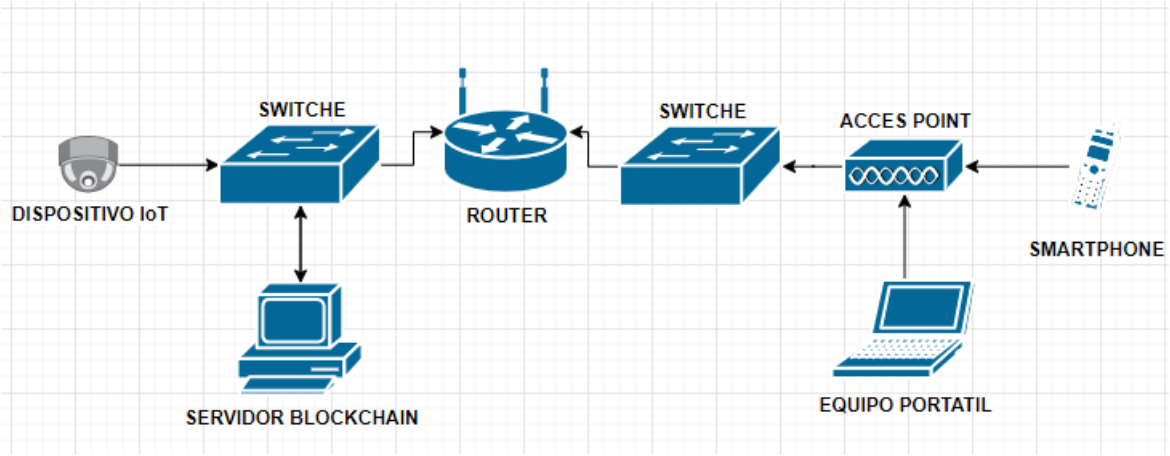


Nota. Arquitectura de la base de integración entre códigos bases y aplicaciones. Fuente: Diseño propio

El diagrama a continuación simulado sobre Draw.io. desde su página en línea, ilustra los equipos necesarios para el escenario que se propuso de forma local, debido al alto costo y variaciones de las transacciones directamente en la red blockchain Ethereum, la complejidad de lograr entrar en la cadena de bloques ya que requiere de mayor experiencia y conocimiento técnico, irreversibilidad en las configuraciones, y alto consumo de energía. De acuerdo con lo anterior se procedió con establecer un equipo de cómputo con sistema operativo Windows 10 profesional, en el cual se instaló una máquina virtual específica sobre el ambiente de prueba de Ethereum (Hardhad), un ambiente donde se simula una blockchain Ethereum con las librerías de Ethers y

complementos de plugin necesarios que permiten tener una blockchain local y configurable, con transacciones de valores a cero Ethers. De esta manera se despliega la Blockchain local comunicada con el resto de los dispositivos mediante la conexión de red local establecida con los switches y el router. Así las interacciones son dirigidas hacia el equipo local donde está la Blockchain y no hacia la red pública de Ethereum.

Fig. 23: Dispositivos necesarios en el escenario de pruebas



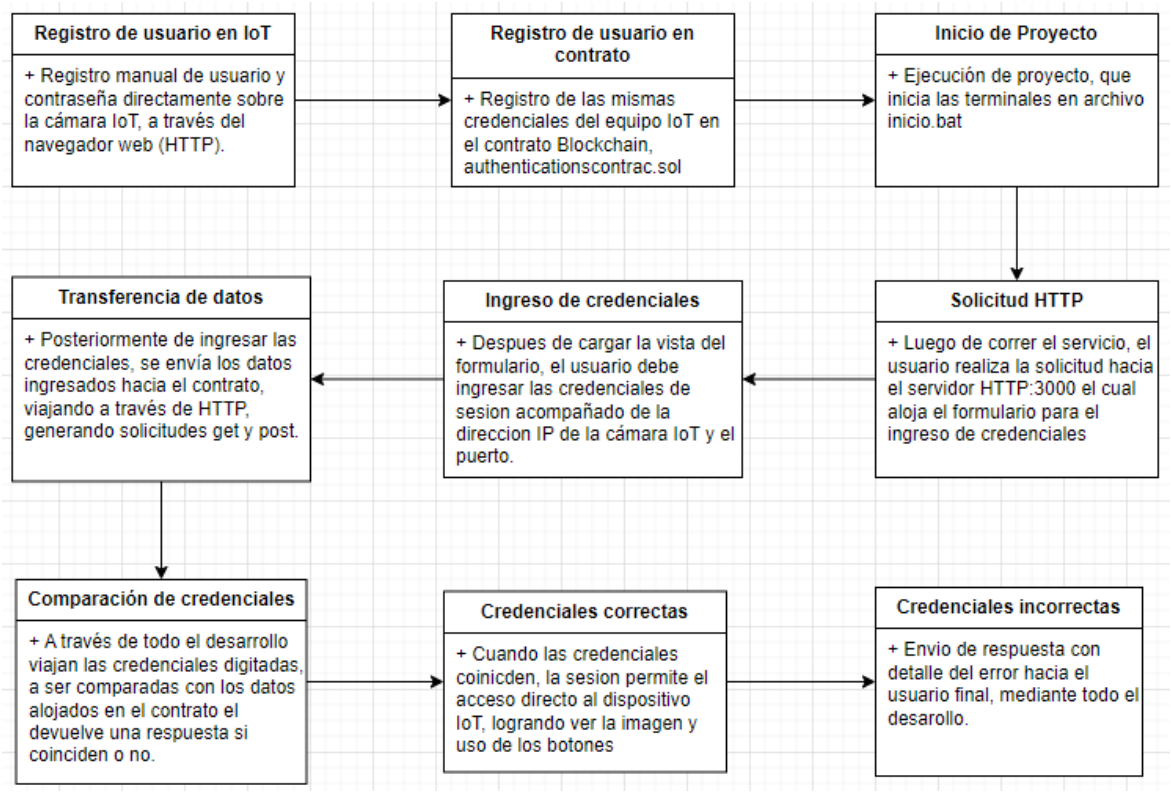
Nota. Dispositivos necesarios para la puesta en funcionamiento y pruebas del sistema propuesto. Fuente: Diseño propio

2.3.2 Construcción modelo de autenticación.

En esta actividad se inició la ejecución y desarrollo del modelo de autenticación a través de las herramientas definidas para Ethereum, interactuando con el SDK suministrado por el proveedor.

En esta actividad se inició la ejecución y desarrollo del modelo de autenticación a través de las herramientas definidas para Ethereum, bajo un contrato inteligente, interactuando con el SDK (kit de desarrollo de software) suministrado por el proveedor. La base del código esta sobre java script y parte en solidity necesario para la blockchain Ethereum y por último interacción con la herramienta Hardhad la cual es indispensable para el despliegue del contrato inteligente en Ethereum.

Fig. 24: Modelo de Autenticación



Nota. Modelo de autenticación en ciclo de comunicación entre los elementos y programas que interactúan con las peticiones del usuario, elaborado de acuerdo con la información recolectada en la fase 1, la tabla 4 y el manual del equipo, plataforma usada en línea en la página https://app.diagrams.net/?src=about. Fuente: Diseño propio.

Para llevar a cabo los registros y funcionamiento del desarrollo, se relacionan códigos relevantes:

Index.js: Responde al directorio raíz, despliega el contrato y responde las solicitudes get y post.

Demo.js: Contiene las instrucciones de iniciar el plugin, WebvideoCtrl.js, alista e interpreta en conjunto con las acciones de cada botón disponible en la vista.

AuthenticationContrac.sol: Contrato, funciones de autenticación de usuario, almacena y compara las credenciales almacenadas con las digitadas.

Hardhat: Es el entorno de desarrollo para el software de Ethereum. Consta de diferentes componentes para editar, compilar, depurar e implementar sus contratos

inteligentes y aplicaciones descentralizadas, todos los cuales trabajan en conjunto para crear un entorno de desarrollo completo.

Ethers.js: Librería que permite interactuar y automatizar las interacciones con el contrato, sobre la cual trabaja Hardhat. Librería instalada en node.modules.

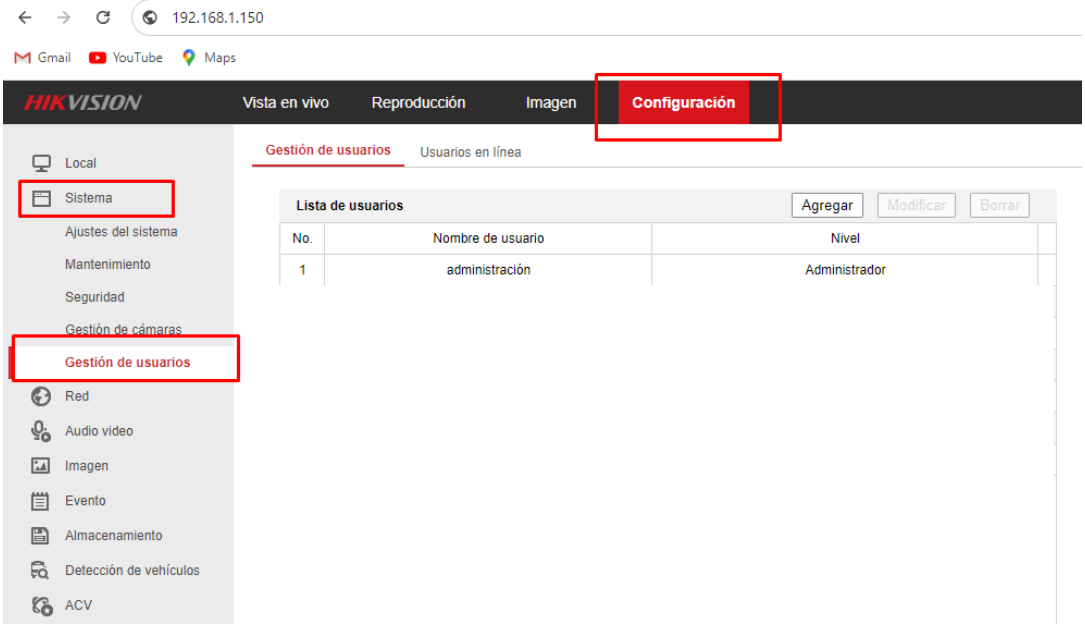
WebSDK: Específicamente para los productos HikVision® se dispone de un WebSDK(Botiquín de desarrollo de software Web) provisto por la compañía con el cual podemos realizar procesos de la cámara de forma automatizada, así se puede dirigir técnicamente el usuario hacia la cámara de manera indirecta, de modo que se puede controlar el comportamiento de la autenticación; El instalador se puede descargar desde la página del fabricante (https://www.hikvision.com/es-co/support/tools/hitools/cl6add977aed3eacb1/).

Express: Para establecer el punto de acceso a la cámara por la red, se ayuda de su librería que permite configurar un puerto de la PC para escuchar e interpretar solicitudes HTTP, es decir, leer las solicitudes, decodificarlas y decidir qué hacer para solucionarlas.

Registro de usuario y contraseña en el dispositivo IoT:

Se ingresa al dispositivo directamente sobre la dirección IP asignada (HTTP), con el usuario administrador previamente activado, en el menú configuración, sistema, administración de cuentas (gestión de usuarios), se crea el usuario o se edita el predeterminado (admin).

Fig. 25: Registro de usuario y contraseña en equipo IoT

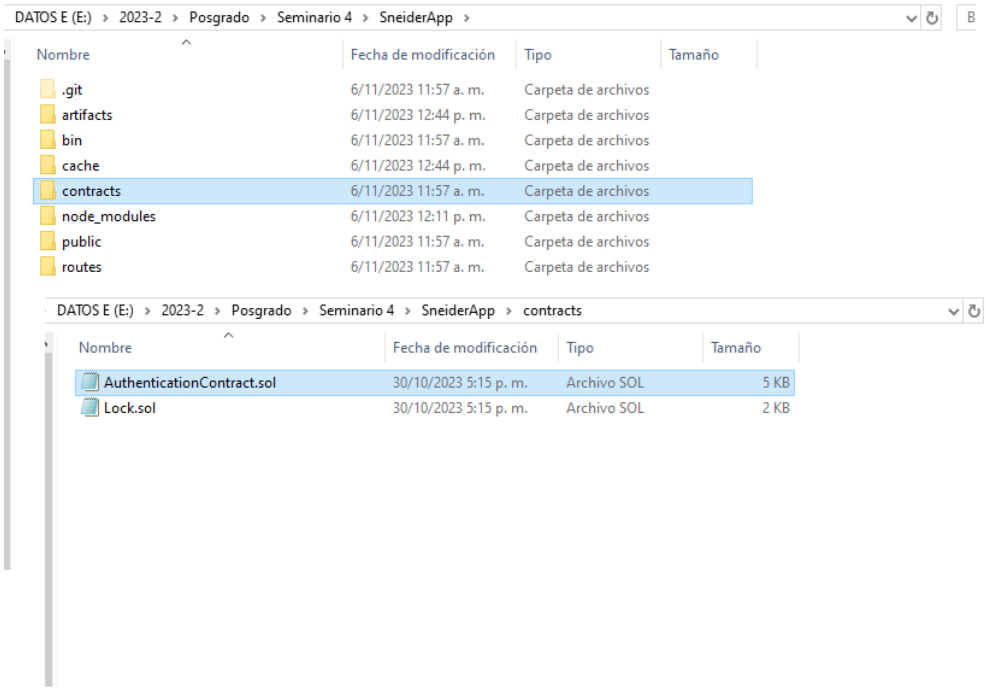


Nota. Acceso y ruta para creación de usuario y contraseña en dispositivo IoT. Fuente: Diseño propio

Registro de usuario y contraseña en el contrato

Luego de configurar el usuario y la contraseña en el dispositivo IoT, se replican los mismos datos en una variable interna encargada de almacenar las credenciales que el contrato debe conocer, del contrato inteligente, quemarlas directamente sobre el código, el cual se encuentra en \contracts\AuthenticationContract.sol.

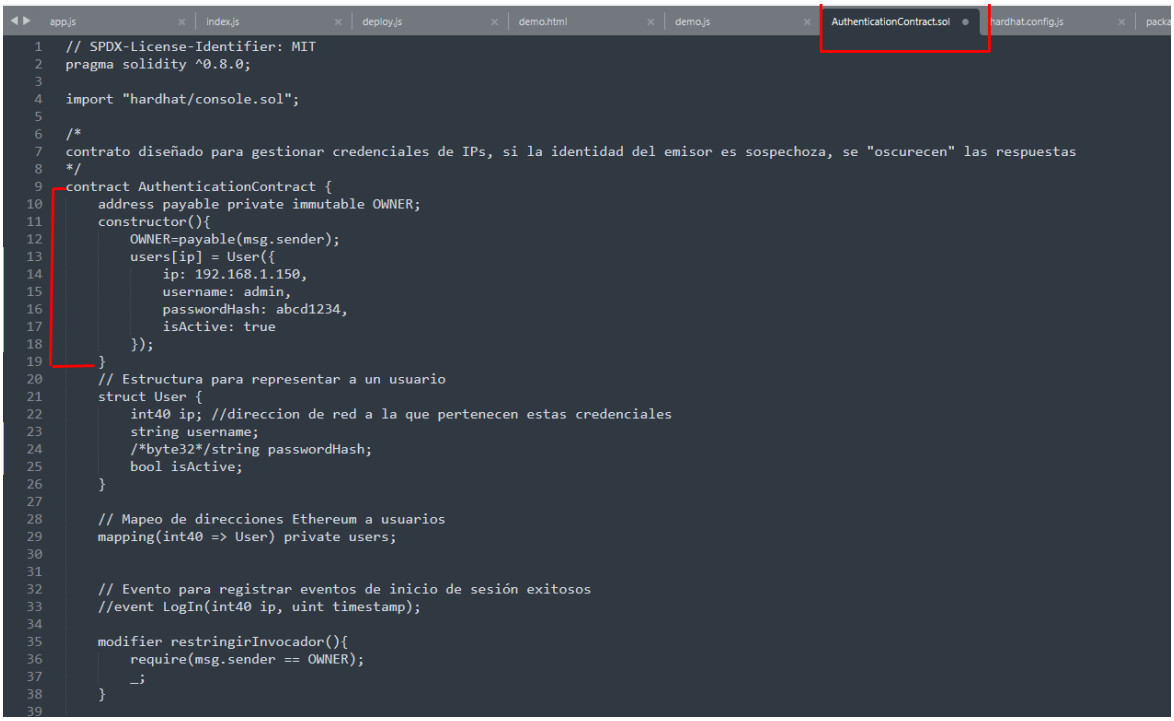
Fig. 26: Ruta de código de contrato



Nota. Acceso y ruta donde se encuentra el contrato que almacena las contraseñas de la Blockchain. Fuente: Diseño propio

Se abre el archivo y entre las líneas 09 y 18 del código, se registran las credenciales del dispositivo IoT.

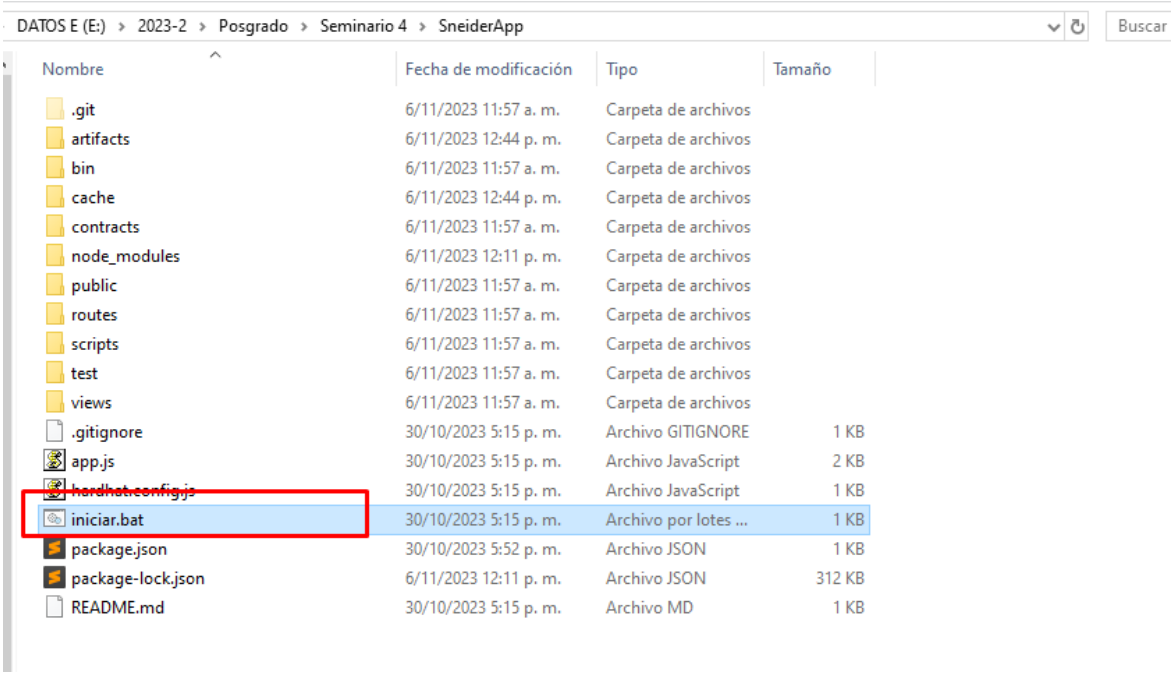
Fig. 27: Líneas de código usuario y contraseña en contrato



Nota. Acceso y líneas de código donde se encuentran las credenciales del contrato de la Blockchain. Fuente: Diseño propio.

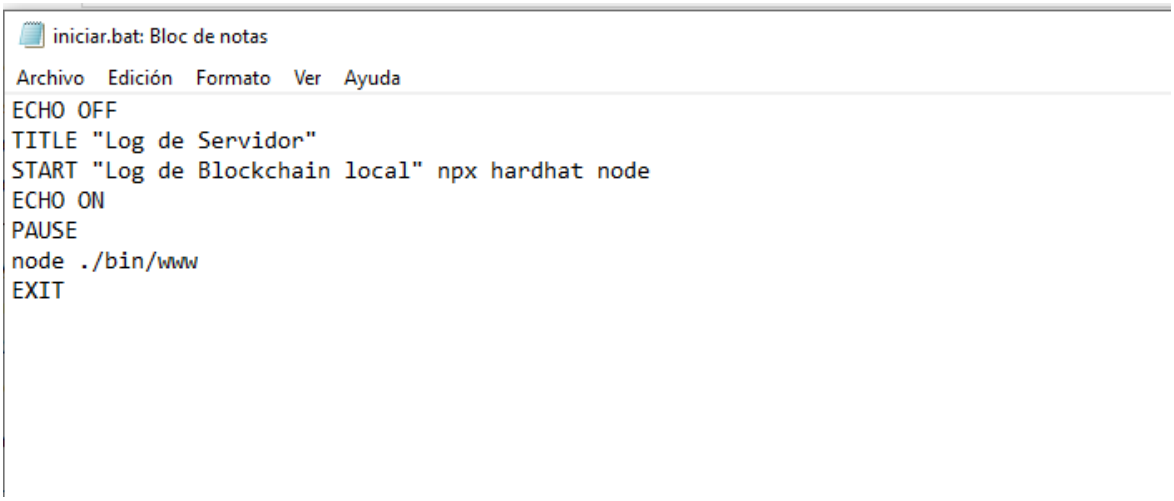
Con el usuario ya registrado en el dispositivo IoT y en el contrato, se procede a ejecutar el proyecto (iniciar.bat) servicio de express, mediante un scritp que recorre las rutas, abriendo una terminal del contrato y otra terminal del log de express, desencadenando posteriormente el resto del desarrollo.

Fig. 28: Ruta e inicio de proyecto



Nota. Ubicación del proyecto e inicio a través de script, iniciar.bat. Fuente: Diseño propio.

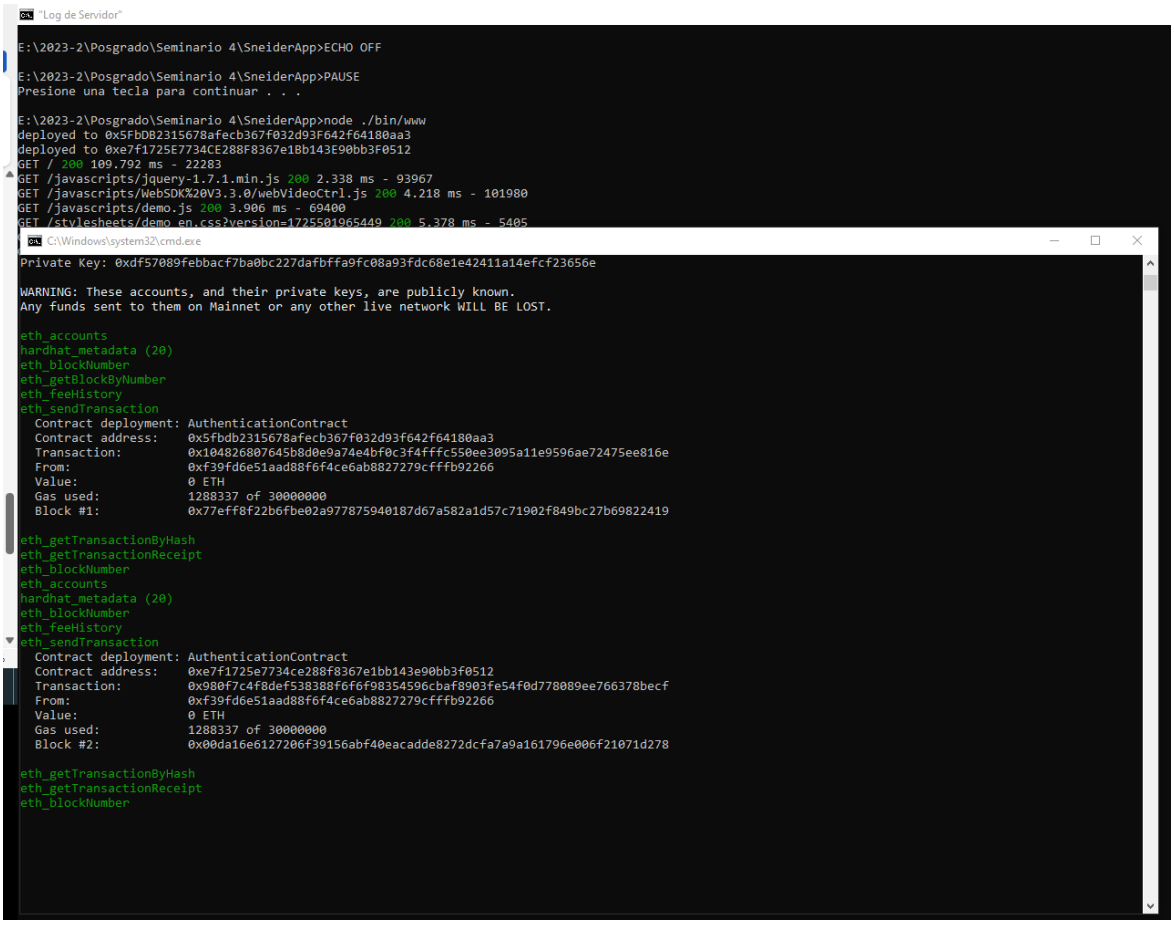
Fig. 29: Secuencia dentro del archivo iniciar.bat



Nota. Instrucciones que componen el archivo iniciar.bat. Fuente: Diseño propio.

Con el proyecto ya iniciado, y ejecutado el archivo iniciar.bat, se abren las terminales de registro del contrato desplegado.

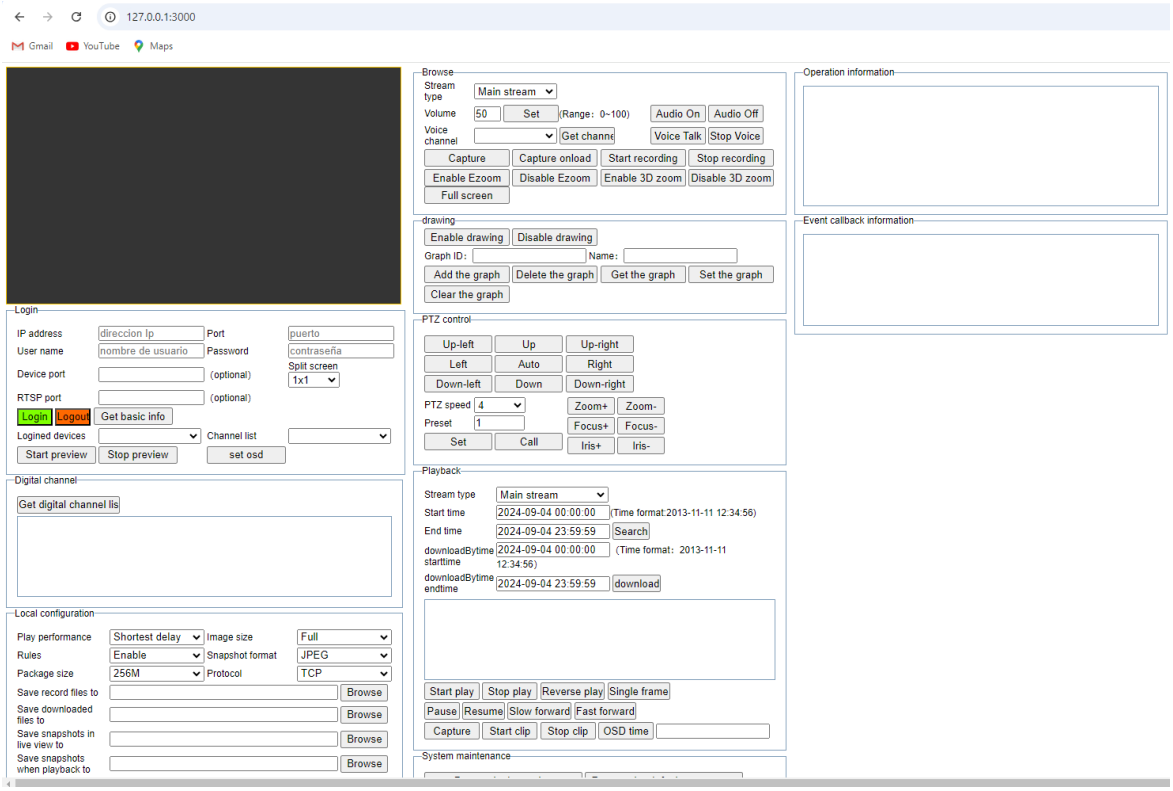
Fig. 30: Terminales de registro despliegue contrato



Nota. Ilustración de dos terminales donde se describe la ejecución del contrato. Fuente: Diseño propio.

Al tener el servicio corriendo, ya se habilita la vista HTTP (formulario), sobre localhost:3000 o la dirección del servidor donde está alojado los servicios más el puerto 3000. La cual recibe los datos de las credenciales del usuario.

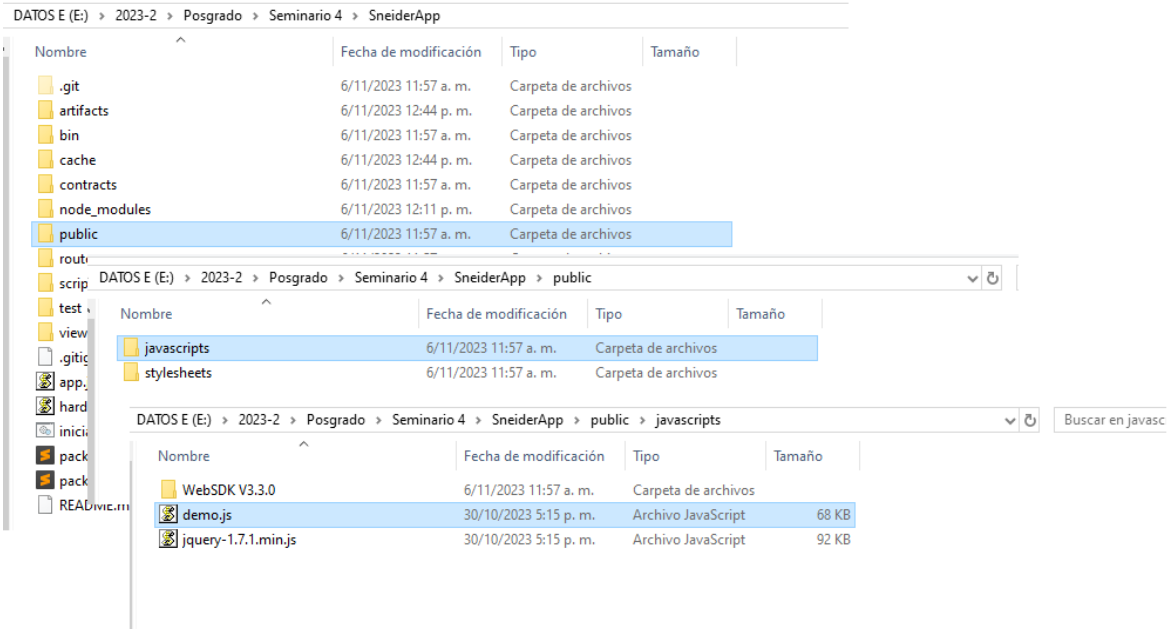
Fig. 31: Vista intermediario del proceso



Nota. Vista para el ingreso de las credenciales que van entre el dispositivo IoT y la Blockchain. Fuente: Diseño propio.

Esta vista es un intermediario en el proceso, el desarrollo se encuentra en la ruta (\public\javascripts\demo.js), funcionalidad gracias al uso del WebSDK instalado y descargado de la página del fabricante Hikvision (https://www.hikvision.com/es-co/support/tools/hitools/cl6add977aed3eacb1/). El cual permite ser instalado para provechar sus librerías y programar sus funciones, ubicado sobre:

Fig. 32: Ruta de acceso al archivo del código



Nota. Ruta donde se encuentra el archivo que contiene el código demo.js Fuente: Diseño propio.

Sobre el formulario, se deben de ingresar los datos del dispositivo IoT al cual se va a acceder, Usuario, contraseña, IP y puerto (si es default no es necesario, el puerto por defecto es el 80). Después de ingresar los datos, el usuario debe presionar el botón login.

Hasta este punto que ya hay un avance sobre las credenciales, puede surgir una pregunta y es ¿en qué momento sucede el guardado de las credenciales en la blockchain ?, pues bien, el guardado de las credenciales en la Blockchain ocurre durante la implementación del contrato inteligente, las credenciales (usuario, contraseña y dirección IP de la cámara) se queman en el código del contrato inteligente en la variable interna. Esto significa que las credenciales son insertadas como literales directamente dentro del código del contrato antes de su despliegue en la blockchain,

Una vez que el contrato inteligente esté listo (con las credenciales quemadas en el código), se despliega en la blockchain (en este caso, usando Hardhat en una blockchain local).

Al desplegar el contrato, todas las credenciales quemadas se almacenan permanentemente en la blockchain.

En esta prueba de concepto, las contraseñas se almacenan en texto claro directamente en el contrato inteligente. Esta decisión se tomó para simplificar y agilizar el desarrollo durante la fase de prueba, permitiendo una verificación más directa de las funcionalidades relacionadas con la autenticación y el acceso a los servicios.

Es importante destacar que esta forma de manejo de contraseñas no es la adecuada y la definida para un entorno de producción, ya que en una blockchain pública los datos son visibles para todos los participantes de la red. Al almacenar contraseñas en texto plano, se expone a los usuarios a posibles riesgos de seguridad, lo cual es inaceptable para una implementación real.

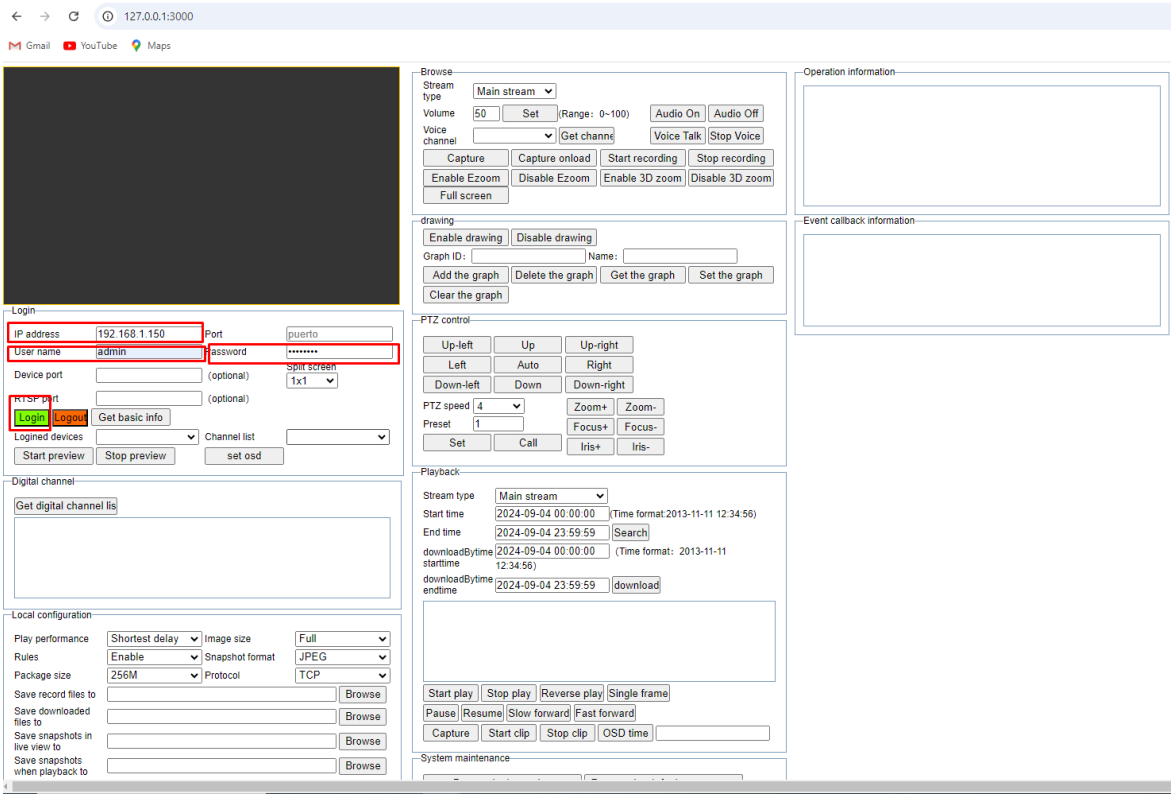
Para un entorno de producción, se debe implementar un enfoque seguro, utilizando algoritmos de hash para proteger las contraseñas. Esto implica:

1. Hashear las contraseñas antes de almacenarlas en la blockchain, utilizando algoritmos como Keccak256 o SHA-256. Un hash es una representación irreversible de la contraseña, por lo que, aunque el hash esté visible en la blockchain, no es posible derivar la contraseña original.

2. Comparar hashes en lugar de contraseñas en texto claro. Cuando un usuario intenta autenticarse, el sistema calcula el hash de la contraseña ingresada y compararlo con el hash previamente almacenado.

Teniendo un panorama ya más claro, sobre las credenciales, seguimos con el siguiente paso, sobre la vista que recopila los datos del usuario.

Fig. 33: Login de ingreso con credenciales



Nota. Ingreso de credenciales del usuario respecto al dispositivo IoT, para el uso del botón login. Fuente: Diseño propio.

Al usuario digitar login, sin diligenciar ningún dato, formulario en blanco, no realizará ninguna acción, de lo contrario, se almacenará un registro de direcciones accedidas exitosamente en \public\javascripts\demo.js. si la dirección IP se encuentra en la lista, se asume que está registrada, de esta manera generará un llamado POST "http://localhost:3000/autenticacion" con los respectivos datos de sesión. En ese momento las credenciales pasan por HTTP desde el equipo donde se hizo la solicitud, hacia el servidor donde se encuentra alojado EXPRESS instalado, el mismo servidor Blockchain.

Fig. 34: Instrucciones en el código Express

```
var szIP = $("#loginip").val(),
    szPort = $("#port").val(),
    szUsername = $("#username").val(),
    szPassword = $("#password").val();

if ("" == szIP || "" == szPort || szUsername == "" || szPassword == "") {
    return;
}
var szDeviceIdentify = szIP + "_" + szPort;
var dispositivoRegistrado = $('#ip > option[value="'+szDeviceIdentify+'"]').length > 0;
if (dispositivoRegistrado){
  $.ajax("/autenticacion", {
    type: "POST",
    data: {
      ip: parseInt(szIP.replaceAll(/[^0-9]/g, "")),
      username: szUsername,
      password: szPassword
    }
  }).done((data, textStatus, jqXHR)=>{
    terminarAutenticacion();
  }).fail((jqXHR, textStatus, errorThrown)=>{
    showOPInfo(szDeviceIdentify + " Autenticacion blockchain fallida", textStatus, "");
  });
} else {
  terminarAutenticacion();
}
function terminarAutenticacion(){//depende fuertemente de las variables asignadas anteriormente
  WebVideoCtrl.I_Login(szIP, 1, szPort, szUsername, szPassword, {
      timeout: 3000,
      success: function (xmlDoc) {
          showOPInfo(szDeviceIdentify + " Autenticacion exitosa");
          if (!dispositivoRegistrado)
            $("#ip").prepend("<option value='" + szDeviceIdentify + "'>" + szDeviceIdentify + "</option>");
          setTimeout(function () {
              $("#ip").val(szDeviceIdentify);
              setTimeout(function() {
                  getChannelInfo();
              }, 1000);
              getDevicePort();
```

Nota. Instrucciones de línea en Express al recibir credenciales. Fuente: Diseño propio.

La función de EXPRESS, al recibir las credenciales es bajo la sentencia de código registrada en \routes\index.js. Si todos los datos que le fueron enviados contienen información, le genera una transacción al contrato inteligente, (la tarea es iniciada por EXPRESS, pero la transacción la ejecuta ETHERS con una billetera de prueba), informando mediante una función específica y lleva las credenciales digitadas como parámetros de esa función.

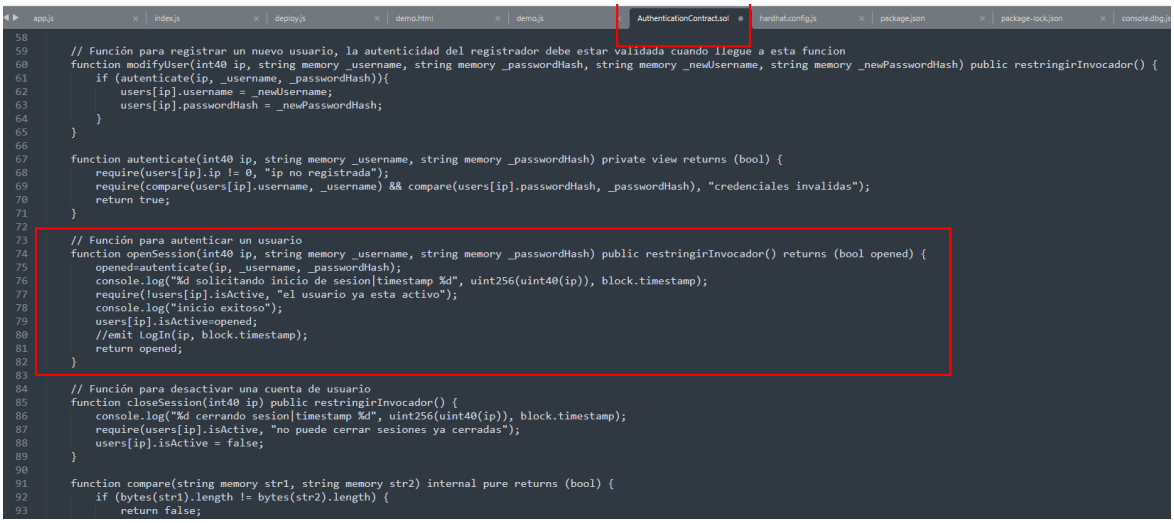
Fig. 35: Generación de transacción a contrato

```
var router = express.Router();
const desplegarContrato = require('../scripts/deploy');
var path = require('path');

var authenticationContract;

/* GET home page. */
router.get('/', async function(req, res, next) {
    authenticationContract = await desplegarContrato();
  res.sendFile(path.join(__dirname, '../views/demo.html'));
});

router.post('/registro', async function(req, res, next) {
    if (req.body.ip && req.body.username && req.body.password){
        try {
            await authenticationContract.registerUser(req.body.ip, req.body.username, req.body.password);
            res.status(200).end();
        } catch (e) {
            res.status(500).send(e);
        }
    } else {
        res.status(400).send('debe especificar ip, nombre de usuario y clave secreta');
    }
});
router.post('/autenticacion', async function(req, res, next) {
    if (req.body.ip && req.body.username && req.body.password){
        try {
            await authenticationContract.openSession(req.body.ip, req.body.username, req.body.password);
            res.status(200).end();
        } catch (e) {
            res.status(500).send(e);
        }
    } else {
        res.status(400).send('debe especificar ip, nombre de usuario y clave secreta');
    }
});
```

Nota. Generación de transacción siempre y cuando contenga toda la información del login. Fuente: Diseño propio.

La función a la que se hace mención está en el código del contrato inteligente, \contracts\AuthenticationContract.sol, en las líneas de openSession.

Fig. 36: Líneas de código de función OpenSession

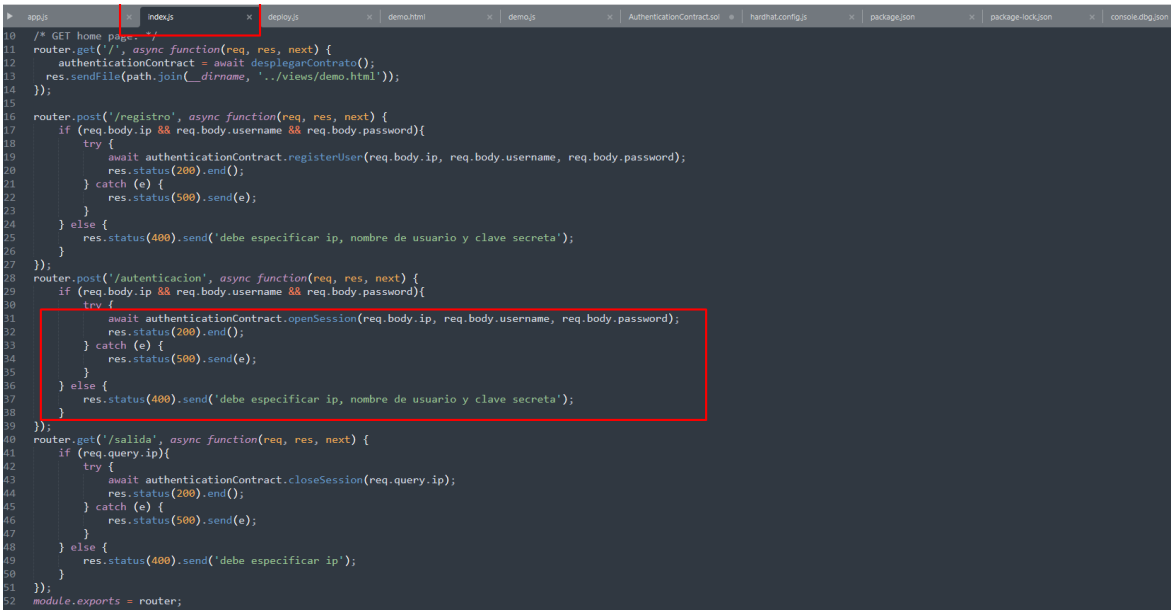


Nota. Función para autenticar un usuario en solidity. Fuente: Diseño propio.

Al crearse la transacción, se registra en la cadena de bloques, lo que se conoce en el mundo de Ethereum como validada (tarea realizada por los validadores, parecido a los mineros en bitcoin). En este caso Hardhat se asegura de que haya cadena de bloques y que el servidor usado para la prueba sirva de validador. Una vez la transacción queda validada, el contrato toma acción de acuerdo con las indicaciones de la transacción, establecida en la función openSession: revisar dentro de la lista de credenciales guardadas si existe una para la IP indicada, así es así, comparar los datos de sesión introducidos con las credenciales guardadas y responder si son idénticas o no, si no son válidas, responde con el mensaje (credenciales invalidas). Desde ese instante inicia el proceso de regreso para responder la solicitud inicial de acceder al dispositivo IoT, siendo EXPRESS el que recibe la respuesta y la interpreta de acuerdo con el código \routes\index.js.

Responde la solicitud POST que le hicieron con un aviso de respuesta de la transacción, si fue exitosa u ocurrió un error (ya sea por no tener acceso a la cadena de bloques o los datos de sesión introducidos son incorrectos).

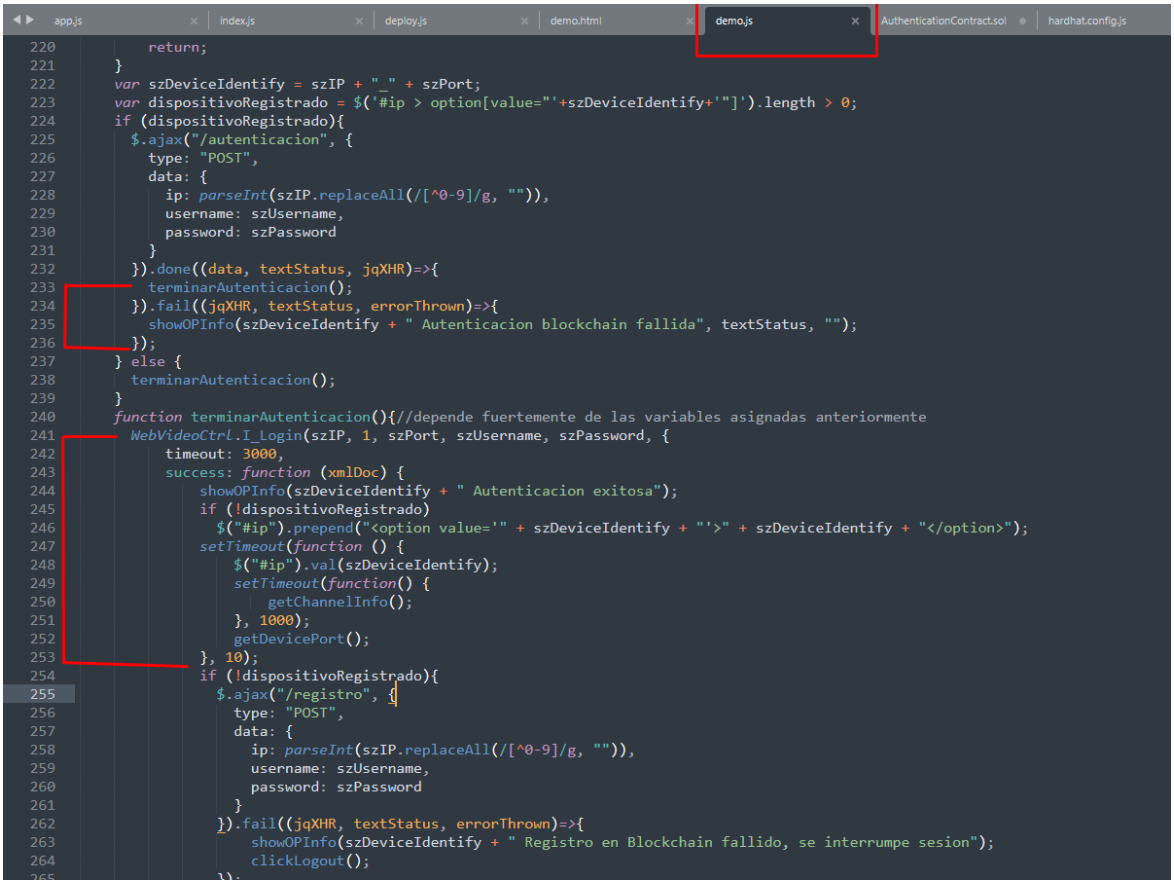
Fig. 37: Respuestas a solicitudes Post



Nota. Respuestas a solicitudes recibidas Post. Fuente: Diseño propio.

La respuesta llega directamente al código alojado en \public\javascripts\demo.js, el cual actúa según la respuesta anterior (\routes\index.js) entre las líneas 233 y 236. De ser exitosa la respuesta, se identifica entre las líneas 241 y 253.

Fig. 38: Terminar o iniciar autenticación




```
        return;
    }
    var szDeviceIdentify = szIP + "_" + szPort;
    var dispositivoRegistrado = $('#ip > option[value="'+szDeviceIdentify+'"]').length > 0;
    if (dispositivoRegistrado){
      $.ajax("/autenticacion", {
        type: "POST",
        data: {
          ip: parseInt(szIP.replaceAll(/[^0-9]/g, "")),
          username: szUsername,
          password: szPassword
        }
      }).done((data, textStatus, jqXHR)=>{
        terminarAutenticacion();
      }).fail((jqXHR, textStatus, errorThrown)=>{
        showOPInfo(szDeviceIdentify + " Autenticacion blockchain fallida", textStatus, "");
      });
    } else {
      terminarAutenticacion();
    }
    function terminarAutenticacion(){//depende fuertemente de las variables asignadas anteriormente
      WebVideoCtrl.I_Login(szIP, 1, szPort, szUsername, szPassword, {
          timeout: 3000,
          success: function (xmlDoc) {
              showOPInfo(szDeviceIdentify + " Autenticacion exitosa");
              if (!dispositivoRegistrado)
                $("#ip").prepend("<option value='" + szDeviceIdentify + "'>" + szDeviceIdentify + "</option>");
              setTimeout(function () {
                  $("#ip").val(szDeviceIdentify);
                  setTimeout(function() {
                      getChannelInfo();
                  }, 1000);
                  getDevicePort();
              }, 10);
              if (!dispositivoRegistrado){
                $.ajax("/registro", {
                  type: "POST",
                  data: {
                    ip: parseInt(szIP.replaceAll(/[^0-9]/g, "")),
                    username: szUsername,
                    password: szPassword
                  }
                }).fail((jqXHR, textStatus, errorThrown)=>{
                    showOPInfo(szDeviceIdentify + " Registro en Blockchain fallido, se interrumpe sesion");
                    clickLogout();
```

Nota. Función para iniciar o terminar una autenticación. Fuente: Diseño propio.

WebVideoCtrl.I_Login: es la función proveniente del WebSDK que permite autenticarse de manera automática a la cámara, la cual se usa pasándole las credenciales ingresadas por el usuario (ahora confirmadas como correctas en los pasos anteriores, después de compararlas) y finalmente la cámara concede acceso al usuario que consulta, permitiendo ver la imagen en vivo y hacer uso de todos los botones, mover a la derecha, a la izquierda, subir, bajar, tomar captura, etc.

¿Ahora bien, como es entonces este vínculo de WebSDK y Hardhat? el vínculo entre WebSDK y el Hardhat en este proyecto es crucial para permitir la comparación de las credenciales ingresadas a través de la interfaz web con las credenciales almacenadas en la blockchain. A continuación, se detalla cómo se integran estos dos componentes para que el proceso de autenticación funcione correctamente.

1. Interfaz web y WebSDK:

La interfaz web es la puerta de entrada para el usuario. Allí, el propietario de la cámara ingresa las credenciales necesarias, como la dirección IP, el usuario y la contraseña. Este formulario interactúa con el WebSDK provisto por Hikvision, que permite a la interfaz web enviar solicitudes de autenticación y control a las cámaras de forma automatizada.

Cuando el usuario presiona el botón de "Login" en la interfaz, el WebSDK toma los datos proporcionados y los pasa al servidor backend. En este proceso, el WebSDK maneja la lógica de interacción con las cámaras, mientras que el acceso y comparación de credenciales ocurre a nivel de la blockchain.

2. Backend con HardHat:

Una vez que la interfaz web ha recolectado las credenciales, estas se envían al servidor backend, que está configurado para escuchar peticiones HTTP utilizando Express.js. En este punto, entra en juego HardHat, que simula una blockchain local donde el contrato inteligente de autenticación ha sido desplegado.

HardHat actúa como una red de pruebas de Ethereum que permite desarrollar y testear contratos inteligentes sin los costos asociados a una red pública de blockchain. Es en este entorno donde las credenciales ingresadas por el usuario se comparan con las almacenadas previamente en la blockchain.

3. Proceso de autenticación y comparación de credenciales:

Cuando el servidor backend recibe las credenciales ingresadas a través de la interfaz web, se genera una transacción a la blockchain local de HardHat. A través de la librería Ethers.js, que facilita la interacción entre el backend y los contratos inteligentes, se invoca una función del contrato inteligente llamada openSession.

Esta función compara las credenciales ingresadas (usuario, contraseña e IP) con las credenciales almacenadas en la blockchain. Las credenciales están quemadas directamente en el código del contrato inteligente para esta prueba de concepto, pero el proceso de comparación sigue los pasos necesarios para verificar si la información coincide con lo que está almacenado en la blockchain.

- Si las credenciales coinciden, se genera una respuesta positiva que permite el acceso a la cámara.

- Si las credenciales no coinciden, el sistema rechaza la solicitud con un mensaje de "credenciales inválidas".

4. Respuesta al WebSDK:

Una vez que el contrato inteligente en la blockchain (simulada por HardHat) ha validado las credenciales, el resultado de la comparación es devuelto al backend. Express.js recibe esta respuesta y la interpreta, devolviendo un mensaje a la interfaz web. Si las credenciales fueron correctas, el WebSDK de Hikvision se encarga de completar el proceso de autenticación y permite al usuario acceder a la cámara de manera automática.

Adicional a las funciones ya usadas, se cuentan con otras que permiten manejo adicional entre el dispositivo IoT y la Blockchain.

Fig. 39: Obtención datos de canal IP

```
// get IP channel
async function clickGetDigitalChannelInfo() {
    var szDeviceIdentify = $("#ip").val(),
        iAnalogChannelNum = 0;

    $("#digitalchannellist").empty();

    if (null == szDeviceIdentify) {
        return;
    }

    try {
        var oAnalogChannelInfo = await WebVideoCtrl.I_GetAnalogChannelInfo(szDeviceIdentify, {});
        iAnalogChannelNum = $(oAnalogChannelInfo).find("VideoInputChannel").length;
    } finally {
        WebVideoCtrl.I_GetDigitalChannelInfo(szDeviceIdentify, {
            success: function (xmlDoc) {
                var oChannels = $(xmlDoc).find("InputProxyChannelStatus");

                $.each(oChannels, function () {
                    var id = parseInt($(this).find("id").eq(0).text(), 10),
                        ipAddress = $(this).find("ipAddress").eq(0).text(),
                        srcInputPort = $(this).find("srcInputPort").eq(0).text(),
                        managePortNo = $(this).find("managePortNo").eq(0).text(),
                        online = $(this).find("online").eq(0).text(),
                        proxyProtocol = $(this).find("proxyProtocol").eq(0).text();

                    var objTr = $("#digitalchannellist").get(0).insertRow(-1);
                    var objTd = objTr.insertCell(0);
                    objTd.innerHTML = (id - iAnalogChannelNum) < 10 ? "D0" + (id - iAnalogChannelNum) : "D" + (id - iAnalogChannelNum);
                    objTd = objTr.insertCell(1);
                    objTd.width = "25%";
                    objTd.innerHTML = ipAddress;
                    objTd = objTr.insertCell(2);
                    objTd.width = "15%";
                    objTd.innerHTML = srcInputPort;
                    objTd = objTr.insertCell(3);
                    objTd.width = "20%";
                    objTd.innerHTML = managePortNo;
                    objTd = objTr.insertCell(4);
                    objTd.width = "15%";
                    objTd.innerHTML = "true" == online ? "online" : "offline";
                    objTd = objTr.insertCell(5);
                    objTd.width = "25%";
                    objTd.innerHTML = proxyProtocol;
                });
                showOPInfo(szDeviceIdentify + " get IP channel success.");
            },
            error: function (oError) {
                showOPInfo(szDeviceIdentify + " no IP channel ", oError.errorCode, oError.errorMsg);
            }
        });
    }
}
```

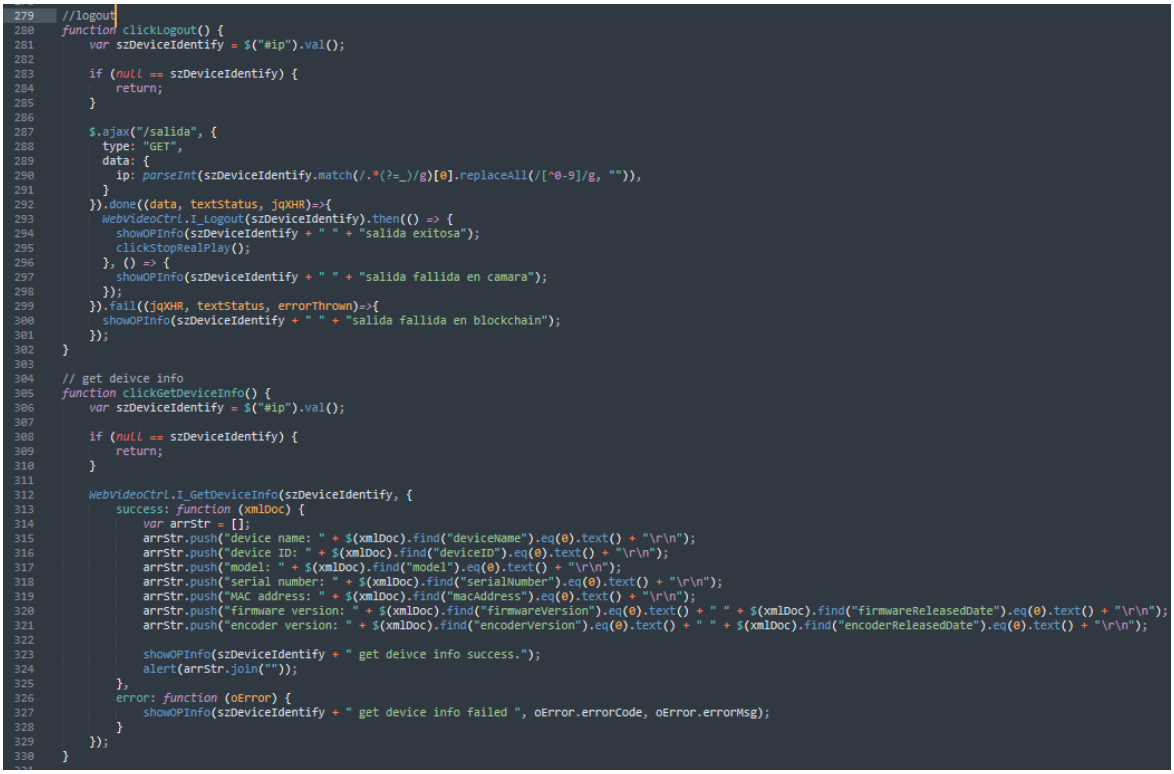
Nota. Despliegue de código del SDK recolección datos del canal IP. Fuente: Diseño propio

Fig. 40: Código de Login

```
// login
function clickLogin() {
    var szIP = $("#loginip").val(),
        szPort = $("#port").val(),
        szUsername = $("#username").val(),
        szPassword = $("#password").val();

    if ("" == szIP || "" == szPort || szUsername == "" || szPassword == "") {
        return;
    }
    var szDeviceIdentify = szIP + "_" + szPort;
    var dispositivoRegistrado = $('#ip > option[value="'+szDeviceIdentify+'"]').length > 0;
    if (dispositivoRegistrado){
      $.ajax("/autenticacion", {
        type: "POST",
        data: {
          ip: parseInt(szIP.replaceAll(/[^0-9]/g, "")),
          username: szUsername,
          password: szPassword
        }
      }).done((data, textStatus, jqXHR)=>{
        terminarAutenticacion();
      }).fail((jqXHR, textStatus, errorThrown)=>{
        showOPInfo(szDeviceIdentify + " Autenticacion blockchain fallida", textStatus, "");
      });
    } else {
      terminarAutenticacion();
    }
    function terminarAutenticacion(){//depende fuertemente de las variables asignadas anteriormente
      WebVideoCtrl.I_Login(szIP, 1, szPort, szUsername, szPassword, {
          timeout: 3000,
          success: function (xmlDoc) {
              showOPInfo(szDeviceIdentify + " Autenticacion exitosa");
              if (!dispositivoRegistrado)
                $("#ip").prepend("<option value='" + szDeviceIdentify + "'>" + szDeviceIdentify + "</option>");
              setTimeout(function () {
                  $("#ip").val(szDeviceIdentify);
                  setTimeout(function() {
                      getChannelInfo();
                  }, 1000);
                  getDevicePort();
              }, 10);
              if (!dispositivoRegistrado){
                $.ajax("/registro", {
                  type: "POST",
                  data: {
                    ip: parseInt(szIP.replaceAll(/[^0-9]/g, "")),
                    username: szUsername,
                    password: szPassword
                  }
                }).fail((jqXHR, textStatus, errorThrown)=>{
                    showOPInfo(szDeviceIdentify + " Registro en Blockchain fallido, se interrumpe sesion");
                    clickLogout();
                });
              }
```

Nota. Líneas de código para el Login. Fuente: Diseño propio

Fig. 41: Líneas de código de logout



```
//logout
function clickLogout() {
    var szDeviceIdentify = $("#ip").val();

    if (null == szDeviceIdentify) {
        return;
    }

    $.ajax("/salida", {
      type: "GET",
      data: {
        ip: parseInt(szDeviceIdentify.match(/.*(?=_)/g)[0].replaceAll(/[^0-9]/g, "")),
      }
    }).done((data, textStatus, jqXHR)=>{
      WebVideoCtrl.I_Logout(szDeviceIdentify).then(() => {
        showOPInfo(szDeviceIdentify + " " + "salida exitosa");
        clickStopRealPlay();
      }, () => {
        showOPInfo(szDeviceIdentify + " " + "salida fallida en camara");
      });
    }).fail((jqXHR, textStatus, errorThrown)=>{
      showOPInfo(szDeviceIdentify + " " + "salida fallida en blockchain");
    });
}

// get deivce info
function clickGetDeviceInfo() {
    var szDeviceIdentify = $("#ip").val();

    if (null == szDeviceIdentify) {
        return;
    }

    WebVideoCtrl.I_GetDeviceInfo(szDeviceIdentify, {
        success: function (xmlDoc) {
            var arrStr = [];
            arrStr.push("device name: " + $(xmlDoc).find("deviceName").eq(0).text() + "\r\n");
            arrStr.push("device ID: " + $(xmlDoc).find("deviceID").eq(0).text() + "\r\n");
            arrStr.push("model: " + $(xmlDoc).find("model").eq(0).text() + "\r\n");
            arrStr.push("serial number: " + $(xmlDoc).find("serialNumber").eq(0).text() + "\r\n");
            arrStr.push("MAC address: " + $(xmlDoc).find("macAddress").eq(0).text() + "\r\n");
            arrStr.push("firmware version: " + $(xmlDoc).find("firmwareVersion").eq(0).text() + " " + $(xmlDoc).find("firmwareReleasedDate").eq(0).text() + "\r\n");
            arrStr.push("encoder version: " + $(xmlDoc).find("encoderVersion").eq(0).text() + " " + $(xmlDoc).find("encoderReleasedDate").eq(0).text() + "\r\n");

            showOPInfo(szDeviceIdentify + " get deivce info success.");
            alert(arrStr.join(""));
        },
        error: function (oError) {
            showOPInfo(szDeviceIdentify + " get device info failed ", oError.errorCode, oError.errorMsg);
        }
    });
}
```

Nota. Líneas de código logout, para desconexión después de haber iniciado sesión. Fuente: Diseño propio

Fig. 42: Líneas de código para el inicio de grabación.

```
// start record
var g_szRecordType = "";
function clickStartRecord(szType) {
    var oWndInfo = WebVideoCtrl.I_GetWindowStatus(g_iWndIndex),
        szInfo = "";

    g_szRecordType = szType;

    if (oWndInfo != null) {
        var szChannelID = $("#channels").val(),
            szFileName = oWndInfo.szDeviceIdentify + "_" + szChannelID + "_" + new Date().getTime();

        WebVideoCtrl.I_StartRecord(szFileName, {
            bDateDir: true,
            success: function () {
                if ('realplay' === szType) {
                    szInfo = "start recording success.";
                } else if ('playback' === szType) {
                    szInfo = "start clip success.";
                }
                showOPInfo(oWndInfo.szDeviceIdentify + " " + szInfo);
            },
            error: function (oError) {
                if ('realplay' === szType) {
                    szInfo = " start recording failed.";
                } else if ('playback' === szType) {
                    szInfo = " start clip failed.";
                }
                showOPInfo(szDeviceIdentify + szInfo, oError.errorCode, oError.errorMsg);
            }
        });
    }
}
```

Nota. Código relacionado para el inicio de grabación del dispositivo IoT. Fuente: Diseño propio

Fig. 43: Línea de código de parada de grabación

```
// stop record
function clickStopRecord(szType, iWndIndex) {
    if ("undefined" === typeof iWndIndex) {
        iWndIndex = g_iWndIndex;
    }
    var oWndInfo = WebVideoCtrl.I_GetWindowStatus(iWndIndex),
        szInfo = "";

    if (oWndInfo != null) {
        WebVideoCtrl.I_StopRecord({
            success: function () {
                if ('realplay' === szType) {
                    szInfo = "stop recording success.";
                } else if ('playback' === szType) {
                    szInfo = "stop clip success.";
                }
                showOPInfo(oWndInfo.szDeviceIdentify + " " + szInfo);
            },
            error: function (oError) {
                if ('realplay' === szType) {
                    szInfo = "stop recording failed.";
                } else if ('playback' === szType) {
                    szInfo = "stop clip failed.";
                }
                sshowOPInfo(szDeviceIdentify + szInfo, oError.errorCode, oError.errorMsg);
            }
        });
    }
}
```

Nota. Líneas de código donde relaciona el proceso para detener la grabación del dispositivo. Fuente: Diseño propio.

Fig. 44: Código base contrato solidity

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

import "hardhat/console.sol";

/*
contrato diseñado para gestionar credenciales de IPs, si la identidad del emisor es sospechoza, se "oscurecen" las respuestas
*/
contract AuthenticationContract {
    address payable private immutable OWNER;
    constructor(){
        OWNER=payable(msg.sender);
    }
    // Estructura para representar a un usuario
    struct User {
        int40 ip; //direccion de red a la que pertenecen estas credenciales
        string username;     // Nombre de usuario
        /*byte32*/string passwordHash; // Hash de la contraseña (debe ser almacenada de manera segura fuera del contrato)
        bool isActive;       // Estado de la sesion (activa o inactiva)
    }

    // Mapeo de direcciones Ethereum a usuarios
    mapping(int40 => User) private users;


    // Evento para registrar eventos de inicio de sesión exitosos
    //event LogIn(int40 ip, uint timestamp);

    modifier restringirInvocador(){
        require(msg.sender == OWNER);
        _;
    }

    // Función para registrar un nuevo usuario, la autenticidad del registrador debe estar validada cuando llegue a esta funcion
    function registerUser(int40 ip, string memory _username, string memory _passwordHash) public restringirInvocador() {
        bool isCredentialUnregistered = users[ip].ip == 0;
        if(isCredentialUnregistered){
            console.log("Registrando credencial para %d|timestamp %d", uint256(uint40(ip)), block.timestamp);
        } else {
            console.log("Se intento sobreescribir una credencial|timestamp %d", block.timestamp);
        }
        require(isCredentialUnregistered, "Ya hay una credencial registrada");
        users[ip] = User({
            ip: ip,
            username: _username,
            passwordHash: _passwordHash,
            isActive: true
        });
    }

    // Función para registrar un nuevo usuario, la autenticidad del registrador debe estar validada cuando llegue a esta funcion
    function modifyUser(int40 ip, string memory _username, string memory _passwordHash, string memory _newUsername, string memory _newPasswordHash) public restringirInvocador() {
        if (autenticate(ip, _username, _passwordHash)){
            users[ip].username = _newUsername;
            users[ip].passwordHash = _newPasswordHash;
        }
    }
```

Nota. Líneas de código diseñado para gestionar credenciales de IP, función para autenticar usuario, desactivación de cuenta. Fuente: Diseño propio.

Solidity: Lenguaje de programación para desarrollar contratos inteligentes que enlazan la autenticación sobre Ethereum.

Ether.js: es una biblioteca completa y compacta para interactuar con Ethereum Blockchain y su ecosistema.

Hardhad: Ambiente de prueba de blockchain sin recurrir en consumo de gas(dinero) en la cadena de bloques en línea.

Sublime Text: Editor de código fuente para el proyecto.

Con las herramientas que se reunieron, se construyó el modelo en el cual se configuró la cámara con una dirección IP fija (192.168.1.150) el puerto HTTP se dejó por defecto el cual corresponde al puerto 80, seguidamente se asigna una contraseña(abcd1234) al usuario por defecto admin el cual no se puede deshabilitar o renombrar; continuamente se configura el proyecto en Node.js sobre JavaScript, el cual interactúa con el SDK también sobre

JavaScript los cuales permiten y responden a las solicitudes HTTP, requiriendo la instalación de un plugin hcweb para el correcto funcionamiento.

Entre las variables, el contrato hace una comparación de lo registrado y lo recibido por cada nueva solicitud.

Fig. 45: Comparación de registro

```
function compare(string memory str1, string memory str2) internal pure returns (bool) {
    if (bytes(str1).length != bytes(str2).length) {
        return false;
    }
    return keccak256(abi.encodePacked(str1)) == keccak256(abi.encodePacked(str2));
}
```

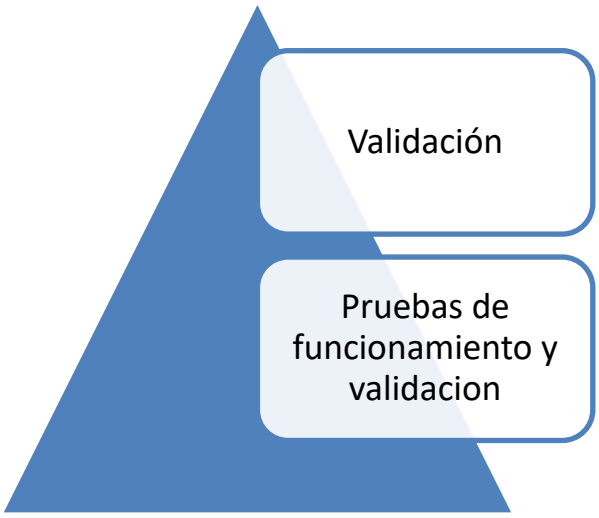
Nota. Comparación de variable de registro y datos entrantes en la solicitud. Fuente: Captura propia

Nota: Para mayor descripción dirigirse al anexo (Instalación de aplicaciones).

2.4 Fase 4 Validación.

Con el modelo ya construido y las herramientas definidas, esta fase corresponde al proceso final de las pruebas sobre el modelo de autenticación a través de blockchain mediante Ethereum.

Fig. 46: Esquema de actividades referente a la fase 4



Nota. Actividades relacionadas en ejecución a la fase 2. Fuente: diseño propio

2.4.1 Pruebas de funcionamiento y validación

Con el código creado y alojado entre el dispositivo y el usuario se validan con la cámara comprada y el nodo local, validando la autenticación con blockchain, verificando en los mismos las características positivas o negativas.

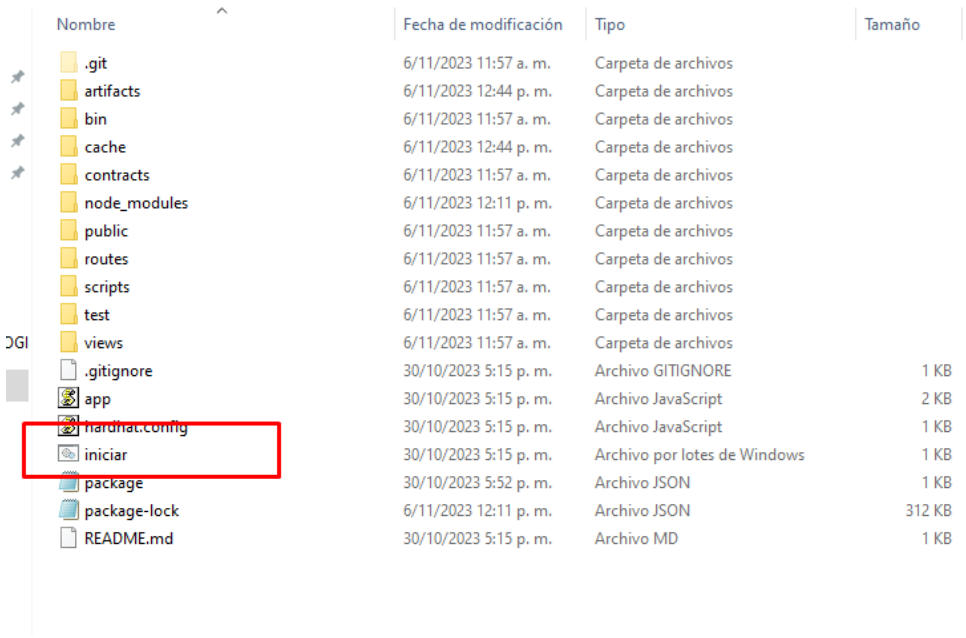
Con las pruebas realizadas al equipo disponible, se tomaron muestras digitales de los escenarios resultantes, generando un informe digital con los resultados y conclusiones obtenidas, definiendo la viabilidad de este.

Se realizó prueba de concepto en el cual se logró interactuar entre el usuario quien realiza las peticiones por HTTP y la cámara IoT que responde a las peticiones, todo bajo la cadena de bloques Ethereum la cual registra satisfactoriamente los contratos sobre cada bloque, asignando un numero de transacción y bloque a cada llamada o petición realizada a la cámara sobre la Blockchain**.**

El primer paso y teniendo en cuenta que ya todas las herramientas se encuentran instaladas y el código compilado, es ejecutar el proyecto, dando doble clic sobre el script

iniciar.bat, el cual inicia el contrato de la blockchain y el SDK, como se identifica en la imagen siguiente.

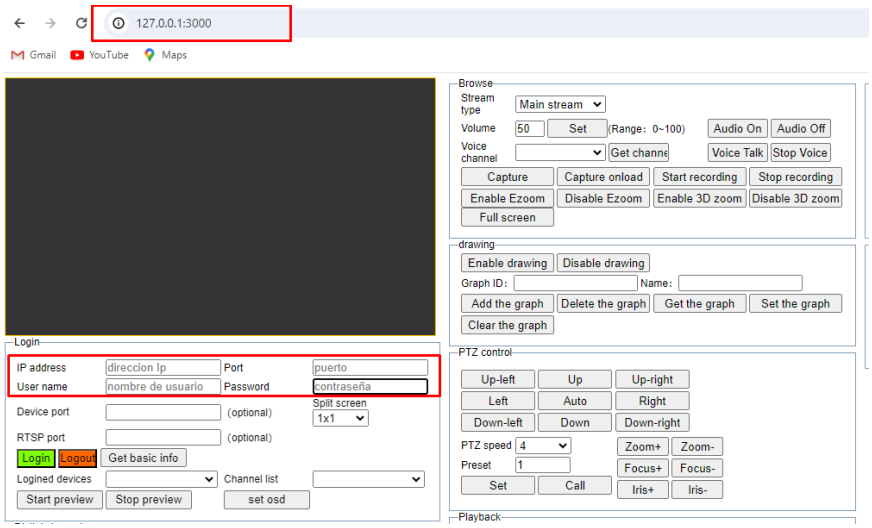
Fig. 47: Ejecución de contrato



Nota. Ejecución de proyecto dando doble clic sobre script. Fuente: Captura propia

Al ejecutar este proyecto, se inician dos terminales, una registra el log del SDK y la segunda terminal registra el despliegue del contrato junto con las transacciones y bloques asignados a cada transacción. Estando ya el proyecto ejecutado, se procede con la petición hacia el servidor blockchain con su respectivo puerto, local host (127.0.0.1:3000) o si es desde otro equipo en la misma red seria bajo la IP del servidor y el puerto (192.168.1.97:3000). Se carga un entorno grafico con los campos de autenticación en blanco, dirección IP, puerto, usuario y contraseña, como se identifica en la imagen siguiente.

Fig. 48: Petición servidor Blockchain

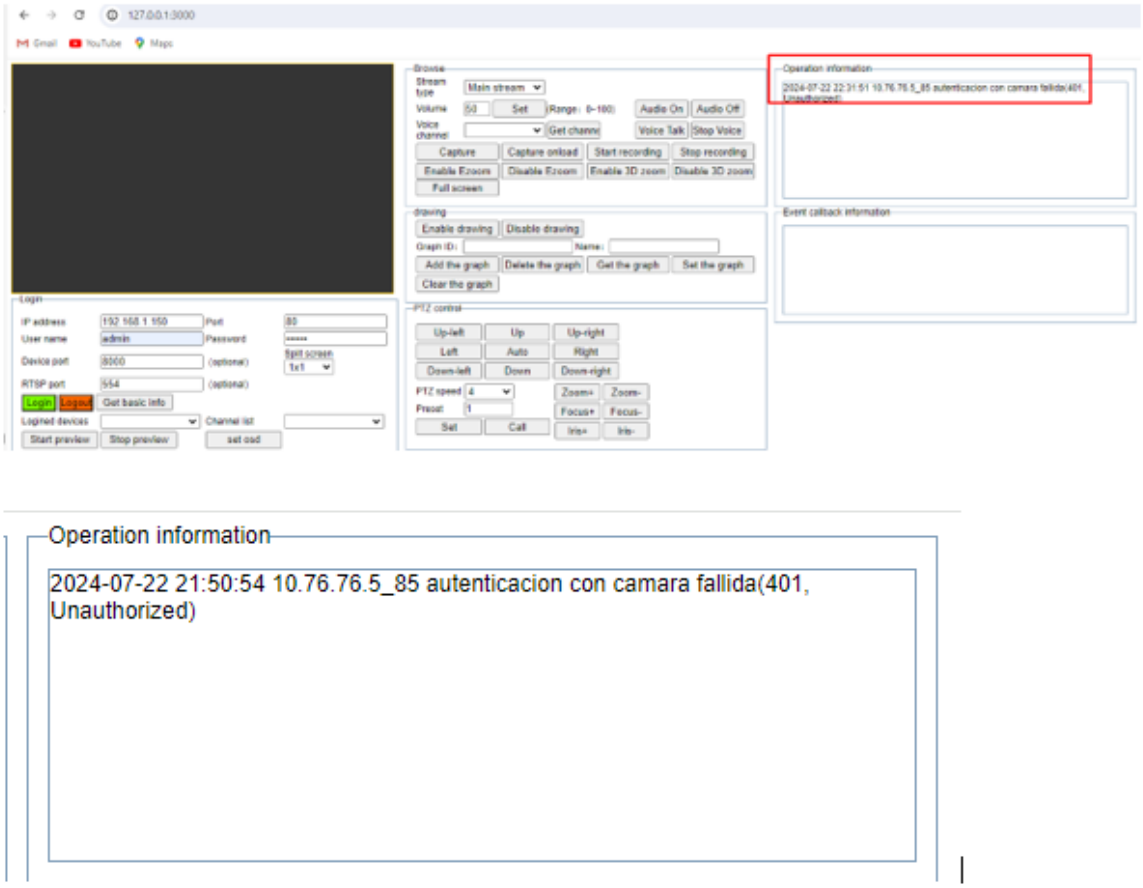


Nota. Petición HTTP hacia servidor Blockchain y cargue de entorno grafico de sesión. Fuente: Captura propia.

Teniendo en cuenta que al trabajar con Blockchain Ethereum, se puede personalizar los resultados, se ajustaron las respuestas a las peticiones que se realizan mediante el SDK hacia la Blockchain. Inicialmente se ingresan contraseñas erradas con el fin de ver el comportamiento.

En el momento que el usuario debe ingresar los datos de sesión, tales como usuario y contraseña, y estos son inválidos, el sistema deniega el acceso, hacia el dispositivo retornando un error 401 – autenticación con cámara fallida.

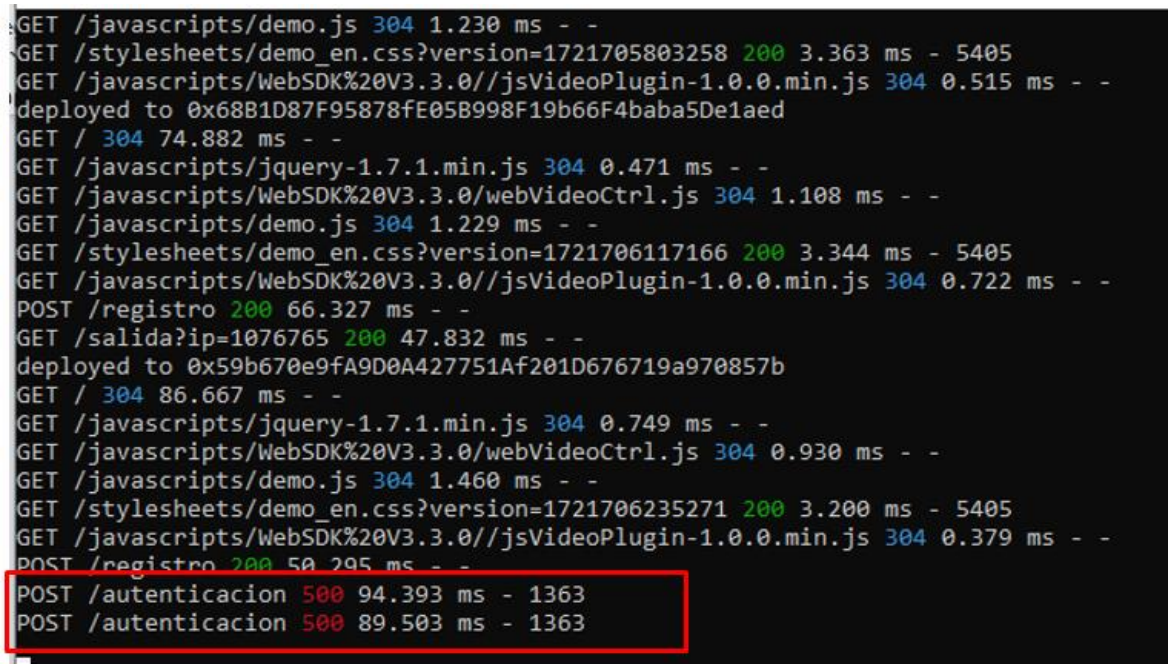
Fig. 49: Inicio de sesión incorrecta



Nota. Registro de sesión con contraseñas erradas. Fuente: Captura propia

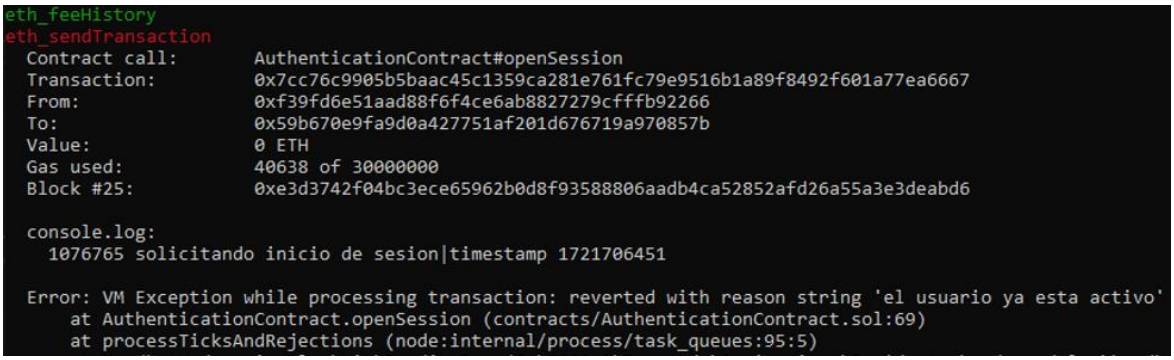
En la imagen a continuación, se identifica un código de error 500, este código es generado cuando el usuario se encuentra logueado en una sesión activa ante el dispositivo IoT mediante la blockchain, y presiona de nuevo el botón login para iniciar de nuevo sesión, pero como aun cuenta con una sesión activa el sistema no le permite volver a ingresar. En este proceso el SDK envía la petición a volver a desplegar un contrato en la blockchain, pero como ya hay un contrato activo, la blockchain rechaza un nuevo contrato con una sesión activa arrojando error 500 usuario ya activo.

Fig. 50: Código de error Sesión activa



Nota. Terminal de registro de los logs de sesión ya activa. Fuente: Captura propia

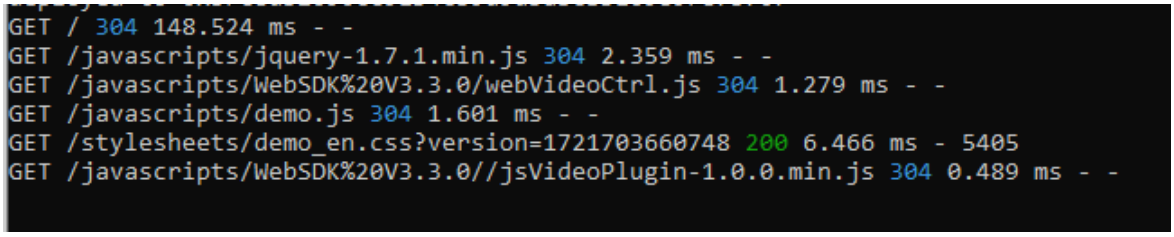
Fig. 51: Complemento sesión activa



Nota. Terminal de registro de los logs del servidor con usuario ya activo, que intenta iniciar de nuevo. Fuente: Captura propia.

Adicionalmente cuando el usuario cierra la sesión, el sistema no le permite volver a iniciar sesión, hasta que no recargue de nuevo la página, lo que genera un código 304 de nueva solicitud.

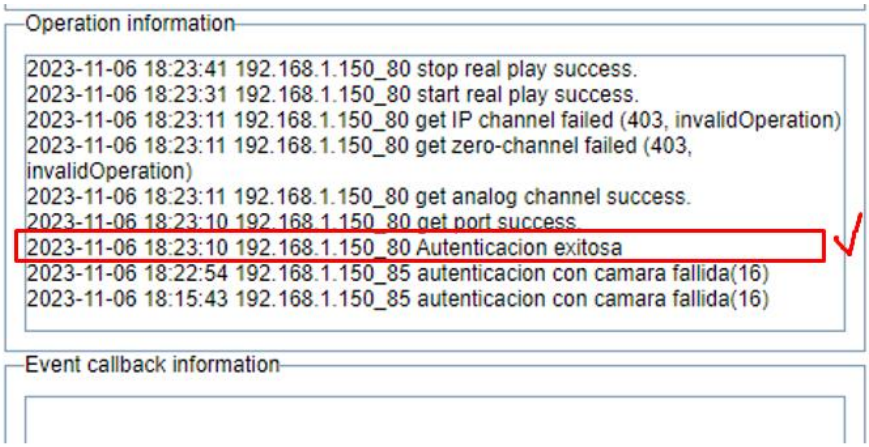
Fig. 52: Código de nueva solicitud



Nota. Terminal de registro de los logs del servidor, nueva solicitud de usuario. Fuente: Captura propia

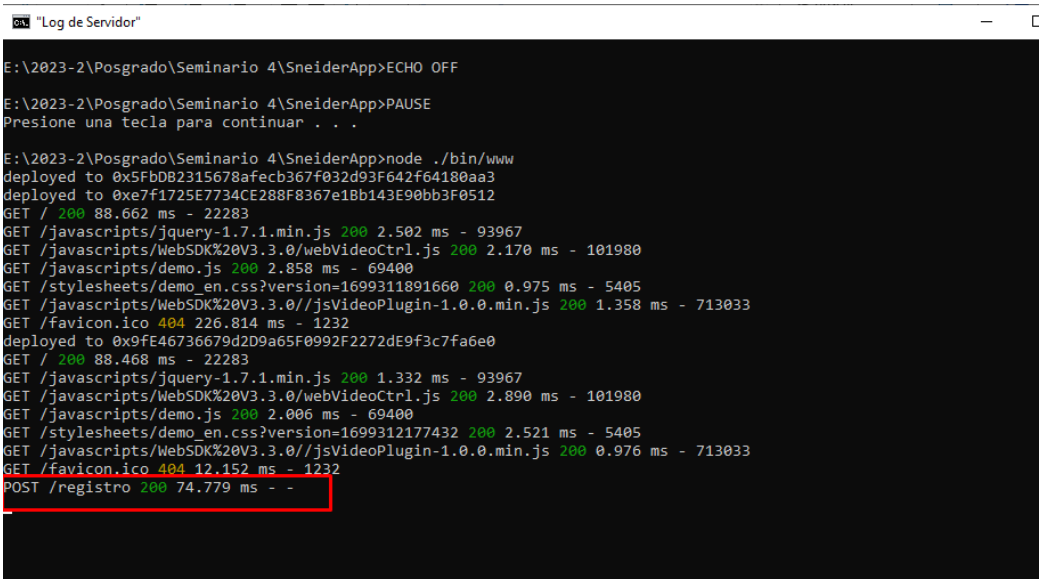
Nuevamente se realiza la prueba, de solicitud HTTP al servidor Blockchain, pero con autenticación correcta.

Fig. 53: Inicio de sesión correcta



Nota. Inicio de sesión correcta mediante solicitud HTTP al servidor. Fuente: Captura propia

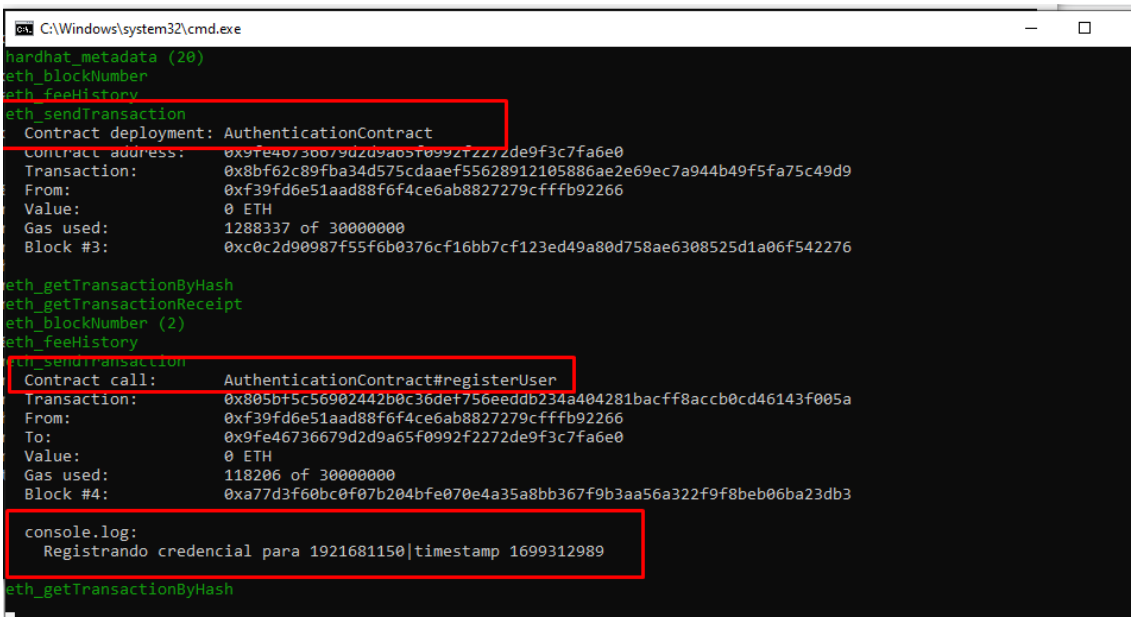
Fig. 54: Inicio de sesión correcta



Nota. Terminal de registro de los logs, contraseñas correctas. Fuente: Captura propia

En la fig. 55 (Terminal log de contrato inteligente) a continuación, se encuentra la terminal que genera los registros relacionados con el despliegue del contrato inteligente, las llamadas al contrato, las transacciones sobre la blockchain de acuerdo con el inicio de sesión correcto bajo el registro de la imagen anterior.

Fig. 55: Terminal log de contrato inteligente (SDK)



Nota. Terminal de registro l contrato inteligente y transacciones sobre la blockchain (SDK). Fuente: Captura propia.

Contract deployment: Despliegue de contrato, nombre del contrato desplegado

Contract address: Dirección de contrato de la blockchain

Transaction: Numero ascendente de transacción

From: Dirección desde donde se hacen las transacciones

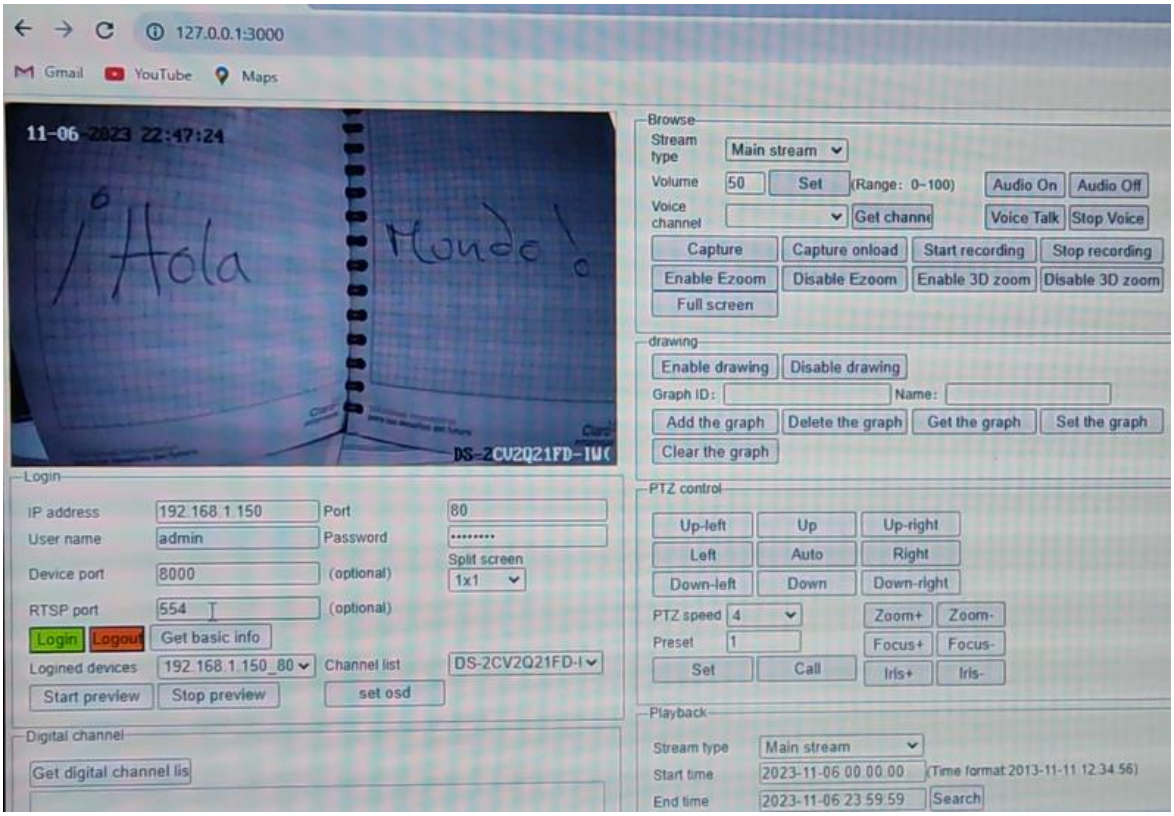
Value: Valor de la transacción en Ethers

Gas used: Valor descontado de la transacción

Block #: identificación de bloque creado relacionado a la transacción

Al tener ya el contrato corriendo se procedió con el acceso o solicitud HTTP hacia la cámara, pasando por la Blockchain, la cual por ser un entorno local se encuentra corriendo en localhost bajo el puerto 3000, localhost:3000 siendo lo mismo ingresando a través de 127.0.0.1:3000, al iniciar sesión, en el sistema, la terminal log servidor, registra una línea de código 200, la cual se refiere al registro del usuario en la Blockchain.

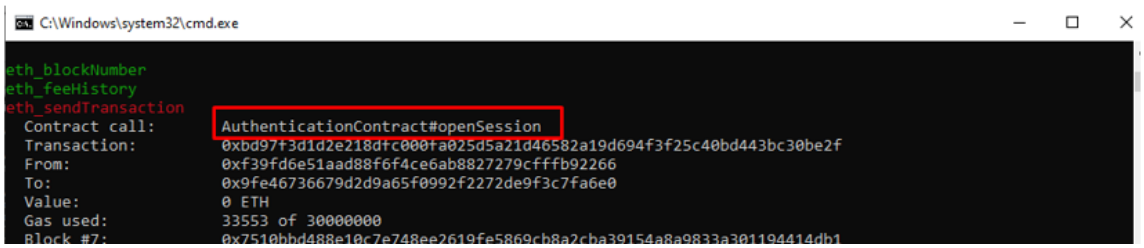
Fig. 56: Inicio de sesión en el dispositivo IoT



Nota. Inicio de sesión en el dispositivo IoT mediante de la cadena de bloques Ethereum. Fuente: Captura propia.

La imagen anterior ilustra el video en vivo del dispositivo, la página cuenta con unos botones gracias al SDK del proveedor, uno en color verde (login) el cual es necesario para iniciar sesión, luego se digita usuario, dirección IP, puerto y contraseña en la parte superior de los botones mencionados. Al lado derecho se encuentran los botones (Up-left, Up, Up-right, Down-left, Down, entre otros) que permitieron girar la cámara hacia arriba, hacia abajo, a la derecha, a la izquierda, entre otros.

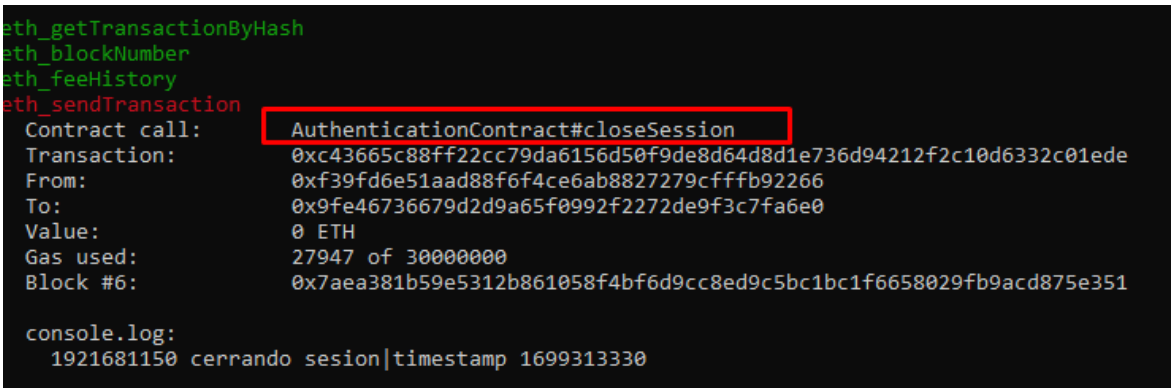
Fig. 57: Llamada de contrato inicio de sesión



Nota. Registro de inicio de sesión en el contrato Blockchain. Fuente: Captura propia.

Para cerrar el video y salir del dispositivo es solo presionar el botón logout, el cual finaliza la sesión, y para terminar el contrato, simplemente se cerraron las terminales descritas anteriormente, el cual finaliza el contrato.

Fig. 58: Llamada de contrato cierre de sesión



Nota. Registro de cierre de sesión en el contrato Blockchain. Fuente: Captura propia.

Entendiendo que, para esta prueba de concepto, es suficiente que la comparación de las credenciales ingresadas por el usuario coincida con las almacenadas directamente en la blockchain. Este enfoque permite verificar el correcto funcionamiento del sistema de autenticación y acceso.

Sin embargo, es fundamental tener en cuenta que, por mejores prácticas de seguridad, y dado que la blockchain tiene una naturaleza pública donde la información es accesible para cualquier nodo, el manejo de credenciales en texto claro no es adecuado para una implementación en producción. Aunque Ethereum proporciona una infraestructura descentralizada y confiable, los datos almacenados en la blockchain pueden ser visibles para

cualquier participante de la red, lo que pone en riesgo la confidencialidad de la información sensible como las contraseñas.

Para garantizar la protección de las credenciales en un entorno real de producción, se deben tomar medidas adicionales. A continuación, algunos pasos que se podrían seguir para mejorar la seguridad:

1. Hasheo de contraseñas: En lugar de almacenar las contraseñas en texto claro, se deben almacenar utilizando algoritmos de hash como Keccak256 o SHA-256. De esta forma, aunque el hash esté expuesto públicamente en la blockchain, no se podrá revertir para obtener la contraseña original, asegurando así su confidencialidad.

2. Encriptación de datos sensibles: Además de hashear las contraseñas, otros datos sensibles podrían ser encriptados antes de almacenarse en la blockchain. Solo los nodos autorizados, que posean la clave de desencriptación, podrán acceder a esa información.

3. Autenticación de nodos: Se puede establecer un mecanismo de autenticación que permita que solo ciertos nodos autorizados interactúen con los contratos inteligentes o accedan a la información sensible. Esto podría lograrse mediante contratos inteligentes adicionales o una capa de autorización fuera de la blockchain.

La implementación de este modelo de autenticación con blockchain Ethereum genera una estrategia efectiva para mitigar las brechas de seguridad en cámaras IoT para hogares, toda vez que ofrece una capa adicional y contribuyendo a un nivel de seguridad, con base a las vulnerabilidades y riegos como: autenticación incorrecta, contraseña en el archivo de configuración, acceso no autorizado, ataque de fuerza bruta, ataque de reenvío de contraseña, teniendo en cuanta que no todo sistema es cien por ciento seguro, lo que se trabajó fue la mitigación y que bajo los parámetros trabajados se genera un ambiente más seguro, que a través del tiempo requiere mayor robustes, actualización y mejoras, debido a que los avances tecnológicos y entornos IoT cambian constantemente. Con el objetivo de esta mitigación se trabajaron las capas de seguridad sobre:

Eliminación de contraseñas en archivos de configuración: En el sistema de autenticación basado en blockchain Ethereum, los usuarios no dependen de contraseñas tradicionales almacenadas en archivos de configuración. En su lugar, se utiliza claves cifradas, únicas asociadas con cada usuario, distribuidas sobre los bloques que componen la cadena, lo que elimina la necesidad de almacenar contraseñas en texto plano, en un único directorio y reduce el riesgo de compromiso de contraseñas.

Identificación segura de usuarios: En esta implementación, los usuarios pueden identificarse de manera segura mediante el uso de claves criptográficas únicas. Estas claves son identificadas por cada usuario dentro de los contratos, son difíciles de comprometer y proporcionan una forma de autenticación que reduce el riesgo de acceso no autorizado. Esto garantiza que solo los usuarios autorizados puedan acceder a las cámaras IoT y que cualquier intento de acceso no autorizado pueda ser detectado y bloqueado de manera temporal y/o efectiva.

Firmas digitales: Cada transacción en la red Ethereum requiere una firma digital válida, lo que dificulta la falsificación de la autenticación y el acceso no autorizado a los dispositivos. Las firmas digitales proporcionan una forma de verificar la autenticidad de las transacciones y garantizar que solo los usuarios legítimos puedan acceder al dispositivo.

Registro inmutable de transacciones: Todas las transacciones en la red Ethereum se registran de forma inmutable en la cadena de bloques, lo que significa que no se pueden modificar ni eliminar. Esto proporciona un registro transparente y verificable de todas las actividades de autenticación, lo que facilita la detección temprana de intentos de acceso no autorizado.

Control descentralizado: La red Ethereum es descentralizada y distribuida, lo que significa que no está controlada por una sola entidad. Esto hace que sea más difícil para un atacante comprometer la autenticación centralizada y obtener acceso no autorizado a los dispositivos.

3. Conclusiones y recomendaciones

3.1 Conclusiones

La investigación destaca la creciente importancia de abordar las preocupaciones de seguridad en el contexto de IoT, especialmente en dispositivos ubicuos como las cámaras para el hogar. La vulnerabilidad de estos dispositivos puede tener consecuencias significativas para la seguridad y privacidad de los usuarios. De esta forma el proyecto puede ser aplicado a todo dispositivo, Cámaras IoT del sector masivo (hogares) siempre y cuando manejen protocolo HTTP, HTTPS. La mayor dificultad de la implementación estaría en el alcance y código del SDK, que pueda ser compartido por el fabricante y se permita editar para poder indexar códigos de programación blockchain, algunas líneas de JavaScript y solidity.

La tesis demuestra la relevancia y aplicabilidad de la tecnología blockchain, específicamente Ethereum, como una de las soluciones para abordar las vulnerabilidades de seguridad en la autenticación de dispositivos IoT. La capacidad de mantener un registro descentralizado de las transacciones de autenticación que puede mejorar la integridad y confiabilidad del sistema, de esta forma se puede contribuir significativamente a la mitigación de brechas de seguridad en las cámaras IoT del sector hogar. Adicional la investigación también identifica posibles desafíos y limitaciones asociados con la implementación de un modelo de autenticación basado en blockchain, como la escalabilidad, la interoperabilidad y los costos. Estos aspectos deben considerarse en futuros desarrollos y aplicaciones prácticas.

La investigación subraya la necesidad de fomentar la colaboración interdisciplinaria y multisectorial en el desarrollo y aplicación de soluciones de seguridad en IoT. La implementación efectiva del modelo de autenticación basado en blockchain requiere la colaboración entre expertos en seguridad informática, desarrolladores de blockchain, fabricantes de cámaras IoT, reguladores y otros actores relevantes. Al reunir diversas perspectivas, se puede abordar de manera más completa la complejidad de los desafíos de

seguridad y garantizar una solución más robusta y adaptativa a las necesidades del sector hogar.

La limitación en la capacidad de procesamiento de transacciones en la red de Ethereum representa un desafío significativo para la integración de dispositivos IoT. Para superar esta barrera, es esencial seguir lo establecido por la tecnología blockchain, que incluye el particionamiento de la blockchain para mejorar la escalabilidad. Además, es importante considerar las limitaciones de los procesadores utilizados en los dispositivos IoT y los posibles riesgos de seguridad relacionados con la capacidad limitada para inscribir información privada en estos dispositivos. Estos desafíos deben abordarse de manera efectiva para asegurar el rendimiento, la eficiencia y la seguridad en la interacción entre los dispositivos IoT y la red blockchain.

Por otra parte, la interoperabilidad, juega un papel importante integrar cámaras IoT con tecnología blockchain requiere compatibilidad entre diferentes tecnologías y estándares. Asegurar que todos los dispositivos puedan comunicarse y operar de manera coherente puede ser un desafío técnico significativo. Adicionalmente las limitaciones de recursos en las cámaras IoT generalmente tienen limitaciones en términos de procesamiento, memoria y almacenamiento. Integrar tecnología blockchain, que puede ser intensiva en recursos, puede sobrecargar estos dispositivos.

3.2 Recomendaciones

Una de las recomendaciones es que el modelo de autenticación con base en blockchain desarrollado podría tener aplicaciones más amplias en otros contextos de seguridad informática y dispositivos IoT, no limitándose solo a cámaras del hogar. Esto podría abrir oportunidades para futuras investigaciones y desarrollos en diversos campos.

Se recomienda como trabajo a futuro, desarrollar e implementar en mayor proporción cadenas de bloques a las tecnologías IoT, logrando expandir la tecnología blockchain como mecanismo de seguridad pasando de entornos locales a nubes públicas y redes WAN.

A. Anexo: Instalación de aplicaciones.

En este apartado se relaciona el detalle y paso a paso de las instalaciones correspondientes a las herramientas necesarias para la ejecución y prueba de funcionamiento de la tesis de grado.

Bibliografía

1] J. M. Norman, «historyofinformation.,» 06 07 2022. [En línea]. Available: https://www.historyofinformation.com/detail.php?id=3411. [Último acceso: 16 10 2022].

2] J. Gil Herrera, «Biblioteca Virtual Miguel De Cervantes,» 2013. [En línea]. Available: https://www.cervantesvirtual.com/obra/configuration-and-management-in-internet-of-things-middleware-environments-configuracion-y-administracion-de-middleware-para-el-internet-de-las-cosas-856463/. [Último acceso: 30 septiembre 2022].

3] «Statista,» mayo 2022. [En línea]. Available: https://es.statista.com/estadisticas/517654/prevision-de-la-evolucion-de-los-dispositivos-conectados-para-el-internet-de-las-cosas-en-el-mundo/#statisticContainer. [Último acceso: 14 septiembre 2022].

4] «Dahua tecnology,» 28 06 2022. [En línea]. Available: https://www.dahuasecurity.com/support/cybersecurity/details/1017. [Último acceso: 16 10 2022].

5] «Redhat,» [En línea]. Available: https://www.redhat.com/es/topics/internet-of-things/what-is-iot. [Último acceso: 15 octubre 2023].

6] A. S. Tanenbaum, «Academia.edu,» 2003. [En línea]. Available: https://www.academia.edu/22813409/Redes_de_Computadoras_4ta_Edici%C3%B3n_Andrew_S_Tanenbaum_bye_Axedrez. [Último acceso: 07 03 2022].

7] U. i. d. valencia, «universidadviu,» 09 10 2018. [En línea]. Available: https://www.universidadviu.com/int/actualidad/nuestros-expertos/redes-de-datos-todo-lo-que-hay-que-saber-sobre-ellas. [Último acceso: 15 10 2022].

8] J. C. Cadavid Parra, «America Comunicaciones,» 16 noviembre 2017. [En línea]. Available: https://www.americacomunicaciones.com/videovigilancia-historia/. [Último acceso: 07 03 2022].

«ieeexplore,» 2014 05 23. [En línea]. Available: https://ieeexplore.ieee.org/document/6850855/references#references. [Último acceso: 15 10 2022]. [9]

«Red Hat,» 08 01 2019. [En línea]. Available: https://www.redhat.com/es/topics/internet-of-things/what-is-iot. [Último acceso: 16 10 2022]. [10]

D. Evans, «Cisco.com,» abril 2011. [En línea]. Available: https://www.cisco.com/c/dam/global/es_mx/solutions/executive/assets/pdf/internet-of-things-iot-ibsg.pdf. [Último acceso: 30 septiembre 2022]. [11]

«Redhat,» [En línea]. Available: https://www.redhat.com/es/topics/internet-of-things/what-is-iot. [Último acceso: 15 10 2023]. [12]

C. point, «checkpoint,» [En línea]. Available: https://pages.checkpoint.com/iot-protect-solution-brief.html. [Último acceso: 15 10 2022]. [13]

P. Jara Werchau y P. Nazar, «Universidad tecnologica nacional,» 2013. [En línea]. Available: https://cmapspublic2.ihmc.us/rid=1NG37M7DF-P60JX9-26BS/standard_802_11.pdf. [Último acceso: 15 10 2022]. [14]

«Pearsonitcertification,» 9 junio 2009. [En línea]. Available: https://www.pearsonitcertification.com/articles/article.aspx?p=1329709&seqNum=5. [Último acceso: 07 03 2022]. [15]

kaspersky, «latam kaspersky,» [En línea]. Available: https://latam.kaspersky.com/resource-center/definitions/wep-vs-wpa. [Último acceso: 15 10 2022]. [16]

A. Freda, «Avg,» 3 06 2022. [En línea]. Available: https://ieeexplore.ieee.org/abstract/document/5234856. [Último acceso: 15 10 222]. [17]

«NXP Community,» [En línea]. Available: https://community.nxp.com/t5/Wireless-Connectivity-Knowledge/802-11-Wi-Fi-Basic-concepts/ta-p/1124409. [Último acceso: 16 11 2023]. [18]

«Hikvision,» [En línea]. Available: https://www.hikvision.com/es-co/products/IP-Products/Network-Cameras/Wi-Fi-Series/DS-2CV2Q21FD-IW/. [Último acceso: 16 11 2023]. [19]

[20] IEEE, «Ieee xplore,» 17 agosto 2007. [En línea]. Available: https://ieeexplore.ieee.org/document/4293165. [Último acceso: 07 03 2022].

[21] A. Montoya Ros y A. Zamora Gómez, «sistema de autenticacion basado en blockchain para la gestion de billetes en un entorno de transporte inteligente,» 2021.

[22] chakray, «chakray.com,» [En línea]. Available: https://www.chakray.com/es/blockchain-seguridad-iot-lo-saber/. [Último acceso: 25 septiembre 2022].

[23] Ethereum, «Ethereum.org,» 14 10 2022. [En línea]. Available: https://ethereum.org/es/what-is-ethereum/. [Último acceso: 16 10 2022].

[24] «solidity,» [En línea]. Available: https://soliditylang.org/about/. [Último acceso: 06 11 2023].

[25] «Github,» [En línea]. Available: https://github.com/NomicFoundation/hardhat/tree/hardhat%402.18.1. [Último acceso: 06 11 2023].

[26] S. M. RIBERO CORZO y Y. A. PRIETO GUERRERO, «IDENTIFICACION DE RIESGOS EN LA SEGURIDAD DE LA INFORMACION DE,» 2021. [En línea]. Available: https://repository.ucatolica.edu.co/bitstream/10983/26242/1/Identificaci%C3%B3n%20de%20riesgos%20en%20la%20seguridad%20de%20la%20informaci%C3%B3n%20de%20c%C3%A1maras%20de%20vigilancia%20domesticas%20en%20entornos%20IOT.pdf. [Último acceso: 30 septiembre 2022].

[27] «Repository ucatolica,» julio 2020. [En línea]. Available: https://repository.ucatolica.edu.co/bitstream/10983/26242/4/ArticuloIEEE.pdf. [Último acceso: 24 noviembre 2022].

[28] «repository unipiloto,» 2020. [En línea]. Available: http://repository.unipiloto.edu.co/bitstream/handle/20.500.12277/10478/PROTOTIPO%20DE%20UN%20SISTEMA%20DE%20MONITOREO%20Y%20VIDEOVIGILANCIA%20PARA%20EL%20HOGAR%20CON%20EL%20ENFOQUE%20DE%20IOT.pdf?sequence=1. [Último acceso: 24 Noviembre 2022].

«IEEEXPLORE,» 2023. [En línea]. Available: https://ieeexplore.ieee.org/document/10084957. [Último acceso: 28 04 2024].

A. ALBERRACIN ESTRADA, D. E. SOTO DURAN, J. GIL HERRERA y A. F. A. VARGAS, «Revista Ibérica de Sistemas e Tecnologias de Informação,» 04 2022. [En línea]. Available: https://www.risti.xyz/issues/ristie49.pdf. [Último acceso: 16 06 2024].

«Cointelgraph,» 29 09 2020. [En línea]. Available: https://cointelegraph.com/news/blockchain-technology-now-powers-a-privacy-focused-security-camera. [Último acceso: 28 04 2024].

C. A. Sanchez Venegas y J. E. Salinas Santiago, «Repositorio escuela colombia de ingenieria Julio Garavito,» 07 2023. [En línea]. Available: https://repositorio.escuelaing.edu.co/bitstream/handle/001/2779/S%C3%A1nchez%20Venegas%20%2C%20Carlos%20Andr%C3%A9s.-2023.pdf?sequence=1&isAllowed=y. [Último acceso: 03 06 2024].

«CTIC Centro Tecnológico,» 05 02 2021. [En línea]. Available: https://www.fundacionctic.org/es/actualidad/blockchain-iot-ahora-si. [Último acceso: 03 06 2024].

«Asmag security,» 04 12 2022. [En línea]. Available: https://www.asmag.com/rankings/. [Último acceso: 12 diciembre 2022].

R. H. SAMPIERI, METODOLOGIA DE LA INVESTIGACION, mexico: McGRAW-HILL / INTERAMERICANA EDITORES, S.A. DE C.V., 2014.

Hikvision, «Hikvision.com,» [En línea]. Available: https://www.hikvision.com/content/dam/hikvision/products/S000000001/S000000002/S000000003/S000000801/OFR000386/M000002452/User_Manual/UD28967B-A_Network-Camera_User-Manual_5.7.20_20221215.PDF. [Último acceso: 23 10 2023].

C. V. a. Exposures, «cve mitre,» [En línea]. Available: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=hikvision. [Último acceso: 24 10 2023].

«National Vulnerability Database,» 05 05 2017. [En línea]. Available: https://nvd.nist.gov/vuln/search/results?form_type=Basic&results_type=overview&qu

ery=CVE-2017-7921&search_type=all&isCpeNameSearch=false. [Último acceso: 25 10 2023].

[39] «National Vulnerability Database,» 05 05 2017. [En línea]. Available: https://nvd.nist.gov/vuln/search/results?form_type=Basic&results_type=overview&query=CVE-2017-7923&search_type=all&isCpeNameSearch=false. [Último acceso: 25 10 2023].

[40] «National Vulnerability Database,» 22 09 2021. [En línea]. Available: https://nvd.nist.gov/vuln/search/results?form_type=Basic&results_type=overview&query=CVE-2021-36260&search_type=all&isCpeNameSearch=false. [Último acceso: 25 10 2023].

[41] Tenable, «tenable.com,» [En línea]. Available: https://es-la.tenable.com/products/nessus. [Último acceso: 25 10 2023].

[42] «First Improving Security Together,» [En línea]. Available: https://www.first.org/cvss/. [Último acceso: 28 04 2024].

[43] «Acis,» [En línea]. Available: https://acis.org.co/portal/content/noticiasdeinteres/video-portero-ip-la-nueva-soluci%C3%B3n-tecnol%C3%B3gica-que-busca-proteger-los-hogares-colombianos#:~:text=El%20uso%20de%20la%20tecnolog%C3%ADa,en%20los%20%C3%BAltimos%20cinco%20a%C3%B1os.. [Último acceso: 4 06 2022].

[44] BBC, «BBC MUNDO,» 03 marzo 2020. [En línea]. Available: https://www.bbc.com/mundo/noticias-51681204. [Último acceso: 02 octubre 2022].

[45] J. Moranga y N. Perez, «CSIRT,» 04 AGOSTO 2020. [En línea]. Available: https://www.csirt.gob.cl/reportes/an2-2020-13/. [Último acceso: 04 09 2022].

[46] E. Research, «welivesecurity,» 19 mayo 2019. [En línea]. Available: https://www.welivesecurity.com/la-es/2019/05/02/vulnerabilidad-camaras-d-link-permite-atacante-espiar-transmisiones/. [Último acceso: 08 06 2022].

[47] Teknofilo, «Teknofilo,» 18 08 2021. [En línea]. Available: https://www.teknofilo.com/una-vulnerabilidad-permite-acceder-a-los-videos-de-millones-de-camaras-del-hogar-2/. [Último acceso: 10 06 2022].

IBM, «IBM,» [En línea]. Available: https://www.ibm.com/es-es/topics/what-is-
48] blockchain. [Último acceso: 08 04 2022].

A. Montoya ros, «Universidad Alicante,» septiembre 2021. [En línea].
49] Available:
https://rua.ua.es/dspace/bitstream/10045/118148/1/Sistema_de_autenticacion_basado_en_blockchain_para_la_ges_Montoya_Ros_Adrian.pdf. [Último acceso: 24 noviembre 2022].

«IeBS,» IRENE GONZALES, 02 SEPTIEMBRE 2022. [En línea]. Available:
50] https://www.iebschool.com/blog/las-mejores-plataformas-blockchain-para-el-desarrollo-de-aplicaciones-financieras-tecnologia/. [Último acceso: 30 SEPTIEMBRE 2022].

«Electronics,» [En línea]. Available: https://www.mdpi.com/2079-
51] 9292/9/3/484. [Último acceso: 28 04 2024].

N. A. ARIAS SILVA, «repository unad,» 2019. [En línea]. Available:
52] https://repository.unad.edu.co/bitstream/handle/10596/33326/naariass.pdf?sequence=1&isAllowed=y. [Último acceso: 30 Septiembre 2022].