Documentación Estado del Arte

El internet de las cosas es una tecnología floreciente y muy útil en diversos campos. Se utiliza a través de dispositivos IoT, que al tener MCUs de limitada capacidad no pueden implementar grandes medidas de seguridad, lo que causa preocupación.

Debido a que los dispositivos necesitan de firmware, este se convierte en un punto de ataque. Para resolverlo, se requiere de actualizaciones. En el RFC 9019 presentado por el IETF se presenta una arquitectura basada en un servidor centralizado para la actualización de dispositivos IoT. Un servidor centralizado sin embargo presenta los siguientes problemas:

* El servidor único presenta un único punto de seguridad, lo que lo convierte en un objetivo de alto valor fácil de atacar.
* El ataque a un único servidor puede denegar completamente el servicio.
* En caso de que el autor (o encargado de mantenimiento del servidor) desaparezca por cualquier motivo, las actualizaciones dejan de estar disponibles.
* El rápido incremento de dispositivos puede sobrecargar el servidor, potencialmente denegando servicio o causando problemas derivados a la falta de capacidad.

Debido a los riesgos de seguridad presentes en arquitecturas centralizadas, se propone la idea de adaptar la arquitectura presentada a un modelo descentralizado utilizando la tecnología blockchain. Estos son algunos de los artículos propuestos previamente que intentan solucionar dichos problemas:

**B.Lee, J.-H. Lee Lee, ‘‘Blockchain-based secure firmware update for embedded devices in an Internet of Things environment,’’:** El primer paper que propone una solución descentralizada. Propone el uso de una blockchain cuyos nodos estaban presentes en los dispositivos IoT, siendo de verificación, nodos de petición y nodos de respuesta. Cada nodo de verificación es conocedor de la última versión y es verificado por un productor. Un nodo de petición manda su versión a un nodo de verificación, y si este último determina que su versión actual es anterior a la ofrecida, el nodo de petición solicita a otro nodo su versión, hasta que todos los nodos tengan la última versión.

Dado que los dispositivos tienen capacidades limitadas que ya dificultaban la seguridad convencional, este modelo presenta el problema de añadir demasiada carga de tarea a los dispositivos.

**A. Boudguiga et al.:** Proponen un modelo donde el productor selecciona los dispositivos a actualizar y sube el software al portal web del productor, y los dispositivos reciben una notificación.

A partir de aquí, los binarios se suben a una transacción en una blockchain. Los bloques se hacen largos, pero los binarios se conservan mientras exista la red. Esa longitud, sin embargo, puede presentar problemas a largo plazo.

Otra opción que se plantea es que los binarios lleguen a nodos inocuos.

**A. Yohan y N.Lo:** Proponen un modelo donde se crean un repositorio vendor y uno broker, que almacenan los binarios y la información asociada. La red se compone de nodos vendor, completos y ligeros. Cuando un nodo vendor recibe una actualización de un repositorio, despliega un contrato requiriendo que los nodos completos correspondientes verifiquen el firmware. Cuando se completa, los nodos de los gateways descargan el firmware y lo distribuyen a los dispositivos.

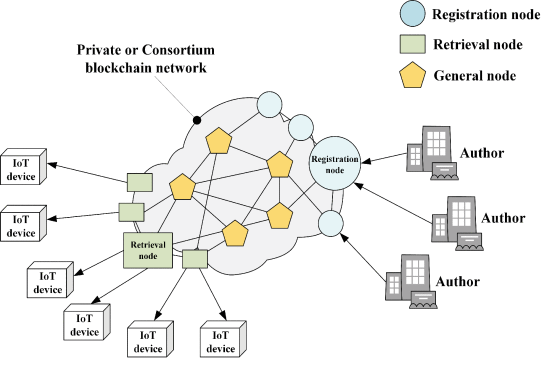
Presenta el problema de tráfico excesivo, duplicados en broker y vendor, y fallos cuando los gateways se ven comprometidos (este está muy mal explicado).

**S. Choi and J. -H. Lee, "Blockchain-Based Distributed Firmware Update Architecture for IoT Devices”:** Se propone una arquitectura que contiene una red blockchain capaz de distribuir archivos con un comportamiento P2P sin un servidor de firmware. El manifiesto se comparte en la red y tiene el hash único de una imagen. Debido a que los dispositivos tienen capacidades limitadas, el nodo de la red no estará en el dispositivo, sino que será externa.

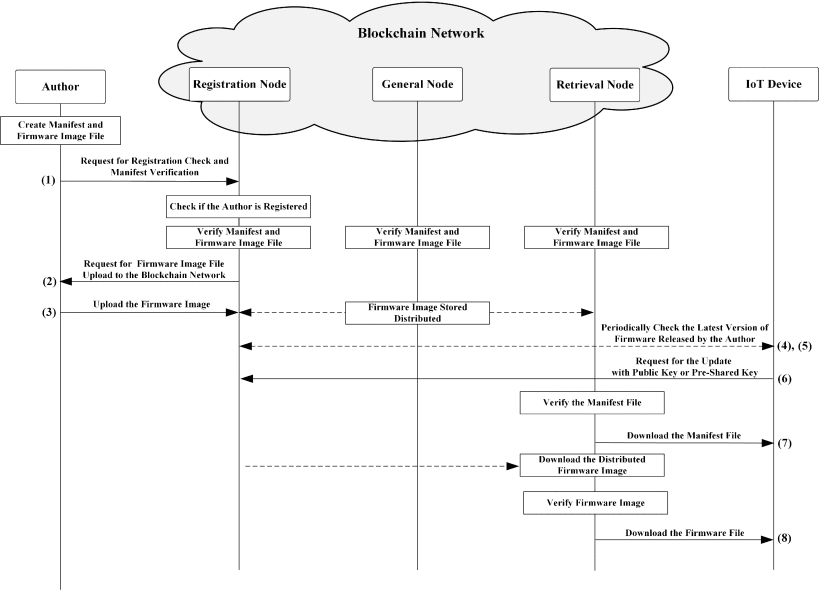
Para evitar problemas de tráfico y cuellos de botella, se definen nodos de registro y nodos de recuperación.

Entidades participantes:

* Autor: fabricante que produce los dispositivos, o distribuidor del firmware y los manifiestos. Suben las imágenes y manifiestos a la red.
* Dispositivos IoT: cualquier producto operando con MCUs, sensores o actuadores. Son los dispositivos que requieren de la imagen para su actualización. Deben contar con una clave pública o pre compartida.
* Nodos de registro: Nodos de la blockchain que gestionan el registro de los autores. Un autor debe ser registrado a través de uno de estos nodos para subir las imágenes y manifiestos.
* Nodos de recuperación: Nodos que gestionan la descarga de los manifiestos para los dispositivos, y de las imágenes a través de las URIs presentes en los manifiestos.
* Nodos generales: Nodos que constituyen la red, a parte de los nodos de registro y recuperación.
* Nodos blockchain: Todos los nodos que componen la red. Se separan en nodos de autor y nodos de no-autor. Los nodos de autor son gestionados por el autor, mientras que los de no-autor son gestionados por voluntarios o seleccionados y certificados por el autor. La distribución del sistemas de ficheros para las imágenes son configuradas con los nodos.



Arquitectura propuesta, propiedad de los autores originales.



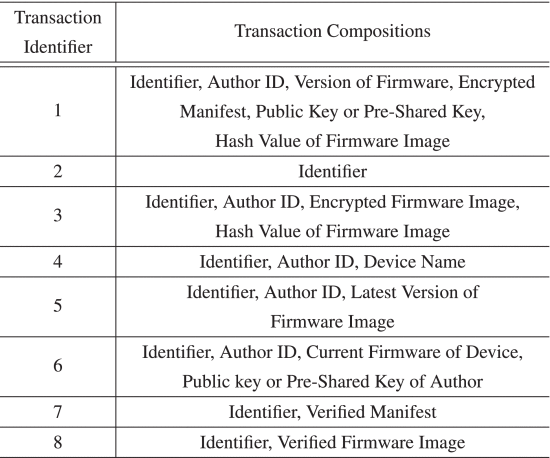
El procedimiento es el siguiente:

1. Registro del autor: el manifiesto y una URI para la descarga de la imagen son proporcionados. Se solicitan al nodo de registro que los suba a la red mediante una transacción.
2. Chequeo del registro: El nodo de registro comprueba la ID y clave pública presentes en la transacción. Si es la primera transacción de un autor, se registra su ID. Se valida el manifiesto con la clave pública.
3. Almacenamiento del manifiesto: Se verifica y almacena el manifiesto, que no podrá ser alterado ni eliminado de la red.
4. Almacenamiento de la imagen en sistema de ficheros distribuido: Las imágenes registradas y verificadas se almacenan en un sistema P2P como IPFS, y los haces se guardan en la blockchain.
5. Dispositivos comprueban la versión de firmware: los dispositivos chequean periódicamente la versión en algún nodo de la red. Si existe una mas reciente, solicitan la descarga a un nodo de recuperación.
6. Descarga de manifiesto: El nodo de recuperación al recibir la petición descarga el manifiesto y lo envía al dispositivo. Nodo y dispositivo deben contar con una relación de confianza.
7. Descarga de firmware: el nodo de recuperación descarga la imagen de los nodos vecinos con la información del manifiesto. Los nodos blockchain comprueban la imagen descargada con la clave pública y se envía al nodo de recuperación, quien debe enviarla de manera segura a los dispositivos.

Composición de transacciones:

1. Autor → Nodo de registro: petición. Se solicita registro y verificación de manifiesto
2. Nodo de registro → autor : petición. Se solicita la imagen
3. Autor → nodo blockchain: respuesta. Se sube la imagen al sistema distribuido.
4. Dispositivo → Nodo blockchain: petición. Se comprueba la última versión.
5. Nodo blockchain → dispositivo: respuesta. Se envía la última versión (info).
6. Dispositivo → Nodo Blockchain: petición. Se solicita la actualización.
7. Nodo recuperación → Dispositivo: respuesta. Se envía el manifiesto.
8. Nodo recuperación → Dispositivo: respuesta. Se envía la imagen.

Se contiene el tipo como identificador en la transacción.



*Mitigación / Seguridad*

Targeting issue: El modelo cliente servidor presenta el problema de que un único servidor gestiona los archivos que necesitan todos los dispositivos, por lo que basta un punto de fallo para atacar a todos los dispositivos o negarles las actualizaciones con denegación de servicios o eliminando los ficheros. También se pueden crear sobrecargas al aumentar los dispositivos conectados.

Desaparición del autor: Si el autor sale del negocio antes de que se descarguen las actualizaciones, por el motivo que sea, las actualizaciones dejan de estar disponibles. Lo mismo pasa si son víctimas de un ataque. Se crea un entorno donde las compañías con menos recursos son más vulnerables.

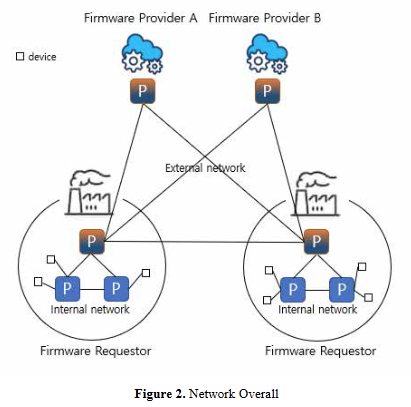
La blockchain elimina el problema del punto único de fallo añadiendo nodos, y al contar con irreversibilidad, aunque los autores desaparezcan, las actualizaciones quedan grabadas y disponibles en la red. Si un nodo se ve comprometido, el resto de la red puede seguir operando gracias a protocolos de consenso que impiden la actualización de la red con transacciones maliciosas.

Para que funcione, sin embargo, es necesario utilizar una red de consorcio o una red privada.

Esta arquitectura no utiliza contratos inteligentes, para generar menos carga.

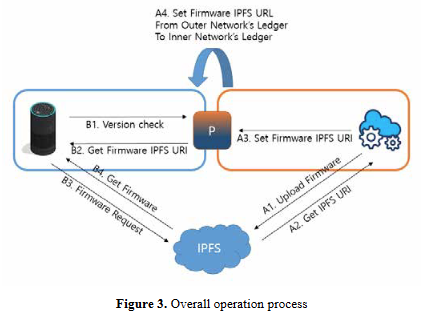
**Minsung Son, Heeyoul Kim “Blockchain-based secure firmware management system in IoT environment”:** Proponen un modelo inicialmente similar al anterior. Sin embargo en este caso se resaltan los ataques de man-in-the-middle y rollback como problemas del modelo cliente-servidor tradicional.

Se propone una red que conecte nodo de proveedores con nodos de consumidores, a los que llamaremos brokers. Estos brokers a su vez están conectados con nodos en otra red, interna a la entidad que utiliza los dispositivos IoT. Dichos nodos internos están conectados con los dispositivos a actualizar.



Existe la figura del manager, quien se encarga del proceso de registro de los dispositivos a la red. Se añade al ledger el estado del dispositivo junto a su número de serie, y se le provee de un par de claves pública y privada.

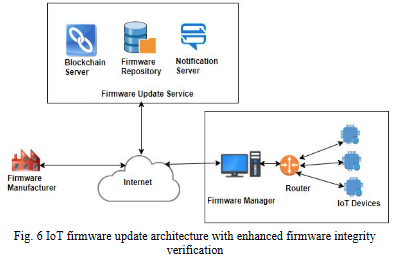
El proceso de actualización es el siguiente: Una imagen se sube a la red IPFS. Después la URI y la información correspondiente a la actualización(entiendo que esto sería a través de un manifiesto) se registran en el ledger utilizando chaincode (los contratos inteligentes de HyperLedger Fabric). A través de la red, los brokers obtienen la información, y cuando reconocen que es necesario, descargan la actualización y la registran en la red interna. Cuando los nodos internos lo consideran necesario, pueden enviar la actualización a los dispositivos cuando estos la solicitan.



**Alexander Yohan, Nai-Wei Lo, Liem Peter Santos “Secure and Lightweight Firmware Update Framework for IoT Environment”:** Nuevamente se expresan las mismas preocupaciones, pero esta vez se propone utilizar skipchain, en pos de darle un enfoque de ligereza al framework. (Debido a que usan una expresión un poco obtusa y ningun gráfico explicativo, lo dejo para el final si es necesario).

**Samip Dhakal Fehmi Jaafar Pavol Zavarsky “Private Blockchain Network for IoT Device Firmware Integrity Verification and Update“:** Proponen utilizar actualizaciones delta además de blockchain, aunque desde el principio avisa de que no siempre es posible. Utilizando este tipo de actualizaciones, los dispositivos solo tendrían que descargar aquellas partes del código que cambian, en lugar del programa completo. Presenta problemas en que se necesita una versión soportada por Delta Patcher ya instalada en el dispositivo, y en que cuando no se puede ejecutar la actualización, se consumen recursos considerables de todas maneras.

La solución que proponen es la utilización de una estructura de blockchain similar a las de los otros papers.

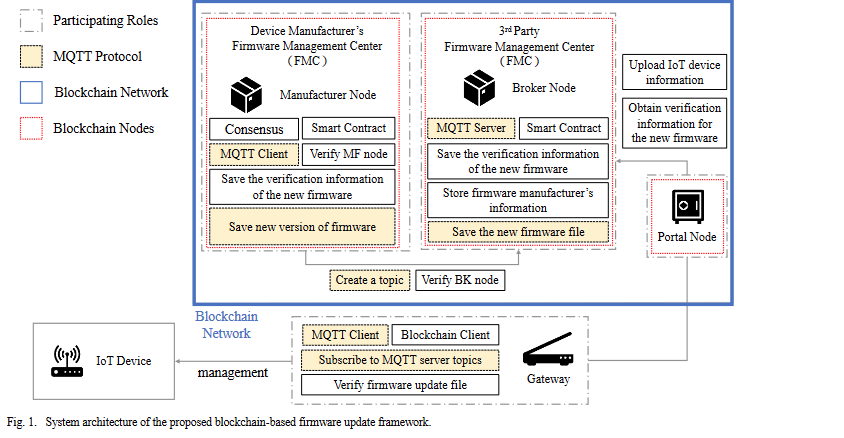


El fabricante genera la imagen, y un fichero delta que contiene las diferencias generadas en la actualización. Lo entrega al servicio de actualización, que contiene una blockchain donde se almacena el registro de todos los metadatos de firmware disponibles y sus archivos delta; y que contiene también un almacenamiento de las imágenes y un servidor de notificación que avisa a los nodos de gestión de firmware cuando hay una actualización disponible.

Cuando recibe esta notificación, el nodo de gestión de firmware, que se encuentra en el gateway de la red de dispositivos, confirma si la actualización les corresponde a los dispositivos de su red y comprueba los hashes de las versiones. Si la versión anterior de firmware no ha sido modificada y la actualización delta es aplicable, descarga e instala los archivos necesarios. Si no es aplicable, se descarga la imagen más reciente completa.

**Meng-Hsuan Tsai, Yu-Cheng Hsu, Nai-Wei Lo “An Efficient Blockchain-based Firmware Update Framework for IoT Environment” :** En este caso, se enfatiza el ataque DDoS como riesgo de seguridad. Se propone un modelo que utiliza blockchain y el protocolo MQTT.

Se definen tres roles: El fabricante, que crea los dispositivos y el firmware; el broker, distribuidor de los dispositivos y encargado de subir el firmware a los servidores MQTT y crear nodos de portal; y los gateways, que se conectan a los nodos portal y a los dispositivos.



Se definen tres funciones dentro de la red:

* Suscribirse: es el proceso mediante el que el gateway otorga a los brokers la información sobre los dispositivos conectados. De la misma manera, el gateway también obtiene información de los brokers.
* Publicar: el proceso con el que se obtiene el firmware de los brokers. El fabricante otorga imágenes a los brokers, quienes piden información a los gateways sobre los dispositivos y les envían el firmware.
* Validar: El proceso mediante el que el gateways verifica integridad y origen del firmware. Se realiza una operación hash sobre el firmware y se compara al log de la red blockchain

<https://www.mdpi.com/1424-8220/22/2/530>

Otro paper que he encontrado y que puedo leer más tarde