Poster: An Efficient Key Management Scheme for IPFS-Blockchain

YoHan Park
Dept. of Computer Engineering,
College of Engineering, Keimyung University,
Republic of Korea
yhpark@kmu.ac.kr

Abstract

Blockchain is an emerging and promising technology in providing a solution to verification of data. However, it is infeasible to store digital contents on blockchain. IPFS is a promising solution to share large data in blockchain. And data uploaded to IPFS should be encrypted protected from unauthorized access. Unfortunately, there are some inefficiencies in sharing data using encryption with symmetric key and public key. In this paper, we propose an efficient key management scheme. The proposed scheme helps to revoke and update the key easily, and not to encrypt and upload repeatedly. Therefore, it could be usefully applied to IPFS-blockchain.

1 Introduction

The blockchain proposed by Nakamoto [1] is the fundamental technology of the cryptocurrency bitcoin. It provides a decentralized and distributed ledger with integrity, resiliency and credibility. The nature of the blockchain has been considering as a key to provide a solution for authenticating digital information [2, 3, 6]. It becomes a distributed ledger for anyone who can access and utilize to verify stored data and content [1]. Especially, the adoption of blockchain with medical environments can provide promising solutions to facilitate healthcare delivery [2].

However, blockchain is an expensive medium for data storage. It is inefficient to store large data and digital content in blockchain [3]. Blockchain may be complex to maintain and verify. To support data storage problem in blockchain, IPFS (Inter-Planetary File System), which is a content-addressable, distributed file system [4] has been introduced. It is a robust and promising solution to build a file sharing system in the blockchain network [7].

IPFS has no central server and data separated and distributed are stored in different IPFS nodes. Therefore, data are safe from single-server disorder and users can share big-size data with efficiency [4]. Files stored in IPFS have their own unique hash value respectively. This distinctive value helps to retrieve data from distributed data chunks.

Data uploaded to IPFS should be encrypted protected from unauthorized access. Many applications using blockchain and IPFS encrypt data with a symmetric key or a public key. Let assume that Alice wants to share a content with Bob and Chery using IPFS-blockchain. If Alice uses a symmetric key for encryption, she can share the key with both Bob and Chery. Unfortunately, the key is exposed by mistake of Bob. Then Alice should revoke the key and re-encrypt the content with another key. Then Alice uploads again the re-encrypted content and distribute the renew key. This key update and revocation process is too complicate and complex when using a symmetric key. Let assume once again that Alice uses a public key for encryption. Alice encrypts with Bob's public key and uploads to IPFS to share a content with Bob. Alice encrypts again with Chery's public key and uploads to share it with Chery. This process is also inefficient to share a content using IPFS-blockchain.

In this paper, we propose an efficient key management scheme using Shamir's threshold technique [5]. We generate secret shares of the encryption key and distribute them in IPFS. Anyone who has an ephemeral key can reconstruct the encryption key and decrypt a content. The ephemeral key can be easily revoked and re-generated. And the encrypted content stored in IPFS is not need to re-encryption for key disclosure problem using a symmetric key and public key.

2 Preliminaries

In this section, we introduce a Shamir's threshold scheme [5]

Shamir's (t,n)-SS. In Shamir's (t,n)-SS based on a Lagrange interpolating polynomial, there are n shareholders $\mathcal{U} = \{\mathcal{U}_1, ..., \mathcal{U}_n\}$ and a mutually trusted dealer D. The scheme consists of two algorithms:

1) **Share generation algorithm:** Dealer D does the following:
   - Dealer D picks a polynomial $f(x)$ of degree $(t-1)$ randomly: $f(x) = a_0 + a_1x + ... + a_{t-1}x^{t-1}$, in which the secret $K = a_0 = f(0)$ and all coefficients $a_0, a_1, ..., a_{t-1}$ are in a finite field $\mathbb{F}_p = GF(p)$ with p elements.

International Conference on Embedded Wireless Systems and Networks (EWSN) 2020
17–19 February, Lyon, France
ISBN: 978-0-9949886-4-5

- D computes all shares: $K_i = f(s_i) \pmod{p}$ for $i = 1, \ldots, n$.
- Then, D outputs a subset Ω of size n, $(K_1, K_2, \ldots, K_n)$, and distributes each share K_i to corresponding shareholder s_i privately.

2) **Secret reconstruction algorithm:** Based on Lagrange interpolation, any subset $A \subset \Omega$ of size t can reconstruct the polynomial $f(x)$ as

$$f(x) = \sum_{i \in A} \lambda_i(x) K_i \pmod{q},$$

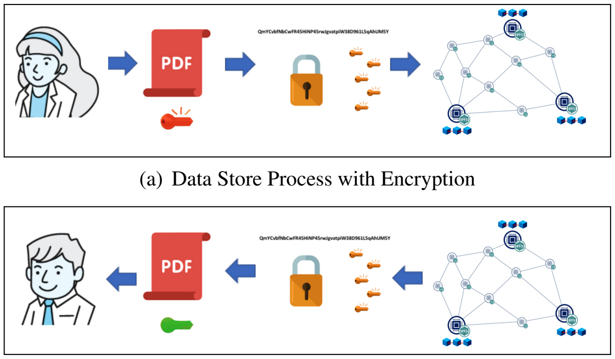
where $A = \{1, \ldots, t\} \subseteq \{1, 2, \ldots, n\}$, $\lambda_i(x) = \prod_{j \in A \setminus i} \frac{s_j - x}{s_j - s_i}$ is called a Lagrange coefficient. The secret K can be reconstructed by computing $f(0)$.

We note that the above scheme satisfies the basic security requirements of secret sharing schemes. For more information on this scheme, readers can refer to the original paper [5].

3 Proposed Scheme

In this section, we propose an efficient key management scheme in IPFS.

Figure 1 shows the proposed process of data store in IPFS with encryption and decryption. In Figure 1 (a), Alice uploads data encrypted with a secret key to IPFS and n secret shares to IPFS as well. Secret shares are distributed, along with separated data chunks. In Figure 2 (b), Bob who wants to get data uploaded by Alice asks a hash string for downloading. Then IPFS sends reconstructed data and $t-1$ secret shares to Bob. Finally, Alice sends an ephemeral key to Bob, then Bob reconstructs a secret key using $t-1$ secret shares from IPFS and an ephemeral key from Alice. We assume that secret shares are distributed to each storage node of IPFS and no more $t-1$ secret shares allows to get together in IPFS.


PDF
QmYCvbfNbCwFR45HiNP45rwJgvatpiW38D961L5qAhUM5Y
PDF
QmYCvbfNbCwFR45HiNP45rwJgvatpiW38D961L5qAhUM5Y

(a) Data Store Process with Encryption

(b) Data Reconstruction Process with Decryption

Figure 1. Process of Data Store in IPFS

The details are as follows:

- **Data Upload** : Alice generates a secret key $K \in \mathbb{Z}_p^*$. Then She determines a random polynomial, $f(x) = K + \sum_{i=1}^{t-1} a_i x^i \pmod{q}$, and computes secret shares $K_j = f(r_i)$, where r_i is a random string and $(1 \le r_i \le n)$. Then Alice encrypts data with a secret key K and uploads an encrypted data, $C = Enc_K(Data)$, and secret shares K_j to IPFS.
- **Data Download** : Bob who wants to get data uploaded by Alice asks Alice of a hash string for data and an ephemeral key. Alice generates an ephemeral key, $K_{emp} = f(R)$, where R is a random string. Bob can find the complete encrypted data C stored in IPFS via the hash string of data and collect $t-1$ secret shares. Then Bob reconstruct a secret key K using $t-1$ secret shares and the ephemera key :

  $$f(x) = \sum_{j \in A} \lambda_j(x) K_j(r_i) \pmod{q},$$

  where $A = [\{1, \ldots, t\} \subseteq \{1, 2, \ldots, n\}, R]$, $\lambda_j(x) = \prod_{k \in A \setminus j} \frac{r_k - x}{r_k - r_j}$ is called a Lagrange coefficient. The secret K can be reconstructed by computing $f(0)$. Finally, Bob decrypt data from encrypted data using the reconstructed key.

  Alice may regenerate an ephemeral key R' if she wants to revoke the key sent to Bob. Alice does not need to re-encrypt data stored in IPFS because she just generate and use another ephemeral key R' for decryption. Alice also share data with various users using ephemeral keys distributed to each user respectively. Each ephemeral key can be used to decrypt data, thus Alice encrypts and uploads no more for each user using their public key.

4 Conclusions

This paper proposed an efficient key management scheme using threshold scheme in IPFS-blockchain. Proposed scheme can solve the key revocation problem using a symmetric key. And it helps not to encrypt and upload data repeatedly using a public key. We believe that the proposed scheme improves the efficiency of IPFS, and that it could be usefully applied to blockchain.

5 Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2018R1D1A3B07050409)

6 References

[1] S. Nakamoto et al. Bitcoin: A peer-to-peer electronic cash system. 2008.

[2] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne. Blockchain for secure ehrs sharing of mobile cloud based e-health systems. *IEEE Access*, 2019.

[3] N. Nizamuddin, H. R. Hasan, and K. Salah. Ipfs-blockchain-based authenticity of online publications. In *International Conference on Blockchain*, pages 199–212. Springer, 2018.

[4] N. Nizamuddin, H. R. Hasan, and K. Salah. Ipfs-blockchain-based authenticity of online publications. In *International Conference on Blockchain*, pages 199–212. Springer, 2018.

[5] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[6] J. Xu, K. Xue, S. Li, H. Tian, J. Hong, P. Hong, and N. Yu. Healthchain: A blockchain-based privacy preserving scheme for large-scale health data. *IEEE Internet of Things Journal*, 6(5):8770–8781, 2019.

[7] R. Yang, F. R. Yu, P. Si, Z. Yang, and Y. Zhang. Integrated blockchain and edge computing systems: A survey, some research issues and challenges. *IEEE Communications Surveys & Tutorials*, 21(2):1508–1532, 2019.