Secure Data Management in Internet-of-Things Based on Blockchain

Jenil Thakker
Computer Engineering
San Jose State University
San Jose, California, USA

Ikwhan Chang
Computer Engineering
San Jose State University
San Jose, California, USA

Younghhee Park
Computer Engineering
San Jose State University
San Jose, California, USA

Abstract — As Internet-of-Things (IoT) infrastructure has significantly dominated our life such as smart homes, hospitals, vehicles, and farms, efficient and secure IoT data management have been a new challenge to handle massive volumes of data and a myriad of IoT devices. To solve these problems, blockchain has become a critical technology to create a decentralized and publicly accessible network for immutable and verifiable data transaction for IoT. This paper proposes an efficient and secure data management system by using the blockchain technology to issue certificates for IoT devices and to retrieve data by using the certificates in any locations. The certified IoT devices in blockchain provide user privacy and data integrity for users while giving efficient data storage and retrieval through our blockchain-based IoT certificate management system. The certificates issued to the IoT devices hold data generated by the device and store it on the blockchain. We evaluate our proposed system on Ethereum to measure transaction cost and speed by issuing and verifying certificates according to the amount of data stored.

I. INTRODUCTION

Embedded computing systems are increasingly adopted in different technological ecosystems. Internet-of-Things (IoT) has a large number of data-gathering and monitoring devices with limited storage sizes, resources, and computing power. IoT devices are typically equipped with small memory and CPU like a few MHz CPU and a few KB RAM/ROM, while they generate a millions of data daily [1]. Given these limitations and the rapid growth of the amount of data generated by the IoT devices, there needs to be a mechanism for efficiently managing and accessing data for worthwhile usage. In addition to efficiently storing data, IoT devices require reliable security mechanisms and a trusted environment. For instance, data collected from wearable health monitoring devices can be valuable clues for determining an appropriate medical treatment for patients [2]. However, people can not utilize data from unauthorized or compromised devices.

Recent implementations for efficient data storage rely on outsourcing data to centralized servers and employ trusted third party services. Although this mechanism drives the current IoT infrastructure, it is not necessarily the most suitable solution for the IoT, as it neglects the locality of data and mandates centralization through third party entities [1]. For example, over 300 different EHR(Electronic Health Record) systems are in use today, but most of them adopt a centralized architecture which suffers from single point of failure [2]. To address scalability and reliability, many research has worked on blockchain to be used for IoT [3]–[6]. In particular, Ethereum is prominently used in state-of-the-art blockchain based solutions for IoT devices for its fast, inexpensive and reliable network [4], [5]. The smart contracts on Ethereum are decentralized applications with a state stored on the blockchain network [7].

To address the storage limitations and to increase reliability in IoT, this paper proposes a blockchain based solution to efficiently store and access data through certificate management. Our proposed approach is an Ethereum based decentralized application that utilizes distributed storage management to achieve scalability and reliability. We design a smart contract on Ethereum to create a user-based decentralized application in the trusted and reliable network, privacy-preserving platform. The Ethereum smart contract is a key component for issuing and accessing certificates associated to the IoT devices. When a user invokes the smart contract to issue or access the certificate data, the state of the smart contracts is stored on the blockchain as a transaction. These transactions are stored on a public ledger giving a user the ability to verify the transaction. By using the blockchain technology, our system achieve a decentralized data-management model through certificate management by eliminating the need for a central authority. Furthermore, decentralized IoT data management will enable the users to retrieve the data stored on the blockchain without having physical access to the IoT device.

Blockchain technologies are able to verify transactions and store information from a large amount of devices, enabling the creation of applications that require no centralized cloud [8]. However, to scale the data generated by the IoT devices we use a decentralized cloud service called Inter-Planetary File System (IPFS). To allow devices to store large chunks of data, we compress the data and use Ethereum as a secure intermediary, until we decompress and transfer the data to the decentralized cloud through a data manager (Web3). Once the data has been transferred to the cloud, it can be retrieved by the users with the IoT certificates owned by the users.

Contribution. In this paper we make the following contributions to address the issues of storage limitations and reliability, and lack of efficient data-management in IoT:

- Efficiently manage data storage and retrieval through certificate management based on Blockchain.
- Adopt decentralized and distributed storage services
through Inter-Planetary File System (IPFS).

- Simulate a lightweight Ethereum client to test transaction costs, transaction confirmation times for issuing and verifying certificates.

**Organization.** The remaining sections of this paper are organized as follows: Section II entails a brief description of recent implementations using blockchain in the IoT domain. Section III explains in detail about the proposed data-management architecture and the system-flow design. Section IV evaluates our proposed solution by simulating a lightweight Ethereum client, an IPFS instance and a data manager to analyze transaction costs and times for certificate management. Section V discusses current challenges and potential improvements in the proposed model. Lastly, Section VI concludes our paper with a review on utilizing blockchain for the IoT infrastructure and our data-management model.

II. RELATED WORK

Blockchain has numerous applications in the IoT domain, ranging from supply-chain management, data-integrity, access-control, distributed storage, secure communication protocols and many more. In terms of data-integrity, researchers have proposed solutions for reliably verifying data for both, data owners and data consumers [3]. This technique aims at eradicating the need for a third party auditor, and promoting transparency. The exchange of IoT data between two entities can further be secured through communication protocols. Researchers at ETH Zurich, propose a protocol for sharing data in streams and associating these streams with access permissions [1]. In addition, secure communication protocols can also be established for a smart-city infrastructure [4] using Ethereum clients.

Considering supply-chain management solutions, blockchain is used in the agriculture and food (Agri-Food) ecosystems to track and store orders, providing transparency, traceability and auditability [5]. Digital supply chain (DSC) integration can also be achieved for business partners and value-added service providers [9], where in addition to products and services, data can be efficiently tracked and shared with organizations. Establishing a trusted architecture in a supply-chain model is significant for maintaining a tamper-proof log of events. TrustChain [6] developed a reputation based model for food supply-chains using a permissioned blockchain.

Blockchain based solutions for IoT devices are also used for developing intuitive machine learning algorithms. These machine-learning models are trained for analyzing customer feedback and accurately predicting customer requirements for IoT devices [10]. To prevent faults, and errors in the manufacturing process of the IoT devices, credit-based consensus solutions have been proposed [11] to guarantee transaction efficiency and quality control.

These state-of-the-art approaches towards adopting blockchain based solutions for the IoT domain prominently use lightweight blockchain clients, given the power limitations. We use Ethereum for the proposed model due to it’s trusted network and efficient file management. Ethereum is a decentralized public platform that holds a record of transactions generated from blockchain based applications. These applications can be developed using smart contracts, that run on Ethereum. One goal of Ethereum is to allow for the development of arbitrary applications and scripts that execute in blockchain transactions, using the blockchain to synchronize their state globally in a manner that can be verified by any participant in the system [12].

III. OUR APPROACH

A. Overview

Our proposed model focuses on retrieving and managing data without having physical access to the IoT device. This model can be further branched into three layers of technology: physical layer, blockchain network layer, and a decentralized cloud layer. The physical layer represents a set of IoT devices produced by a manufacturer, and owned by an organization or a user. These devices are connected to the blockchain network layer through an Ethereum smart contract. The smart contract is responsible for deploying and storing the IoT device’s data on Ethereum. In order to scale large chunks of data, the data stored on Ethereum is compressed. The data is then decompressed and stored on a decentralized cloud platform through a data manager (web3 JavaScript API). The data can be retrieved by the user from the decentralized cloud by invoking the smart contract.

B. System Architecture

1) **IoT Infrastructure Plane:**

- **Physical Layer:** This layer represents a set of IoT devices denoted as $D_i$ where ‘i’ ranges from $0 \to n$. The IoT devices are issued certificates by a smart contract through a transaction on Ethereum. Each device ($D_i$) is issued a certificate $Cert_i$ (where ‘i’ ranges from $0 \to n$) that holds the following metadata:

$$Cert_i = [Owner_i, id_i, firmware_i, l_i, t_i, data_i] \quad (1)$$

In the equation above, $Owner_i$ represents the Ethereum account address of the user owning the device. The $id_i$ represents a unique identification (RFID or Device ID) provided by the manufacturer. The registered location, issue date and firmware version are denoted as $l_i$, $t_i$ and $firmware_i$ respectively. Lastly, the $data_i$ field contains the data generated by the user (stored as a string), which can be updated by invoking a function, ‘issueCert’, from the smart contract, that pushes the updated data into an array. This data field can be retrieved by the user, by invoking a function, ‘verifyCert’ which requests the cloud platform to return the data.

2) **Blockchain Network Layer:**

- **Blockchain Network Layer:** The blockchain network layer is further divided into two components: data manager (Web3) and a certificate contract.
1) **The Certificate Contract:** This contract is responsible for certificate management and storing the IoT device’s data on Ethereum. There are two main functions defined in this contract: ‘issueCert’ and ‘verifyCert’. The ‘issueCert’ function is invoked by the manufacturer, where it creates and issues a certificate (Cert$_i$) for a device (D$_i$) with its associated metadata (as shown in equation 1). This function can later be invoked by the user or the owner (Owner$_i$) to update and store the device’s generated data and sign the data. Ethereum employs elliptic key cryptography to sign the data ensuring true ownership and security.

The ‘verifyCert’ function allows the user (Owner$_i$) to retrieve the stored data generated by the IoT device without having access to the physical device. Before the function returns the requested data, it checks for the owner’s correct Ethereum address and their signature as a security mechanism.

Invoking ‘issueCert’ and ‘verifyCert’ transactions on Ethereum cost execution fee, or otherwise known as gas. The gas limit of an Ethereum transaction is about 3,000,000 gas which allows up to 780 kB of data per transaction [4]. This execution fee is adopted from [13] and calculated based on the following equation:

\[
issue_x = \sum_{x=0}^{n} g_x \cdot SSTORE_x + g_0 \quad (2)
\]

In equation 2, issue$_x$ represents the cost for executing the ‘issueCert’ function for transaction $x$ (where $x$ ranges from $0 \rightarrow n$). The execution fee is defined as the sum of the transaction gas cost for executing the transaction ($g_x$), the storage cost ($SSTORE_x$) and the initial contract deployment cost ($g_0$).

\[
verify_y = \sum_{x=0}^{n} g_x \quad (3)
\]

Equation 3 represents a read transaction that allows users to view the requested data. The execution fee for verifying and viewing data ($verify_y$) is defined as the sum of the gas cost for executing the transaction $x$ ($g_x$). Assuming that the certificate contract has been previously deployed, the deployment cost ($g_0$) is not considered.

2) **Data Manager:** A Web3 JavaScript API is used to interact with the Ethereum node. Web3 is a set of modules that are used in this case to facilitate the connection between the decentralized cloud platform and the certificate contract. Every time a user requests data as an Ethereum transaction, web3 interacts with the Ethereum node to decompress the data and store it on the decentralized cloud server. The data is then retrieved by the user from the cloud.

- **Decentralized Cloud Layer:** The cloud layer focuses on storing large amounts of data with efficiency and providing easier access to the certificate metadata (Cert$_i$) when requested by the user. This can be achieved by utilizing an Inter-Planetary File System (IPFS) that interacts with the web3 JavaScript API to decompress the data and store it on the decentralized cloud. IPFS provides a new platform for writing and deploying decentralized applications, and a new system for distributing and versioning large chunks of data [14]. IPFS integrates both the use of complex Merkle-Linked structure with the data-addressability of P2P file sharing systems [15]. This form of distributed storage management is particularly useful when the total amount of generated data by all the devices (D$_i$) increases over time.

**C. System Flow**

Figure 2 illustrates a system flow design of the data management model. Initially, the manufacturer registers the IoT device and its associated metadata (Cert$_i$) through a smart contract invocation (‘issueCert’). Once the initial metadata has been stored on the blockchain, a certificate is successfully issued to the IoT device. Initially when the certificate is issued to the IoT device, the data field (data$_i$) is empty since no data is generated yet.

Once a user gains ownership to the IoT device, the generated data over time can be updated and stored on Ethereum through the ‘issueCert’ function on the certificate contract. By invoking the ‘issueCert’ function again, the data generated by the device is stored on the IPFS node through a Web3 data manager. Lastly, to retrieve the data, a request needs to be made to the certificate contract by invoking the ‘verifyCert’ function. This function ensures that a valid ethereum account address and a legitimate signature has been provided by the user to confirm the ownership of the IoT device. If the aforementioned credentials are valid, the requested data is returned by the IPFS node.
IV. EVALUATION

A. Setup

To simulate the proposed model, we use an Ethereum test network called Rinkeby [16]. The Rinkeby network enables us to test multiple transactions with "fake" ether, giving us fair estimates on transaction and storage costs. The certificate smart contract is written using solidity language (version 0.4.22), compiled and tested on the Remix interface [17], created by the Ethereum Foundation. Our implementation can be found at [18]. All the transactions interact directly with MetaMask [19], a service enabling blockchain-based transactions and publishing them on the public Rinkeby ledger.

The web3 JavaScript component interacts with the certificate contract by fetching data stored on the blockchain and storing it on decentralized cloud. To adopt a decentralized cloud service, we use an IPFS node. IPFS enables peer-to-peer distributed file access and storage. These components are setup on a MacBook Air running on a macOS High Sierra operating system (version 10.13.6), with processing power of 1.6 Ghz Intel Core i5 and memory of 4 GB.

To evaluate the data-management model, we test for the transaction cost for each of the functions on the certificate smart contract (‘issueCert’ and ‘verifyCert’) and it’s relationship with the amount of data stored and the number of IoT devices operating on the network, as seen in Figure 3. We also test and analyze the amount of time taken for each transaction to be successfully published on the Rinkeby public ledger.

B. Experimental Results

Figure 3 analyzes the transaction costs for issuing and verifying the certificates on Ethereum. The process of verifying the certificate entails verifying ownership of the user and returning the requested data through the IPFS node. Based on the graph (Figure 3) the cost, measured in gas (wei), for issuing certificates increases radically with the increase in the number of devices registered. The maximum amount of data stored (compressed format) on the blockchain is 500 kilo bytes, where the maximum cost to issue a certificate is about 1200 kilo wei, and to verify is about 100 kilo wei. This suggests that retrieving data and verifying ownership is relatively a cheaper operation than issuing certificates.

Figure 4 on the other hand analyzes the transaction confirmation times taken for issuing and verifying certificate metadata. Similar to the results in Figure 3, the time taken for issuing certificate is more than the time taken to verify certificates and returning user data. Considering 2500 IoT devices, the maximum transaction confirmation times for issuing certificates is about 15 seconds, whereas for verifying it, is about 11 seconds. The transaction finality for each block on average is 15 seconds on the Rinkeby testnet [20].

The results from Figure 3 and 4 suggest that the transaction cost for issuing a certificate and storing data on Ethereum is directly proportional to the amount of data stored by the user. On the other hand, retrieving data is a relatively low-cost execution compared to issuing certificates to the IoT devices. Comparing our proposed model and experimental results to recent startups and implementations in the storage space (Filecoin [21], Tendermint [22], and Storj [23]), our system better accommodates for time-series data and optimized storage.
V. DISCUSSION

The popularity and growth of IoT devices commercially is increasing over time. For example, devices that track and monitor with biometric data and health management have become more and more popular in our daily life [24]. Our model uses certificates on Ethereum to solve the issue of storing and accessing large amounts of data while maintaining secure standards for data-integrity and data-privacy. Adopting this architecture also eliminates the need for having physical access to the IoT device since the data generated is stored on a decentralized cloud service.

However, there are some challenges that haven’t been addressed in the proposed system and can potentially be improved in future implementations. Maintaining a record of Certificate revocation lists on the blockchain can be useful if the IoT device is later blacklisted by the manufacturer. In addition to revocation lists, certificate data can be shared amongst owners or organizations by using a secure key management infrastructure. These forms of improvements can enhance our current model for efficient data-management.

Improvements can also be made in testing our architecture for a larger and diverse set of IoT devices. This will allow us to evaluate performance of transactions on the Ethereum platform. The results of the experiment conducted in this paper can be further optimized by employing a computationally powerful computer that can process a large number of transactions faster and efficiently.

VI. CONCLUSION

The blockchain, as an underlying data structure to the proposed solution provides an immutable log of all IoT data operations including sensor data creation and IoT data access [25]. The use of Ethereum, decentralized cloud (IPFS) and Web3 JavaScript API for this proposed solution ensures security and a trusted service due to Ethereum’s consensus algorithm. In addition, the decentralized cloud can achieve scalability and allow users to store large chunks of data over time. In addition, getting rid of a central server, outsourcing search queries to smart contract yields an immutable result, and requires no further verification by the data owner [7]. In this paper we propose a solution for efficiently storing data generated by IoT devices through certificate management. We use the Ethereum platform to issue, store and access certificate data. To retrieve the data stored on the blockchain, we utilize a data manager and a decentralized cloud service that interacts with the smart contract. The proposed architecture provides an efficient solution for managing and scaling data generated by IoT devices.

REFERENCES


