

Blockchain Aplicado a Retos de Ciberseguridad

Blockchain Applied to Cybersecurity Challenges

Integrantes:

Andrés Felipe Pardo Mesa

Fabián Augusto Ardila Rodríguez

Director:

Daniel Orlando Díaz López

Escuela Colombiana de Ingeniería Julio Garavito

Ingeniería de Sistemas

Proyecto de Grado

Bogotá

2019

1. CONTENIDO

CONTENIDO	1
2. RESUMEN	3
3. INTRODUCCIÓN	4
4. ESTADO DEL ARTE	7
5. OBJETIVOS DEL PROYECTO	9
Objetivo General	9
Objetivos Específicos	9
6. PUBLICACIONES	10
7. CREAR UN DAPP EN ETHEREUM	11
8. ANÁLISIS DE LA SOLUCIÓN: “Seguridad y Privacidad en una Smart Home”	16
Generalidades de la solución	19
Características más relevantes de la solución	19
Posibles ataques a la solución propuesta	19
9. ANÁLISIS DE LA SOLUCIÓN: “Control de Acceso en IoT utilizando Blockchain”	21
Caso 1: ACC llamado por el Sujeto	24
Caso 2: ACC llamado por el Objeto	24
Características más relevantes de la solución:	25
Posibles ataques a la solución propuesta:	25
10. PROPUESTA DE SOLUCIÓN	26
Arquitectura de la solución	26
Dispositivos IoT	26
Dispositivos Centinela	27
SIEM Distribuido	27
Inteligencia Interna o Externa	27
Añadir eventos de seguridad a la Blockchain	27
Consumir la Blockchain para detectar ataques distribuidos	27
Detección de ataques bajo escenarios hostiles	27
Auditar un incidente de seguridad	28
Escalar la Infraestructura de IoT segura	28

11. LOGROS DEL PROYECTO	29
12. CONCLUSIONES Y TRABAJOS FUTUROS	30
13. BIBLIOGRAFÍA	31

2. RESUMEN

El contexto tecnológico en el que vivimos hoy en día permite a las compañías la implementación de novedosas soluciones capaces de mejorar la eficiencia y eficacia en los procesos de negocio claves en su operación y de esta forma dar cumplimiento a los objetivos que componen la estrategia organizacional. Sin embargo, la incursión tecnológica en ámbitos donde antes no era considerada ha dado paso a nuevas amenazas enfocadas en afectar el activo intangible más importante de las compañías: los datos.

La Ciberseguridad es el área que se encarga de proteger los datos que residen en sistemas informáticos, garantizando que estos no sean víctimas de ataques informáticos. Por este motivo el concepto se ha vuelto de suma importancia dentro de las compañías, y es por esto que cada día tanto la academia como la industria deciden invertir más en investigaciones y soluciones que garanticen el cumplimiento de este objetivo de la forma más adecuada.

A partir de la aparición de Bitcoin, que desató el auge de las criptodivisas, se permitió visualizar de una forma clara una tecnología disruptiva que permite solucionar problemas de confianza, sincronización, trazabilidad e inmutabilidad que las arquitecturas tradicionales que se usan con mayor frecuencia no resuelven de forma adecuada.

Teniendo en cuenta los problemas de seguridad informática emergentes el proyecto “Blockchain Aplicado a Retos de Ciberseguridad” tiene como objetivo investigar y proponer una solución que haga uso de la tecnología de blockchain a el problema de seguridad con dispositivos IoT (Internet of Things) los cuales al estar conectados dentro de una red aumentan la superficie de ataque permitiendo así vulnerabilidades que antes no fueron contempladas.

La solución que se propone es la implementación de un sistema para la gestión de eventos de seguridad recolectados por dispositivos IoT llamado “Centinela” y regulados dentro de una red Blockchain privada.

3. INTRODUCCIÓN

Blockchain es una red descentralizada que funciona como mecanismo de registro de transacciones de tal manera que vuelve inmodificable la información debido a la naturaleza de los algoritmos de consenso que usa. Todos los nodos que hacen parte de una determinada red mantienen una misma secuencia de datos anidados entre sí, que, si se llegasen a ser modificados, generaría el rechazo de la transacción en proceso. En este proyecto se presenta la propuesta de una arquitectura de solución basada en esta tecnología disruptiva, como lo es Blockchain, permitiendo abordar características de seguridad como autorización, no repudio, auditoría e integridad en la información de las transacciones registradas.



Imagen 1 - Tipos de Arquitectura

Blockchain en pocas palabras es una *base datos* que registra un libro de cuentas con diferentes transacciones.



Imagen 2 - Representación Libro de Cuentas

Lo interesante de esta base de datos es que se distribuye la información a través de todos los nodos participantes de la red, manteniendo así en toda la misma versión del libro de cuentas. Cada vez que ocurre un cambio en alguno de los nodos, todos perciben el cambio y validan que sea correcto, una vez se acepta el cambio queda registrado en un bloque que es inmutable e inmodificable.

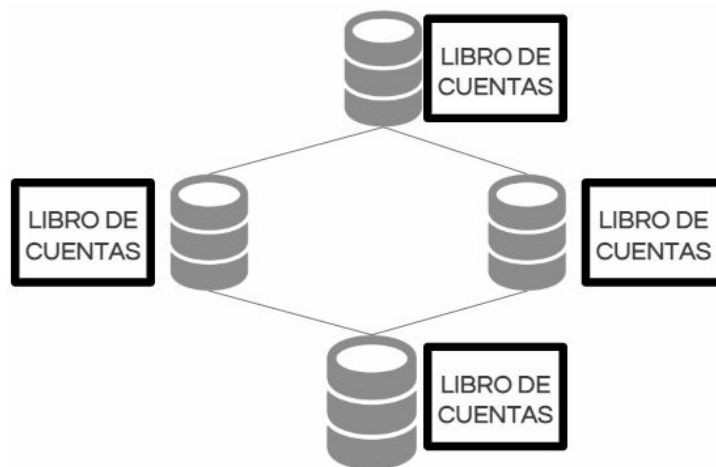


Imagen 3 - Estructura de nodos representados como Libros de Cuentas

Para que la red de Blockchain interactúe como se espera se definirán las reglas que se tendrán en cuenta al momento de solicitar una acción con relación a cierta información; a dichas definiciones se les llamará *Smart Contracts* y allí se especificarán las condiciones con las que los usuarios serán regidos al intento de manipulación o creación de información dentro de la red de Blockchain, es decir las *políticas de acceso*. En la siguiente imagen podemos ver una representación gráfica de ejemplos de *Smart Contracts* propuestos en [1] para realizar el control de acceso lógico a un conjunto de dispositivos IoT.

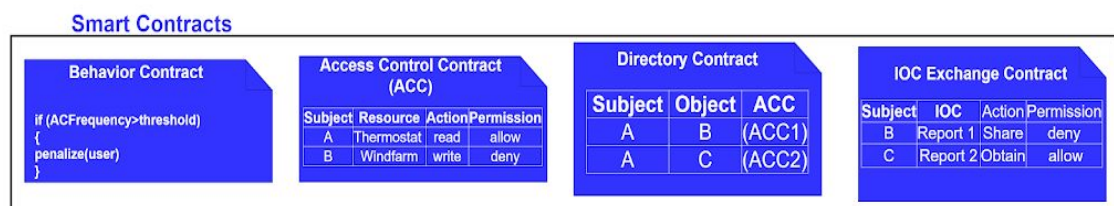


Imagen 4 - Ejemplos de Smart Contracts propuestos

Por la naturaleza descentralizada de una red de Blockchain encontramos que sin intermediarios es necesario el uso de mecanismos de consenso para llegar a los acuerdos que determinen la veracidad de la información de las transacciones que deben ser registradas. Actualmente existe una gran variedad de algoritmos de consenso que difieren entre sí por los criterios de decisión que otorga a cada uno de los nodos de la red. Algunos de estos algoritmos son:

- *Proof of Work (PoW)*
- *Proof of Stake (PoS)*
- *Delegated Proof of Stake (DPoS)*
- *Proof of Importance (PoI)*
- *Byzantine Fault Tolerance (BFT)*

- *Federated Byzantine Agreement (FBA)*

Ethereum es una Blockchain que definió su propia criptomoneda y su objetivo es bajo esta tecnología tener interacción por medio de los smart contracts de tal manera que se les pueda dar un uso funcional, que permita creación de aplicaciones que sean ejecutadas sin posibilidad de fallo del servicio ofrecido, censuramiento, fraude o interferencia de intereses de terceras partes.

Con su IDE oficial llamado *Remix*, *Ethereum* permite realizar la compilación de *Smart Contracts* con su respectivo despliegue a una *red de Blockchain* ya sea pública o de prueba, al igual también ofreciendo una interfaz por medio de la cual sea posible interactuar con los servicios ofrecidos de los *Smart Contracts*, detallando la información de sus respectivas entradas y salida por cada *función*.

El aumento de uso de los dispositivos conectados a internet es ahora mucho más evidente que hace algunos años. Ahora los lugares en los que comúnmente las personas permanecen, como por ejemplo, la casa, el trabajo o centros comerciales están rodeados de dichos dispositivos, los cuales constantemente capturan la información del medio para que luego sea administrada según el proceso que deba ser aplicado en cada caso sobre los datos. Según [2] las organizaciones han adoptado en gran medida el uso de dispositivos IoT, en especial empresas pequeñas y medianas, las cuales han tenido un aumento de 20% en la adopción de estos equipos entre los años 2014 y 2017, mostrando de esta forma cómo el auge es significativo. Además de presentar un crecimiento notorio en la adopción de dispositivos conectados en internet, según [3] en el año 2025 se planea tener más de 75.4 millones de dispositivos conectado a lo largo de internet.

Derivado del incremento en el uso de dispositivos IoT, el flujo de datos a través de la red será una situación inevitable que se tendrá que afrontar y con las soluciones actuales de seguridad probablemente no se logren abarcar la mayoría de los riesgos asociados a ataques a dispositivos IoT vulnerables. Al descubierto se quedará toda la información que pase a través de medios inseguros o que sea emitida por dispositivos IoT que hayan sido accedidos por agentes con intenciones maliciosas.

Se han puesto en práctica distintas soluciones a esta problemática, algunas de ellas con falencias en el diseño de su arquitectura, ya que están basadas bajo principios de centralización; lo que genera una vulnerabilidad asociada a un punto único de fallo.

4. ESTADO DEL ARTE

Como parte de la investigación realizada sobre soluciones Blockchain aplicadas a retos de la ciberseguridad en las redes de dispositivos IoT, se encontró por un lado una solución para gestionar el acceso a los dispositivos IoT de una smart home, “Blockchain for IoT Security and Privacy: The Case Study of a Smart Home” [4], paper que refleja la investigación realizada en la School of Computer Science and Engineering The University of New South Wales en Sydney, Australia en marzo del año 2017; en este paper se creó una arquitectura basada en blockchain, pero elimina uno de los atributos más importantes de esta tecnología: el consenso, el consenso es el que permite que la red blockchain se caracterice por su confiabilidad ya que independientemente el método utilizado, (proof of work, proof of stake, proof of authority), asegura que la información que está siendo distribuida a través de todos los nodos es la información correcta y validada por todos la mayoría de los nodos en la red.

Por otro lado se encontró un paper denominado “Smart Contract-Based Access Control for the Internet of Things” [1] publicado el 13 de febrero del 2018 en la revista de la Cornell University Library; en este paper se presenta un framework de control de acceso para dispositivos IoT basado en smart contracts, que valida la autorización de un Sujeto de forma estática por medio de políticas definidas previamente para un dispositivo IoT que será el representante de los demás recursos que se quieran consultar, y al mismo tiempo valida el comportamiento de este sujeto dentro de la red (por frecuencia de peticiones realizadas) y penaliza aquellos sujetos que incumplan con un límite definido de peticiones.

Además, en [5] los autores presentan una solución de seguridad llamada GHOST, que se comporta como un gateway, proporcionando dos funciones principales: i) proporcionar capacidades de conectividad para dispositivos IoT dentro de un hogar inteligente y ii) implementar mecanismos para preservar la seguridad y privacidad del usuario. GHOST tiene una arquitectura basada en software de varias capas que también está integrada con diferentes componentes adicionales. La primera capa se encarga de recopilar los datos que conmutan a través de la red, agrupando paquetes por protocolo y clasificando el tráfico. Una vez que los datos se almacenan y clasifican, la segunda capa utiliza una red neuronal aleatoria para identificar perfiles para el comportamiento común de los dispositivos considerando los datos clasificados y la comunicación entre pares de dispositivos. Luego, la tercera capa, usando funciones definidas en un contrato inteligente, usa los perfiles creados previamente para desarrollar una evaluación de riesgos sobre el flujo. Finalmente, la cuarta capa presenta informes, muestra un rastro de las acciones de seguridad y recibe comentarios de los usuarios, dándoles la posibilidad de tener el control sobre los dispositivos IoT domésticos inteligentes y los datos que manejan.

A su vez, [5] propone una arquitectura de Sistema de detección de intrusos (IDS) aplicable a los ecosistemas IoT. Esta arquitectura IDS se compone de una cascada de etapas para identificar una amenaza de seguridad. Primero, un filtro de eventos elimina datos ruidosos y redundantes y crea un formato único, es decir, eventos de seguridad, para presentar los datos. A continuación, los eventos se procesan y se comparan con los eventos de correlación, lo que le da al administrador la capacidad de agregar nuevos eventos de correlación o establecer una suscripción para recibir análisis de eventos de correlación. Una vez que los eventos se han clasificado dentro de la categoría normal o anormal, la siguiente etapa automáticamente toma una acción para mitigar el ataque (si es necesario). Por último, los eventos se almacenan en una base de datos que puede ser auditable.

Los autores de [6] presentan un marco de trabajo para monitorear eventos de seguridad en tiempo real en un entorno IoT, que contiene cinco capas diferentes. En la primera capa, los dispositivos IoT recopilan y publican eventos, lo que permite que otros módulos se suscriban a esos eventos para recuperar información relevante. La segunda capa se encarga de re-enviar información al agente de eventos. La tercera capa, a su vez, recopila los eventos disponibles en el agente de eventos y realiza un análisis de seguridad que puede ser compatible con motores externos confiables. La cuarta capa almacena la información procesada utilizada por los algoritmos de aprendizaje automático o los procesos de auditoría. Finalmente, la última capa contiene un panel que informa sobre los problemas de seguridad en tiempo real.

5. OBJETIVOS DEL PROYECTO

Objetivo General

Desarrollar un Sistema de Gestión de Eventos de Seguridad que le permita a un Centinela recolectar y almacenar eventos en una red Blockchain para luego ser clasificados por sistemas SIEM conectados a los mineros dentro de la red.

Objetivos Específicos

- Construir un marco teórico y establecer el estado del arte de la tecnología Blockchain, identificando cuáles son sus ventajas y desventajas y cuáles han sido sus aplicaciones.
- Comprender y compartir conocimiento sobre qué es una DApp (Decentralized Application) y su diferencia con las aplicaciones convencionales además de aprender las bases del desarrollo en Blockchain.
- Investigar las propuestas existentes sobre el uso de Blockchain en ecosistemas IoT identificando sus vulnerabilidades y oportunidades de mejora.
- Diseñar la arquitectura de un SIEM distribuido que utiliza dispositivos IoT (centinelas) para recolección de eventos en ambientes IoT.
- Validar la solución propuesta en diferentes casos de uso relacionados a la gestión de eventos de seguridad en entornos IoT distribuidos.
- Realizar mediciones de desempeño de la solución en aspectos como el tiempo de creación de bloques, la selección del nodo minero en la blockchain, la eficiencia del algoritmo de consenso, entre otros.

6. PUBLICACIONES

- *A. Pardo, F. Ardila, D. Díaz Lopez, and F. Gómez Mármol . BSIEM-IoT: A Blockchain-based and Distributed SIEM for the Internet of Things. 17th International Conference on Applied Cryptography and Network Security (ACNS). 1st International Workshop on Application Intelligence and Blockchain Security (AIBlock). Bogota, Colombia. 2019.*

7. CREAR UN DAPP EN ETHEREUM

Ethereum es una plataforma descentralizada basada en Blockchain que permite la creación de “Smart Contracts”, encargados de la definición de condiciones lógicas (backend) de una aplicación. Una aplicación que utiliza un “Smart Contract” desplegado en una red Blockchain para definir las reglas de negocio de su sistema es conocida como una DAPP (Decentralized Application).

Para la creación de una DAPP en Ethereum es necesario contar con una dirección dentro de la blockchain (wallet), la cual permite realizar transacciones dentro de la red Blockchain. Para esto existen diversas herramientas que permiten la creación de una “wallet” (e.g. Metamask, MyEtherWallet).

En este caso utilizaremos Metamask para crear una wallet

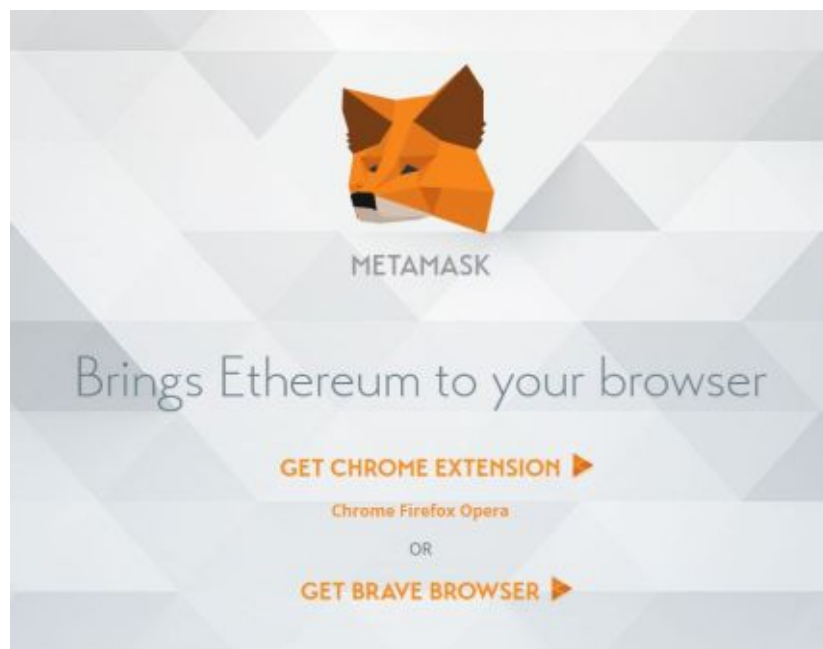


Imagen 5 - Vista principal de la herramienta Metamask

Una vez se cuenta con una wallet, clientes como “Geth” que permiten crear un nodo dentro de la red Blockchain, a este nodo se le puede asociar una cuenta wallet para que pueda realizar transacciones.

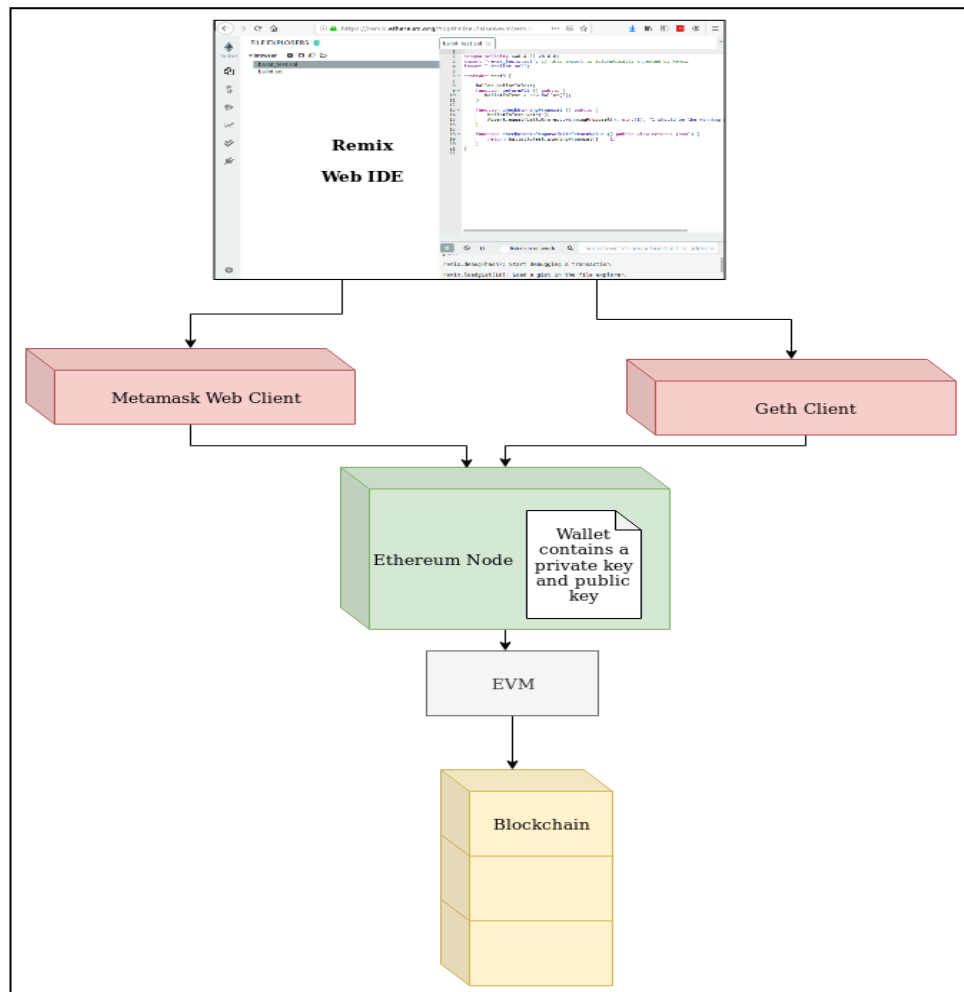


Imagen 6 - Arquitectura propuesta de la DAPP

Una vez se cuenta con una conexión a un nodo y una “wallet”, se debe desarrollar el “Smart Contract” que defina las reglas de negocio que implementa nuestra aplicación. El desarrollo de un Smart Contract dentro de Ethereum se hace usando el lenguaje de programación llamado Solidity. Una herramienta útil para la prueba de estos contratos es Remix, esta herramienta es el IDE online oficial de Solidity que permite interactuar con un nodo de Ethereum de una forma más intuitiva y sencilla.

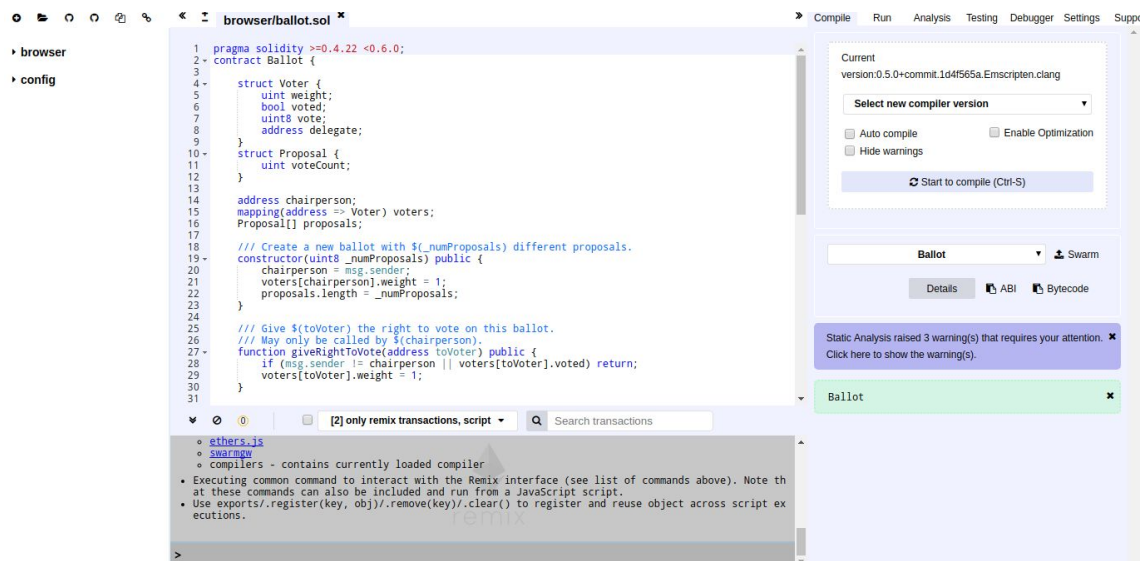


Imagen 7 - Remix, IDE online oficial de Solidity

Una vez se desarrolla el Smart Contract en Solidity se debe compilar para evitar que el contrato pueda ser rechazado por la máquina virtual de Ethereum al encontrar errores en la sintaxis. Luego de que la compilación es aceptada se debe generar la transacción de despliegue de contrato, cuando la transacción es minada y validada se crea un bloque y se le asigna al contrato una dirección que permite su identificación dentro de la Blockchain.

Ahora que tenemos desplegado nuestro contrato debemos consumirlo a través de un cliente web, para esto hacemos uso de web3, el API de Javascript que permite conectarnos con un nodo y por medio de la configuración del ABI (Application Binary Interface) objeto tipo JSON que describe atributos y métodos de un "Smart Contract" para poder consumir en un cliente externo. La configuración básica de un cliente web3 se muestra a continuación:

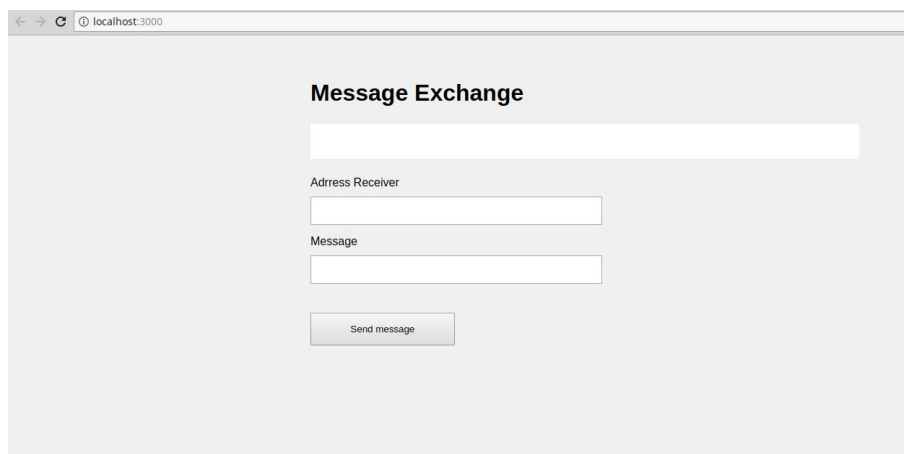


Imagen 8 - Configuración básica de un cliente web3

Los elementos básicos para la configuración de un cliente web3 son:

- **Agregar web3 al proyecto:** Para empezar a utilizar la API de web3, inicialmente debe agregar el paquete de web3 al repositorio donde se esté desarrollando el proyecto, para hacer esto web3 provee diferentes medios (npm, meteor, javascript)
- **Instancia el “provider”:** Como se puede ver en la *Imagen 1.1* se está haciendo referencia a un “*Proveedor Actual*”, en dónde se detectará el origen de dónde se tomarán los servicios, es decir, de cuál librería, así el cliente sabrá de dónde tomar referencia para poder ejecutar los comandos satisfactoriamente.
- **Instancia del contrato:** en la segunda línea de código que podemos observar en la *Imagen 1.1* se declara una variable *contract* que tiene como objetivo almacenar las funciones de un contrato definidas en el ABI que es pasado como parámetro en el método *contract()*. Esta configuración permitirá saber cómo consumir servicios de los *Smart Contracts* para que el cliente tenga un correcto funcionamiento.
- **Referencia del contrato en la red blockchain:** en la tercera línea de código que podemos observar en la *Imagen 1.1* ahora se escribirá la dirección que le fué asignada al contrato al momento de ser desplegada dentro de la red *Blockchain*, así ya teniendo la estructura del contrato y conociendo la ubicación del mismo, el consumo de servicios será completamente satisfactorio.

Una vez el contrato ya ha sido desplegado dentro de la red de *Blockchain* en la que se está trabajando, entonces procederemos a realizar una vista con la cual sea más amigable interactuar dentro de la red. Para lo anterior se realizará la creación de un servicio web para la visualización de la operación de la DAPP. para esto debemos crear archivos *.css*, *JavaScript* y *.html*. Una vista sencilla que nos traiga la información estrictamente necesaria del resultado de interacción con *Smart Contracts* se vería algo similar a la siguiente imagen:



The image shows a web browser window with the address bar displaying 'localhost:3000'. The main content area has a title 'Message Exchange' in bold. Below the title is a large, empty text input field. Underneath this is a label 'Address Receiver' followed by a smaller text input field. Below that is a label 'Message' followed by another smaller text input field. At the bottom of the form is a button labeled 'Send message'.

Imagen 9 - Vista inicial de la DAPP

La anterior vista corresponde a una *DAPP* que tiene como propósito el intercambio de mensajes dejando registro dentro de una *Blockchain*. La información que se captura es el mensaje concreto que se recibe/envía y para enviar la solicitud se especifica la dirección a la cual se desea enviar el mensaje.

Los clientes que se han creado anteriormente correspondientes a la vista y el archivo *web3* son ejecutados por medio de un servidor web local llamado *Lite-Server* que permite el despliegue de nuestro servicio de visualización de la *DAPP*. La creación de la *DAPP* de la cual se hace referencia se hizo con este tipo de servidor, pero esto no restringe a que se pueda realizar con cualquier otro tipo de servicio local o externo que cumpla las mismas funciones.

Después de tener todas las configuraciones en orden con respecto a *Smart Contracts*, *vista* y respectivos clientes que administrarán las comunicaciones, entonces se podrá realizar uso de la *DAPP* creada. Esta prueba del uso lo haremos, por cuestiones de facilidad, con un servicio llamado *MetaMask* el cual nos permitirá simular un nodo de *Ethereum* con el cual podremos interactuar con los servicios ofrecidos en el *Smart Contract*.

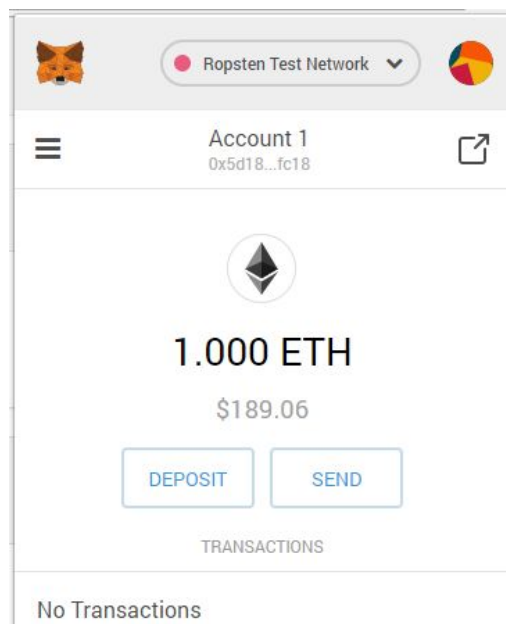


Imagen 9 - Servicio Metamask

Como se puede ver en la Imagen 9, así se visualizará el servicio de *Metamask* una vez hayamos accedido con una cuenta válida.

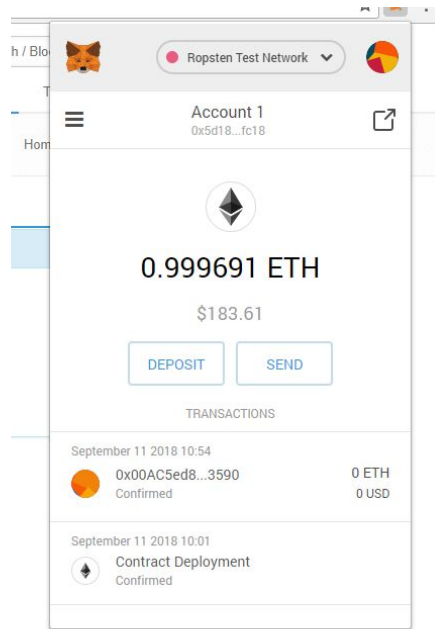


Imagen 11 - Registro de transacción en el servicio Metamask

Cada transacción será necesario aprobarla desde *Metamask*, en donde se confirmarán los costos en *Ether* que cada transacción implica y esto a su vez generará un registro de las acciones tomadas con relación a las transacciones solicitadas.

Cuando la transacción es confirmada como lo podemos observar en la *Imagen 11*, entonces todos los clientes comienzan a capturar los eventos de su respectiva wallet asociada de tal manera que la cuenta a la que fue enviada el mensaje le será notificada y mostrará el texto como lo podemos observar en la *Imagen 12*.

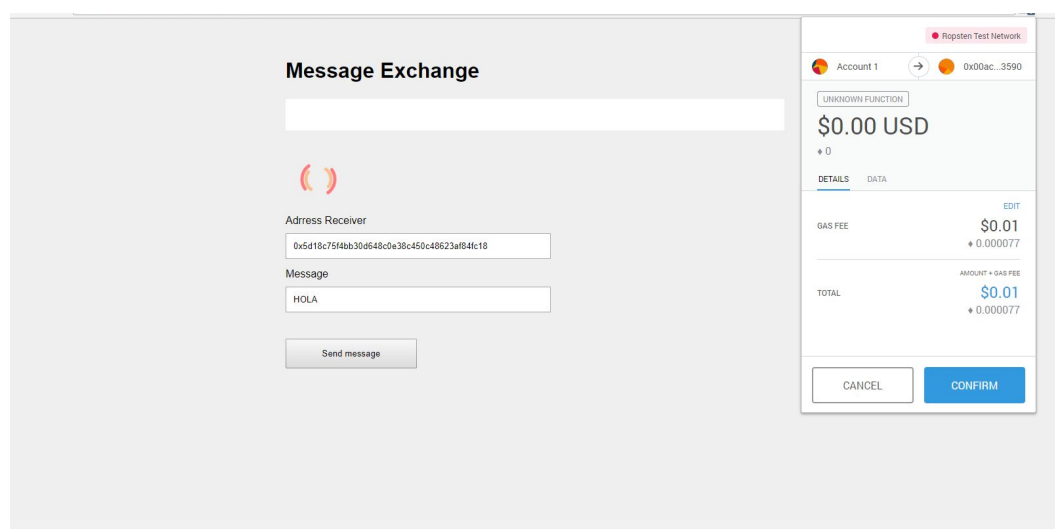


Imagen 12 - Vista inicial de la DAPP al momento de recibir un mensaje

8. ANÁLISIS DE LA SOLUCIÓN: “Seguridad y Privacidad en una Smart Home”

Esta propuesta de solución [4] hace referencia a una implementación de una solución de ciberseguridad para dispositivos *IoT* en donde se parte de una arquitectura como la siguiente:



Imagen 12 - Arquitectura de la solución: “Seguridad y Privacidad en una Smart Home”

Analizando la arquitectura nos encontramos con los siguientes componentes:

- **Proveedor de Servicio:** Corresponde a la estación desde donde se generan y realizan monitoreos de las peticiones relacionada con los dispositivos *IoT*.
- **Almacenamiento en la nube:** Aquí permanece la información recopilada por los dispositivos *IoT* y que dentro del comportamiento normal del sistema se comparte entre cada una de las entidades que pueda hacer uso de dicha información. Algunas de sus características corresponden a:
 - Puede estar integrado con el *miner* o puede ser un dispositivo separado.
 - Usa el método de *First-in-First-out (FIFO)* para almacenar los datos.
 - Guarda los datos de cada dispositivo como un libro de cuentas encadenado al génesis del dispositivo.
- **Cluster Head/Smart Home:** Este componente representa un conjunto de dispositivos *IoT* por medio de los cuales se puede interactuar para realizar acciones con las diferentes entidades. Mantiene una *blockchain* pública donde se encuentra la conjunción de dos listas:
 - **Requesters:** Public Keys (llaves públicas) de los usuarios permitidos para acceder a los datos del *Smart Home*.

- **Requestee:** Public Keys (llaves públicas) de los dispositivos permitidos para ser accedidos.
- **Smart Home Miner:** Correspondiente a una estación de trabajo por medio de la cual se realiza el consenso de una *Blockchain* tradicional, de tal manera que se pueda llegar a un resultado verídico de la información que se planea registrar dentro de la red. Se encarga de mantener (auténtica, autoriza y audita) las comunicaciones internas y externas de la *Smart Home*. Algunas de sus funciones corresponden a:
 - Genera la transacción genesis
 - Distribuye y actualiza llaves
 - Cambia la estructura de las transacciones
 - Gestiona el almacenamiento
- **Dispositivos IoT:** Aparatos electrónicos por medio de los cuales es posible realizar recolección de datos del medio físico y almacenarlos digitalmente para su posterior análisis y gestión correspondiente.

Esta solución plantea una forma de trabajo *jerárquico* que intenta mantener la confianza dentro de la red, pero haciendo uso de un mecanismo similar al sistema Peer-to-Peer de Bitcoin que contiene la característica de arquitectura distribuida pero sin el uso de PoW (Proof of Work) y coins. Para la detección de cambios en la transmisión de transacciones es utilizado un ligero hashing (spongent). Con respecto a la arquitectura propuesta se plantea que la escalabilidad funcione agregando nuevos niveles de *Cluster Heads* con más *Smart Homes* que respondan a las mismas especificaciones de componentes definidos anteriormente.

De acuerdo a la definición de la solución en mención se han generado una serie de instrucciones permitidas dentro del sistema que se definirán a continuación:

- **Store:** generado por dispositivos para guardar información.
- **Access:** generado por el proveedor de servicio o admin para acceder al almacenamiento en la nube.
- **Monitor:** generado por el admin o proveedor de servicio para monitorear periódicamente la información de un dispositivo.
- **Genesis:** adición de un nuevo dispositivo al “Smart Home”.
- **Remove:** quitar dispositivo del *Smart Home*.

La *Blockchain* que se planea utilizar dentro de esta solución tiene una estructura específica de la cual se logra destacar la siguiente información contenida dentro de cada uno de los bloques de la cadena:

- **Block Header:** referencia del bloque anterior.
- **Policy Header:** característica de control y autorización dentro de la cadena, este espacio de información corresponde a ciertos atributos

dentro de sí que permiten ejecutar sus funciones correctamente:

- **Requester:** PK (llave pública) del dispositivo que solicita la transacción, en caso de ser un dispositivo local, entonces registrará el identificador.
- **Requeste for:** Acción solicitada.
- **Device ID:** identificador del dispositivo que se está solicitando acceso dentro de la *Smart Home*.
- **Action:** decisión tomada con respecto a la solicitud, la cual puede corresponder a ser *rechazada* o *aprobada*.

Adicional a los atributos anteriormente mencionados, cada uno de los bloques contiene una secuencia de transacciones que son representadas bajo la siguiente estructura cada una:

- **Previous Transaction/Transaction number:** relación entre transacciones correspondiente a un mismo dispositivo.
- **Device ID:** identificador del dispositivo al que pertenece la transacción.
- **Transaction Type:** tipo de transacción realizada que corresponde a alguna de las *instrucciones permitidas* que anteriormente se han mencionado: *Genesis, Access, Store, Monitor*.
- **Corresponding Multisig Transaction:** transacción guardada sólo si esta viene desde dentro de la *Smart Home*, caso contrario quedará el espacio vacío.

Generalidades de la solución

- La estructura jerárquica de la propuesta y la confianza distribuida hacen que no sea necesario que las transacciones sean "broadcasted" y que además no haya un PoW (Challenge) que consuma energía, la cual es limitada en dispositivos IoT.
- La propuesta utiliza una private BC (existe una por cada *Smart Home*), la cual almacena las transacciones entre la *Smart Home* y los *nodos overlays/dispositivos IoT*. También existe una *Blockchain Pública* conformada por los nodos CH - *Cluster Head* para mantener 2 listados de *Public Keys: Requester y Requestee*.
- La *Blockchain local* es almacenada únicamente en el *storage* del miner.
- El *Policy Header* es quien tiene las reglas de autorización de transacciones. Contiene sujeto, acción, objeto y permiso.
- El *Policy Header* es análogo a un *Smart Contract*
- El miner administra el *Blockchain local*.
- El miner es el que genera, distribuye y revoca PK's (*llaves públicas*) hacia los CH (*Cluster head*),
- Los dispositivos IoT dentro de la *Smart Home* pueden comunicarse con otros dispositivos directamente (sin necesidad de pasar por el Miner). Sin embargo, es el Miner quien asigna una shared key a los dispositivos para que se puedan comunicar entre ellos.
- Cuando una persona tiene dos casas puede tener un mismo miner para

administrar ambas. La unión lógica de las dos casas se denomina un *Shared Overlay*.

Características más relevantes de la solución

- Los métodos tradicionales pueden ser costosos en términos de energía y procesamiento. Por lo general estos métodos son centralizados y dificultan la escalabilidad en sistemas de IoT, e implican un SPF (Single Point of Failure - Único punto de fallo).
- Los métodos utilizados utilizan una técnica de ofuscamiento de datos para la seguridad lo cual dificulta la posibilidad de ofrecer servicios personalizados.
- Debido a las especificaciones de funcionamiento de esta solución, no será posible tener el riesgo de un ataque DDoS (Distributed Denial of Service).
- El *linking attack* corresponde a un ataque que realiza un seguimiento de la información uniéndose a toda la secuencia, pero como los bloques están cada uno unido entre sí en una relación uno a uno, entonces no se tendrá el riesgo de este tipo de ataque.

Posibles ataques a la solución propuesta

De acuerdo con lo analizado del funcionamiento de la solución se ha plantado una vulnerabilidad que si llegara a ser explotada podría causar graves incidentes de seguridad que comprometería los datos y la información de la *Smart Home*. El *Miner* corresponde al eslabón más débil dentro de toda la cadena de seguridad ya que es la entidad que maneja las decisiones y la administración de solicitudes, lo cual lo convierte en un componente importante que puede ser atacada de manera similar a como se llevan a cabo ataques tradicionales sobre un servidor común y corriente.

9. ANÁLISIS DE LA SOLUCIÓN: “Control de Acceso en IoT utilizando Blockchain”

En [1] se propone un Framework para realizar control de acceso para dispositivos IOT basado en Smart Contracts desplegados en Ethereum. Este sistema basado en blockchain ofrece las características de integridad en los datos almacenados en los dispositivos IoT, no repudio, autorización y confiabilidad.

A continuación, presentamos la arquitectura del framework descrito en [1]. Para comprender el framework estudiado inicialmente se deben definir algunos conceptos clave dentro de la arquitectura del framework que permitirán un mejor entendimiento al lector:

- *Red IoT*: Una red peer to peer es definida como una red Blockchain donde existen diferentes tipos de dispositivos como: servidores, dispositivos de almacenamiento, dispositivos de usuario y dispositivos IoT.

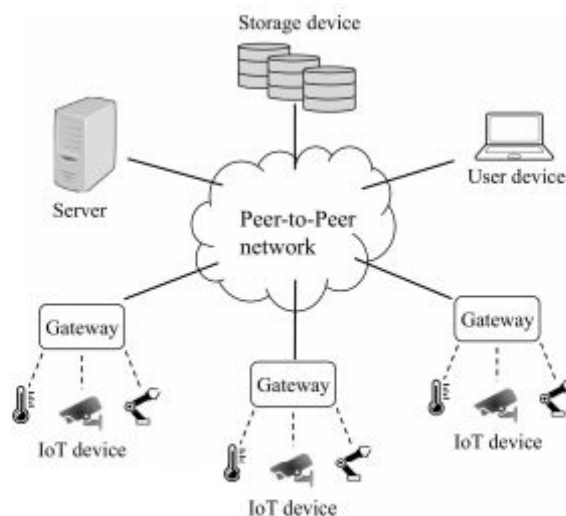


Imagen 13 - Red Peer to Peer

El grupo de dispositivos IoT el cual se quiere proteger es representado por otro dispositivo al que se le llamará gateway, este se encargará de administrar las peticiones hacia cada uno de los dispositivos IoT dentro de su alcance local.

- *ACC (Access Control Contract)*: Este contrato es el encargado de proveer la validación estática para un par Sujeto, Objeto de dispositivos dentro de la red.
- *JC (Judge Contract)*: Este contrato es el encargado de proveer la

validación dinámica, es decir, analizar comportamiento y asignar penalidades para un Sujeto que está intentando acceder a un recurso de un Objeto dentro de la red.

- **RC (Register Contract):** Este contrato permite el registro de ACC para un par Sujeto-Objeto dentro de la red, un Juez (JC) compartido para que los ACC hagan uso de este cuando realicen la validación dinámica.

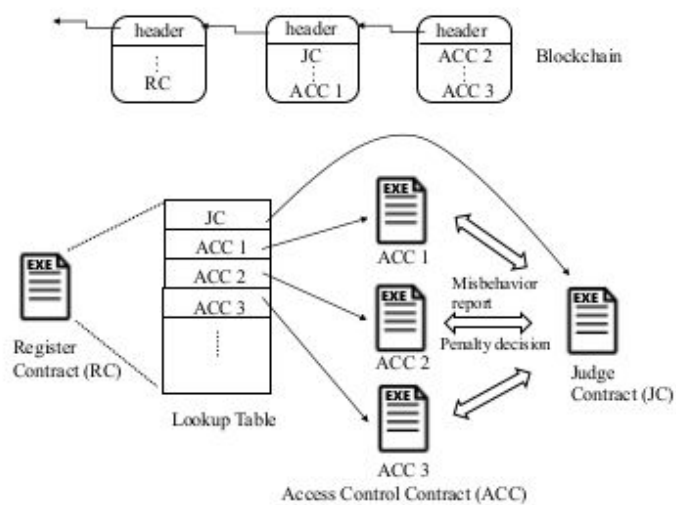


Imagen 14 - Representación visual de “Register Contract”

La *Imagen 14* ilustra a grandes rasgos el comportamiento de los contratos dentro de la Blockchain, a continuación, abordaremos en profundidad cada uno de los contratos que componen esta arquitectura.

- **El DNS de contratos dentro de la Blockchain Privada (RC):** Como ya se había mencionado anteriormente es el Register Contract; este es el encargado de contener para cada tupla (Sujeto, Objeto) permitida dentro de la red asociada, su respectivo ACC.

Para realizar esto el RC cuenta con una tabla denominada Lookup table en la que almacena la dirección del contrato ACC o JC, para que este pueda ser localizado dentro de la red, y el ABI (Application Binary Interface) , para que quien lo requiera pueda interactuar con las funciones con las que cuenta.

MethodName	Subject	Object	ScName	Creator	ScAddress	ABI
Method 1	Server A	Sensor B	ACC 1	Sensor B	0xca35b7d915458ef540ade6068dfe2f44e8fa733c	accessControl(),...
Method 2	Server A	Sensor B	ACC 2	Sensor B	0xab072c469475346532bf47aea86df61761049565	accessControl(),...
Method 3	Sensor B	Server A	ACC 3	Server A	0xb51fd86d4c998531056a501344060fbafc32a48	accessControl(),...
JC			Judge		0x3f23c7b929ceed4191ef6064ffc33902ea1d92b	misbehaviorJudge(),...
...

Tabla 1 - Lookup table

- **Quien juzga y penaliza comportamiento anómalo (JC):** El JC es el encargado de la validación dinámica de la petición para control de acceso, es decir, este contrato decide la penalidad que se debe aplicar sobre un sujeto que presenta un comportamiento anómalo de acuerdo a la fórmula con que haya sido definido.

Adicional a esto el JC almacena el registro de comportamiento malicioso dentro de la red dentro de una tabla que denomina Misbehavior Records, permitiendo así llevar un registro inmodificable de estos comportamientos anómalos dentro de la red.

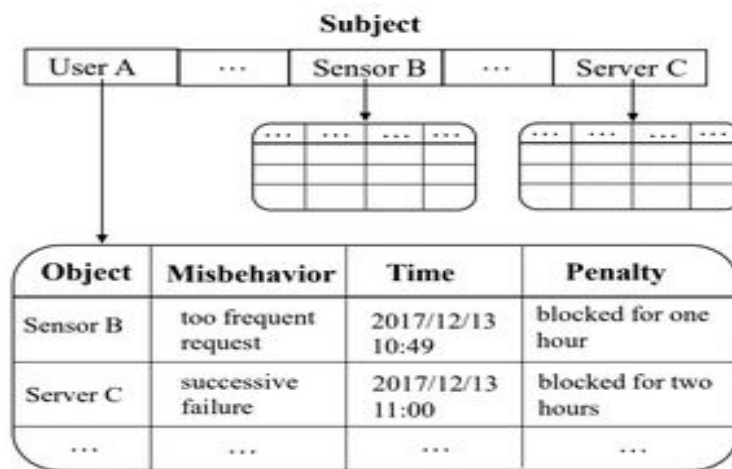


Imagen 15 - Representación de la tabla de “Registro de anomalías”

- **La puerta de entrada ACC:** El ACC define un método de control acceso que se quiere usar para un par Sujeto-Objeto, este realiza la validación estática a través de tablas con políticas de control de acceso de la forma, Recurso - Acción - Permiso

Resource	Action	Permission	ToLR
file A	read	allow	2017-12-11 16:19
file A	write	deny	2017-12-12 20:34
Program A	execute	deny	2017-12-11 16:19
...

Tabla 2 - Ejemplo de tabla con políticas de control de acceso

Nota: La columna ToLR hace referencia al registro de cuándo fue la última petición a el recurso con una acción específica.

Adicional a esto el ACC contiene un registro de las penalidades que han sido aplicadas al sujeto, esto lo hace definiendo una tabla que relaciona el Tipo de comportamiento encontrado, la fecha en la que se produjo y la penalidad que aplicó para el sujeto.

Misbehavior	Time	Penalty
Too frequent access	2017-12-11 16:19	blocked for 2 hours
Too frequent access	2017-12-12 20:34	blocked for 4 hours
...

Tabla 3 - Lista de anomalías para cada recurso

- **Proceso de solicitud de control de acceso:** De acuerdo con el funcionamiento de cada uno de los contratos se proponen dos formas de realizar el proceso de solicitud de control de acceso, el primero cuando el Sujeto obtiene el ACC consultando directamente el RC, y el segundo cuando el Sujeto consulta directamente el Objeto y es el objeto quien consulta el RC para obtener el ACC correspondiente.

Caso 1: ACC llamado por el Sujeto

En este caso el Sujeto sólo conoce la dirección del RC y su ABI para poder ejecutar sus funciones.

- Inicialmente le solicita de acuerdo a el objeto al que quiere acceder la dirección del contrato ACC correspondiente y el ABI que permite interactuar con dicho contrato.
- Una vez el sujeto conoce el ACC envía la solicitud de acceso a uno de los recursos que contiene dicho objeto (Gateway)
- El ACC válida el acceso estático por medio de la tabla de políticas definida y si encuentra algún comportamiento extraño lo reporta al JC
- El JC de acuerdo al comportamiento anómalo calcula la penalidad y la informa al ACC
- El ACC retorna la respuesta al sujeto informando sobre si el acceso fue denegado o autorizado.

Caso 2: ACC llamado por el Objeto

En este caso el Sujeto sólo conoce la dirección IP del Objeto al que quiere acceder.

- El Objeto solicita de acuerdo con el sujeto que quiere acceder la dirección del contrato ACC correspondiente y el ABI que permite interactuar con dicho contrato.
- Una vez el objeto conoce el ACC envía la solicitud de acceso realizada

por el sujeto a uno de sus recursos.

- El ACC válida el acceso estático por medio de la tabla de políticas definida y si encuentra algún comportamiento extraño lo reporta al JC
- El JC de acuerdo con el comportamiento anómalo calcula la penalidad y la informa al ACC
- El ACC retorna la respuesta al objeto informando sobre si el acceso fue denegado o autorizado.
- El Objeto informa al Sujeto sobre la respuesta obtenida.

Características más relevantes de la solución:

- Garantiza no repudio: al tener la auditoría de eventos dentro de la blockchain el acceso a un recurso es compartido para todos los demás nodos y esta información es inmodificable.
- Preserva la integridad de los datos, al contar con una arquitectura descentralizada y con un método de consenso.
- Gestión de autorización, ya que permite el acceso a recursos sólo a sujetos autorizados.

Posibles ataques a la solución propuesta:

Existe la posibilidad de un ataque por suplantación de un objeto: Al obtener la dirección privada y pública de alguna wallet de la red privada de Ethereum, es posible suplantar el Objeto y de esta forma desplegar un contrato ACC que permita el acceso a cualquiera de sus recursos.

10. PROPUESTA DE SOLUCIÓN

Con base en las investigaciones realizadas, sobre como Blockchain puede llegar a solucionar de una mejor forma el reto de ciberseguridad en los dispositivos IoT, se analizaron las arquitecturas propuestas en [1] y [4], luego se realizó la propuesta de una arquitectura, que tuviera en cuenta la comunicación de eventos de seguridad entre cada uno de los nodos dentro de la Blockchain, y un enfoque mercantil que permitiera visualizar cual es la ventaja de contar con un sistema para la gestión de eventos de seguridad, basado en Blockchain en un ámbito empresarial.

Arquitectura de la solución

La Arquitectura de la solución está compuesta por diferentes elementos; tales como: Inteligencia Interna y Externa, SIEM Distribuido, Dispositivos Centinelas y dispositivos IoT, cada una de las interacciones entre estos elementos permite crear un sistema de seguridad robusto, aprovechando cada una de las características que la blockchain ofrece.

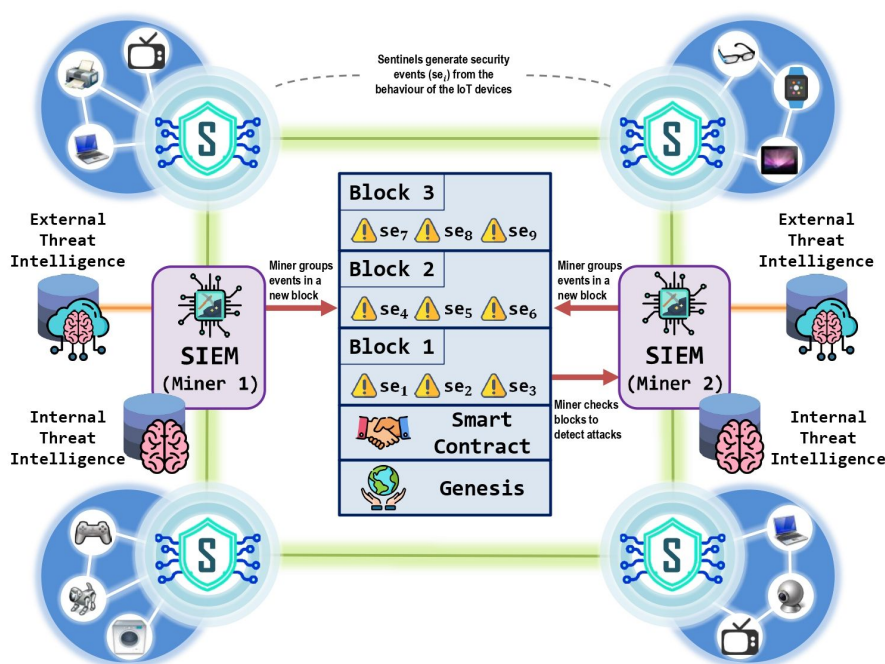


Imagen 16 - Arquitectura de la propuesta de solución del proyecto de grado

Dispositivos IoT

Los dispositivos IoT pertenecen a diferentes escenarios tales como, smart home, smart office entre otros, envían constantemente datos que recolectan a través de la red para ofrecer servicios en tiempo real.

Dispositivos Centinela

El Centinela es un dispositivo IoT que escucha constantemente el medio en una red y recolecta eventos de seguridad, estos eventos de seguridad son concatenados y posteriormente almacenados en la blockchain.

SIEM Distribuido

Actúa como minero dentro de la red debido a su capacidad, adicionalmente recolecta los eventos de seguridad de cada transacción y los usa como datos de entrada para que estos sean analizados y catalogados.

Inteligencia Interna o Externa

Este componente que permite identificar ataques por medio del uso de IoC, indicadores de compromiso y IoA indicadores de Ataque, que usan como datos de entrada los eventos de seguridad recolectados y almacenados en la Blockchain.

La integración de cada uno de estos elementos ofrece ventajas sobre otras arquitecturas que se pueden visualizar en los siguientes casos:

Añadir eventos de seguridad a la Blockchain

El añadir este tipo de eventos a la red Blockchain garantiza la integridad, ya que una vez el bloque no se puede modificar una vez es creado, el no repudio, ya que cada transacción tiene información del emisor y la distribución de la información añadida a cada uno de los nodos dentro de la red.

Consumir la Blockchain para detectar ataques distribuidos

Una vez los eventos de seguridad son añadidos a la blockchain, los nodos mineros usan esta información para que sea analizada por un SIEM, una vez se generan las alertas de seguridad, estas son agregadas de nuevo a la blockchain, gracias a reglas de correlación e indicadores de ataque, los demás mineros son capaces de identificar una amenaza distribuida en el sistema.

Detección de ataques bajo escenarios hostiles

La Blockchain permite que en caso de que uno de los nodos mineros sea atacado y eliminado de la red, los demás nodos mineros van a continuar cumpliendo con sus funciones de minar bloques y analizar los eventos de seguridad recibidos, evitando así un único punto de fallo.

Auditar un incidente de seguridad

El anidamiento de bloques dentro de la Blockchain, permite en caso de que se presente un incidente de seguridad, poder realizar auditoría sobre los bloques y encontrar cual de los centinelas fue quien reportó el evento de seguridad relacionado.

Escalar la Infraestructura de IoT segura

Añadir un nodo confiable dentro de la red, requiere poco esfuerzo en instalación y configuración del nuevo nodo, añadir más nodos garantiza una red más confiable y segura debido a el mecanismo del método de consenso.

11. LOGROS DEL PROYECTO

- Documentación de casos de uso realizados al Control de Acceso a dispositivos IoT y a la compartición de Indicadores de Compromiso entre Centinelas.
- Desarrollo de una propuesta SIEM distribuida para escenarios de IoT que aprovecha los beneficios de una Blockchain (e.g. operaciones sin servidor, integridad, no repudio y resistencia).
- Desarrollo de contratos inteligentes para manejar bloques de eventos de seguridad y detectar ataques de los eventos de seguridad disponibles en blockchain.
- Integración de Inteligencia de amenazas externa e interna del BSIEM IoT para realizar validaciones locales originadas en contratos inteligentes.
- Evaluación y análisis de los resultados de la propuesta y sus características a través de experimentos exhaustivos, que a su vez demostraron la viabilidad de la solución para las organizaciones.

12. CONCLUSIONES Y TRABAJOS FUTUROS

- BSIEM-IoT ofrece características deseables para un sistema de seguridad robusto, como resistencia, confianza, auditabilidad y escalabilidad.
- Los experimentos muestran que BSIEM-IoT es capaz de obtener un rendimiento deseable con tiempos de transacción bajos, que depende de la configuración del umbral de eventos de seguridad (λ_{se}) y el método de consenso.
- Planeamos permitir nuevos tipos de transacciones en nuestra solución de acuerdo con el tipo de evento de seguridad detectado por el centinela IoT, p. Se podrían agregar eventos de seguridad más críticos a la cadena de bloques con una prioridad más alta, mientras que se podría agrupar la prioridad media o baja.
- Estudiaremos la viabilidad de construir una nueva generación de dispositivos IoT que sean capaces de ser nodos de la blockchain, calificados para informar eventos de seguridad interna a blockchain.

13. BIBLIOGRAFÍA

[1] Zhang, Y., Kasahara, S., Shen, Y., Jiang, X., Wan, J.: Smart contract-based access control for the internet of things. arXiv preprint arXiv:1802.04410 (2018)

[2]

https://xh4y28w4m30fiwf22ex7gvfa-wpengine.netdna-ssl.com/wp-content/uploads/2017/11/IoT_Barometro2017-18_ESP-Resumen-Ejecutivo.pdf

[3]

<https://www.iotworldonline.es/las-grandes-estadisticas-del-internet-de-las-cosas-iot/>

[4] Dorri, A., Kanhere, S., Jurdak, R., Gauravaram, P.: Blockchain for IoT Security and Privacy: The Case Study of a Smart Home. In: 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerComWorkshops) (2017) - Pendiente por agregar

[5] Collen, A and Nijdam, N A and Augusto-Gonzalez, J and Katsikas, S K and Giannoutakis, K M and Spathoulas, G and Gelenbe, E and Votis, K and Tzovaras, D and Ghavami, N and Volkamer, M and Haller, P and Sánchez, A and Dimas, M, P.: GHOST - Safe-Guarding Home IoT Environments with Personalised Real-Time Risk Control. In: 2018 Security in Computer and Information Sciences

[6] R. I. Bonilla and C. L. Abad, P.: Towards a Real Time Framework for Monitoring IoT Devices for Attack Detection: Vision Paper In: 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology