

Hybrid Blockchain-based Unification ID in Smart Environment

Nakhoon Choi*, Heeyoul Kim*

* Department of computer Science and engineering, Kyonggi University, Korea

skrgns0411@naver.com, heeyoul.kim@kyonggi.ac.kr

Abstract— Recently, with the increase of smart factories, smart cities, and the 4th industrial revolution, internal user authentication is emerging as an important issue. The existing user authentication and Access Control architecture can use the centralized system to forge access history by the service manager, which can cause problems such as evasion of responsibility and internal corruption. In addition, the user must independently manage the ID or physical authentication medium for authentication of each service, it is difficult to manage the subscribed services. This paper proposes a Hybrid blockchain-based integrated ID model to solve the above problems. The user creates authentication information based on the electronic signature of the Ethereum Account, a public blockchain, and provides authentication to a service provider composed of a Hyperledger Fabric, a private blockchain. The service provider ensures the integrity of the information by recording the Access History and authentication information in the Internal-Ledger. Through the proposed architecture, we can integrate the physical pass or application for user authentication and authorization into one Unification ID. Service providers can prevent non-Repudiation of responsibility by recording their authority and access history in ledger.

Keywords— Blockchain, Identity, Ethereum, Hyperledger

I. INTRODUCTION

According to the report of the ‘smart factory industry and market trend’ [1] Korea Technology Transfer Centre for National R&D Programs in 2018, the domestic smart factory demand market is expected to grow 11.2% per year by 2020 and 8% annual growth of the global smart factory demand market.

Smart Factory have the advantage that automated processes reduce product defect rates and allow for on-site operation by a small number of highly authorized personnel. However, there are problems such as control of employees' access to Smart Factory and production facilities, and responsibility for accidents caused by high authority. Currently, most organizations and enterprises use a server-based centralized architecture for Access Control and user authentication. Such a centralized architecture enables denial of access history through log manipulation and deletion by the administrator, thus avoiding responsibility for accidents. Therefore, the existing authentication and Access Control model is not suitable as a means of authentication and Access Control of Smart Factory.

In order to solve the above problems, this paper records the access history using the integrity of the blockchain, and

integrates the authentication means used for each Smart Factory, or facilities, through the hybrid blockchain structure. This allows you to create a Unification ID to replace a physical pass or an application for access, record the history, and use it for future proof.

II. BACKGROUND

A. Blockchain

In this paper, we use the blockchain for the integration of user ID and decentralization of authority record, and the public blockchain using consensus algorithm represented by PoW (Proof of Work) [2]. The blockchain ensures the integrity of the information contained in the block through consensus among untrusted participants on the network and selects the block creator to maintain a decentralized network without administrators. We use Ethereum [3], a public blockchain platform, for the proposed model. Unlike Bitcoin, which supports Turing incomplete scripts that can only perform simple operations, Ethereum supports the creation and distribution of Smart Contracts [4] for developing DApp (Decentralized Application). Written in distributed Ledger written in Solidity, a Turing complete language, Smart Contract ensures automated execution and integrity of results.

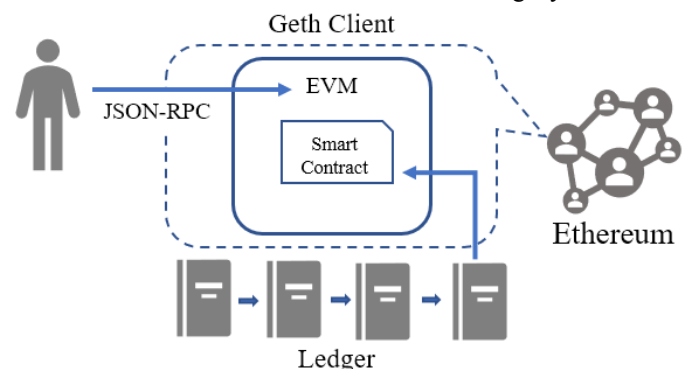


Figure 1. Operation of Ethereum Smart Contract

Smart Contract written in Solidity is compiled and stored in Ledger in the form of EVM (Ethereum Virtual Machine) Byte Code to ensure the integrity of the contract. Geth (Go-Ethereum), an Ethereum Client, calls the Byte Code of the Smart Contract and executes it in the EVM environment, ensuring automated execution and the integrity of the results according to the contract. Users operate Smart Contracts without Ethereum Clients on websites that can be accessed

through MetaMask [5], replacing authentication and information storage in a decentralized way.

B. Hyperledger Fabric

Hyperledger Fabric [6] is a blockchain technology development project for enterprise among the Hyperledger projects led by the Linux Foundation. It is the most popular open source-based private blockchain platform. Unlike the structure of public blockchain that anyone can participate in, it uses a certificate and PKI to verify the participant with a licensed network and is managed through the Membership Service Provider (MSP).

The Hyperledger Fabric refers to the companies and organizations that make up the network as "Organization". Organizations have their own set of peers and use peers to form a network. Blocks created in the blockchain network are delivered to each peer and stored. Peers of each Organization ensure data integrity and enable distributed storage of data through verification of each other's blocks. Also, unlike PoW of Public Blockchain(e.g. Bitcoin.), Hyperledger Fabric does not require much resources for consensus for block creation, and it collects transactions of peers through ordering service and creates them in blocks and distributes them to the network.

III. PROPOSAL MODEL

A. Proposal Model Overall

The proposed model is composed of Ethereum (Public) - Hyperledger Fabric (Private) Hybrid-Blockchain. Private layer means internal system such as each Smart Factory and Smart City.

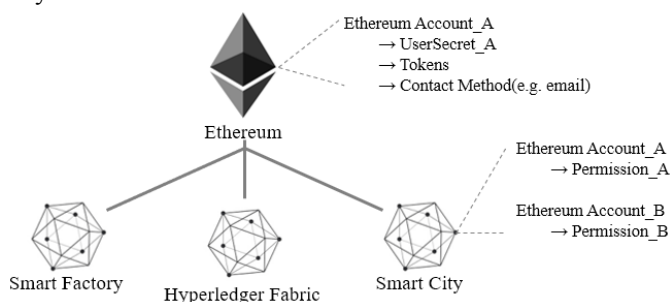


Figure 2. Proposal Model Overall

The user creates a Unification ID through Ethereum Smart Contract and sets authorization based on the Unification ID by user authentication of the organization composed of Hyperledger. Set authority information is saved in Ledger of Hyperledger, and user can prove authority to several organizations through one Ethereum Account. In addition, Token, information about authority, is stored in Ethereum ID and used as proof of history.

B. Proposal Model Layer

1) **Unification Management Layer - Ethereum:** Integrated management layer, Ethereum stores the information in Table 1 is based on the User Account through the Smart Contract.

TABLE 1. USER IDENTIFICATION

Identification	Meaning
UserSecret	Identifying information for initial authentication of users
Token	Identifying information of the organization issued for secondary authentication of user
Contact Method	How users receive tokens

$$\text{UserSecret} = \text{Keccak-256}(\text{Name}, \text{Birth}, \text{PhoneNum}) \quad (1)$$

UserSecret is created through (1) and does not expose personal information through the irreversibility of Keccak256 Hash Function. In addition, Smart Contract performs access control by comparing the transaction sender account for the above information with the ID creator's account.

TABLE 2. SMART CONTRACT FUNCTION

Function	Meaning
registration	Create a User ID and save the UserSecret to Ethereum
editSecret	Modify the saved UserSecret, check the revision-transaction generator, and perform access control
addToken	Store Token for secondary authentication in Ethereum

Table 2 shows the functions that operate on the surface of Ethereum Smart Contract.

2) **Permission Layer - Hyperledger Fabric:** Each department of the Group, such as Smart Factory and Smart City, forms the Organization of Hyperledger, and creates a Channel consisting of the Consortium for access control and peers of each department. The node of the Ordering service is managed by the Group manager. Privileges and access history are stored in Ledger, and all participants in the network are given read permission to the Ledger. Each Group meets the requirements of Table 3.

TABLE 3. REQUIRED COMPONENTS

Components	Meaning
Internal Information System	Creates a Secret by processing the information provided by the user's group subscription according to (1)
Ethereum Sync Node	User authentication is performed by reading a Secret and a Token stored in the Ethereum smart contract by Ethereum Node.
Admin	Hold a key that can represent an organization to sign a Token for secondary authentication. It also establishes appropriate access rights for the position for authorized users.
QR Reader	Access Control to objects is performed and includes ecRecover function.

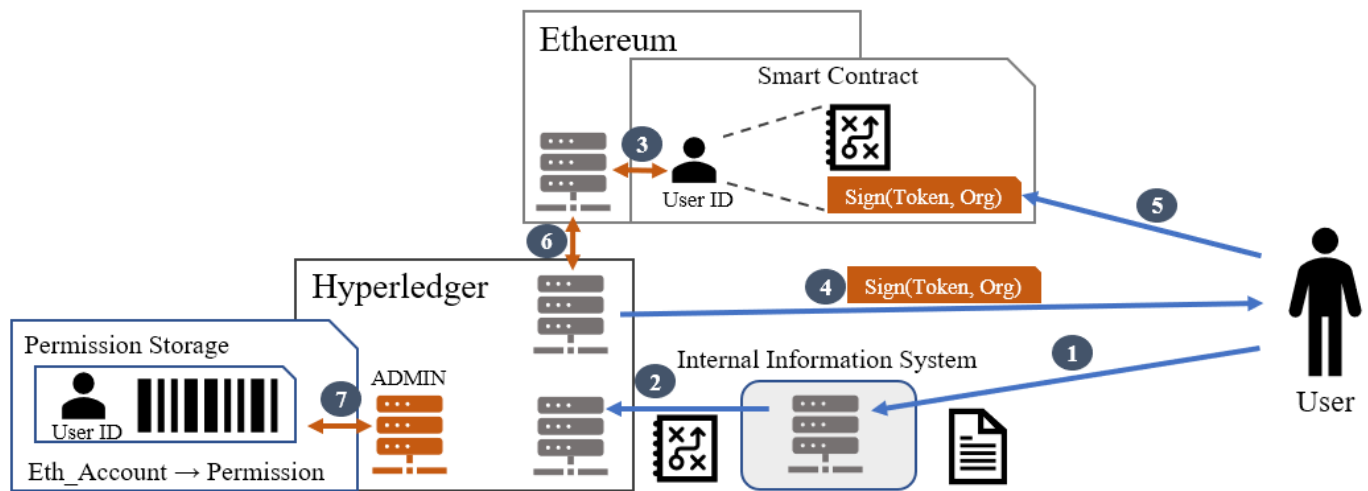


Figure 3. Permission Setting Flow

In addition, Group of the Legacy System can be included in the permission layer, and the compatibility conditions are shown in Table 4.

TABLE 4. REQUIRED COMPONENTS IN LEGACY SYSTEM

Components	Meaning
Ethereum Sync Node	User authentication is performed by reading a Secret and a Token stored in the Ethereum smart contract by Ethereum Node.
PKI	Public Key Infrastructure (PKI) that can represent organizations signing tokens for secondary authentication
QR Reader	Access Control to objects is performed and includes ecRecover function. Performs an equivalent role as the Client App of Hyperledger Fabric

C. Unification ID Create

Before using the proposed model, the user creates Unification ID based on his Ethereum Account in the Smart Contract through Web DApp and registers Secret and Contact Method(How to get your token shipped). This paper registers email in the Contact Method and receives the token of the organization.

D. Permission Setting

Figure 3 shows how a user who completes the ID Create process is granted access based on their Ethereum ID in a new organization.

- User Alice joins an organization that uses Hyperledger (e.g. Smart Factory) and provides personal information including Secret generation standards (Name, Birth, PhoneNum).
- Node linked with Smart Factory's Internal Information System generates Secret according to generation standard based on the above information and delivers it to Ethereum Sync Node of Hyperledger Fabric.

- Ethereum Sync Node checks whether the created Secret is registered in Smart Contract.
- If Alice is a registered user in Ethereum, she creates a token for each user who is digitally signed as shown in (2) as the administrator's key of Hyperledger or the representative key of the Group for the second authentication of the ID.

$$\text{Token} = \text{Sign}(\text{Keccak-256}(\text{Seed_Alice}), \text{Org}) \quad (2)$$

- ⑤ Alice registers the issued token in her ID.
- ⑥ The Ethereum Sync Node accesses Alice's ID on the Smart Contract and checks if the issued token is registered.
- ⑦ Hyperledger's Admin stores permission information in Permission Storage (ACL: Access Control List) based on Alice's Ethereum Account.

E. Access Control

Figure 4 is the process in which Alice, whose Permission Setting has been completed, performs access control to the Smart Factory's Access-System.

- ① User Alice creates a QR through the QR Generator App registered with her private key and performs authentication for access or operation of the Smart Factory. QR includes Timestamp and Sign (Timestamp, Alice) to prevent Replay Attack.
- ② QR Reader reads Alice-generated QR and delivers it to Hyperledger's Access Control Node with QR_Serial_Number.
- ③ Access Control Node verifies Signature Data (r, s, v). It also restores Alice's public key and account through ECDSA-based public key recovery algorithm ecRecover [7].
- ④ Retrieve the restored account from Permission Storage and check the access right for QR_Serial_Number.
- ⑤ When the access right is confirmed, record the access history in the history storage and grant access.

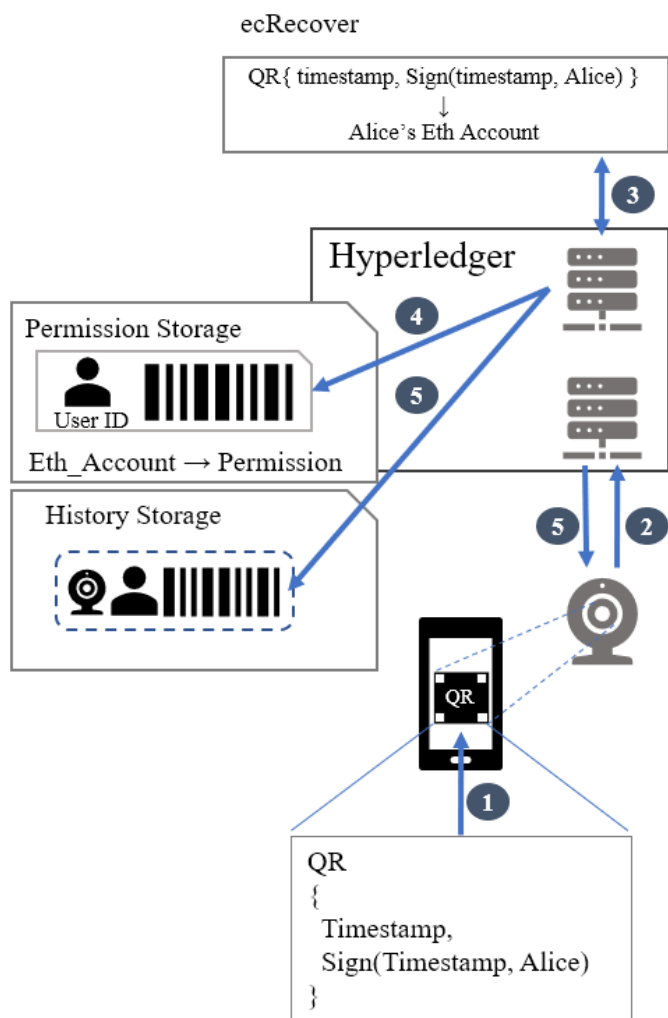


Figure 4. Access Control Flow

IV. IMPLEMENTATION

In this paper, a prototype was implemented to verify the behaviour of the proposed model.

A. Development environment

TABLE 5. DEVELOPMENT ENVIRONMENT

	Environment	Language
WebApp	Chrome, MetaMask	JavaScript
Smart Contract	Ethereum Ropsten Testnet	Solidity ^0.5.0
Chaincode	Hyperledger Fabric 1.4	Node.js
QR App	Android Emulator	Java
QR Reader	Raspberry Pi	Node.js

B. Implementation

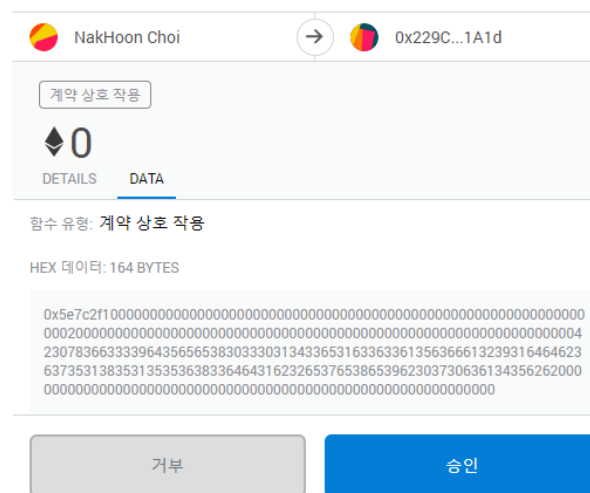


Figure 5. Create Unification ID in MetaMask

Figure 5 shows the user creating a transaction through MetaMask to register Secret and Contact Method in Web DApp.

```

=== newUser Detected ===
Name : Alice
Address : Suwon, Kyonggi
Phone : 010-0000-0000
User MetaInfo : 0x6c39d5ee8030143e1c63a5cfa291ddb675185155683dd1b2e7e8
Serving to Hyperledger...
Success.

```

Figure 6. Log of User Join an Organization

Figure 6 shows the log that Internal Information System sends user's Secret to Hyperledger Fabric when user joins group.

```

=== newUser Detected ===
User MetaInfo : 0x6c39d5ee8030143e1c63a5cfa291ddb675185155683dd1b2e7e8
Searching in Ethereum...
=== User Detected ===
User Address : 0xF2563715Ca207a40efb2008b35D590766d7D01e3
Contact Method : skg4463@gmail.com
Serving Token : 0xcF9F37229CcF9F374542174547af815835817af81583d581c9bE7C

```

Figure 7. Log of User Registration Confirmation

When Secret is delivered to Hyperledger through Figure 6, Ethereum Sync Node checks if the above Secret is a registered user in Smart Contract, and sends the Token containing the organization's signature to the user's Contact Method (Email). Token is registered through addToken function of.

```

=== Token Registration Detected ===
User Address : 0xF2563715Ca207a40efb2008b35D590766d7D01e3
User Name : Alice
Token Confirm : 0xcF9F37229CcF9F374542174547af815835817af81583d581c9bE7C
Permission Set.

```

Figure 8. Log of Permission Setting

As shown in Figure 8, when the user registers the received token and completes the second authentication, the Ethereum Sync Node checks this and records the Ethereum Account and access rights in the Hyperledger's Permission Storage.

```

QR{
q0 : 0x9c304714ae532199b8cb0dd67eaf3248fe4d77b15672633855c
8c579329eedff5145c2d9f502e5d80c1a4f8422dc426f7174161c701f1
ea6b711adddb5f5ff621c,
q1 : 1570294439,
q2 : 0001
}
q0 : signature data,
q1 : UNIX timestamp,
q2 : QR Reader Serial Number

```

Figure 9. QR Generator Information

Figure 9 shows the information contained in the QR Code generated by the user through the QR Generator App. q0 is user's Signature Data (r, s, v), q1 is Timestamp used for Signature, q2 is Serial Number of QR Reader which read QR Code.

```

=== User Access Detected ===
QR Signature : 0x9c304714ae532199b8cb0dd67eaf3248fe4d77b15672633855c8c5793
QR Timestamp : 1570294439
QR Serial : 0001

ecRecovering...
Recovered User Address : 0xf2563715ca207a40efb2008b35d590766d7d01e3
0xf2563715ca207a40efb2008b35d590766d7d01e3 is 'Alice', An authorized User.
History Save.

```

Figure 10. Log of ecRecover

QR Reader sends q0-2 in Figure 9 decrypted from QR to the Access Control Node. The Access Control Node recovers the user's public key and account through ecRecover. After that, access control is performed based on Serial of QR Reader. In Figure 10, you can see that your account has been successfully recovered from QR.

V. SCALABILITY

The proposed model of this paper is aimed at Unification ID in smart environments. From a broader perspective, it integrates the authentication methods required for various smart factories, companies and organizations in the smart city, and manages authentication and access history. Currently, the details are stored only in the Private layer, but by additionally storing them in the Public layer, all access details in the smart city can be integrated and managed.

VI. CONCLUSIONS

As smart environments such as Smart Factory and Smart City have increased, the possibility of manipulating the access history of the centralized model for access and access control has become a problem. We proposed a model using the blockchain, a decentralized platform, to ensure the integrity and transparency of information.

The proposed model integrates authentication means for access control in Smart Environment through Hybrid Blockchain. This allows users to prove their authority through a single Ethereum Account without creating a physical pass or application for authorization of multiple organizations. In addition, the access history and access to the facilities of the Smart Factory are stored in the blockchain, thus preventing non-repudiation of responsible materials in the event of an accident.

ACKNOWLEDGMENT

This research was supported by Ministry of Science ICT Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(2018R1C1B6002903)

REFERENCES

- [1] Hyeon-gyu Lee, Smart Factory Industry and Market Trends, Korea Technology Transfer Centre for National R&D Programs .2018
- [2] NAKAMOTO, Satoshi, et al. Bitcoin: A peer-to-peer electronic cash system. 2008.
- [3] The Ethereum Foundation. "White Paper: A Next-Generation Smart Contract and Decentralized Application Platform" website.[Online].Available: <https://github.com/ethereum/wiki/wiki/White-Paper>
- [4] SZABO, Nick. Smart contracts: building blocks for digital markets. EXTROPY: The Journal of Transhumanist Thought,(16), 1996, 18: 2.
- [5] METAMASK. website.[Online].Available: <https://metamask.io>
- [6] (2019)Hyperledger-fabric read the docs website.[Online].Available: <https://hyperledger-fabric.readthedocs.io/en/release-1.4/>
- [7] Danial R. L. Brown. "SEC1: Elliptic Curve Cryptography," Standard for Efficient Cryptography, pp. 47-48, May. 2009

Nakhoon Choi is in the undergraduate course of Computer Science Department, Kyonggi University, His major research interests include security and blockchain.



Heeyoul Kim received the B.E. degree in Computer Science from KAIST, Korea, in 2000, the M.S. degree in Computer Science from KAIST in 2002, and the Ph.D. degree in computer science from KAIST in 2007. From 2007 to 2008, with the Samsung Electronics as a senior engineer. Since 2009 he has been a faculty member of Department of Computer Science at Kyonggi University. His major research interests include cryptography, security and blockchain.