A Blockchain-based Access Control Scheme

Xiong Jiang
Software Engineering, Tongji University, Shanghai, Shanghai, 201800, China
1831589@tongji.edu.cn

Abstract. Access control plays an important role in protecting data privacy. However, traditional access control models such as RBAC, ABAC and UCON need to be built based on a central trusted server. Once the central trusted server is controlled by an adversary, it will pose a serious threat to data security. Therefore, this article proposes a blockchain-based access control scheme, which uses the blockchain to replace the central trusted server and uses smart contracts to complete access. The proposed scheme solves the centralization problem of the traditional access control model. Furthermore, it has been tested on representative equipment, and the experimental results show that the proposed scheme is efficient and feasible.

1. Introduction
Access control technology is one of the countermeasures used to protect user privacy. The traditional access control model is usually based on a trusted center which stores access policies and grants permissions based on the access policies. RBAC, ABAC, UCON are three common access control strategies. RBAC [1] introduces the concept of "role" between users and permissions. Users obtain corresponding permissions based on the roles defined by the system. Each user is associated with one or more roles, and each role is associated with one or more permissions. Therefore, RBAC can realize flexible rights management. ABAC [2] introduces attributes between users and permissions, and permissions are granted by judging whether the attributes meet certain constraints. In addition, ABAC can achieve different granular authority control according to requirements. Different from RBAC and ABAC, UCON [3] introduces variable attributes, which will change with the result of accessing objects, while immutable attributes can only be changed through management behaviors. UCON solves the problem of the variability of attributes during user access.

However, RBAC, ABAC and UCON have a common drawback, that is, a trusted authorization center needs to be set up. Once the trusted authorization center is captured by an opponent, it will pose a serious threat to the security of resources. Therefore, a decentralized access control countermeasure needs to be considered to solve this drawback. Blockchain is a recording system based on a distributed ledger that is shared, replicated, and continuously synchronized between peers (ie, software agents) hosted by distributed network participants. In the blockchain, the distributed ledger records transactions between participants in the distributed network. For example, the exchange of documents or data between participants in the distributed network will be treated as a transaction record by all participants in the distributed network. In the blockchain ledger, each record contains a timestamp and a digital signature, so that all transactions in the blockchain network can be audited and transaction records are difficult to tamper with once written into the ledger. In addition, through decentralization, the blockchain improves the efficiency and availability of data distribution. The blockchain architecture diagram is shown in the Figure 1 as follows.
Figure 1. The architecture of blockchain

At present, the schemes for constructing access control based on blockchain technology are mainly divided into two categories. The first category is by replacing the trusted center in the traditional access control model with a blockchain. Cruz et al. [4] proposed an RBAC using smart contracts to solve the problem of cross-organization access control in traditional RABC. Similarly, Alansari et al. [5] proposed a blockchain-based ABAC, which uses the non-tamperable feature of block data, so that the access policy can not be changed by malicious users. The second category is to not only use the blockchain as the trusted center in the access control model, but also implement the access control policy by constructing smart contracts. Dorri et al. [6] proposed a blockchain-based access control model. The core idea is to store the access control policy in the blockchain, and users can call smart contracts through blockchain nodes to complete access. Each access record will be written into the block. Similarly, the access control scheme based on blockchain also has [7-12].

2. Preliminaries

In this section, we introduce the preliminaries of the proposed scheme briefly as follows. We denote a digital signature scheme consists of KeyGen, Sign and Verify by $DS = (\text{KeyGen}, \text{Sign}, \text{Verify})$. KeyGen represents the algorithm for key generation, it takes system security parameter as input, then outputs a public/private key pair denoted by $(pk, sk)$. We denote the message space by $M$; Sign represents the algorithm for signature generation, it takes a message denoted by $m$ from $M$ and private key $sk$ as input, outputs a signature of $m$ denoted by $\text{sig}_m$. Verify represents the algorithm for signature verification, it takes a message $m$, corresponding signature $\text{sig}_m$ and public key $pk$ as input, outputs "accept" represents successful verification or "reject" represents fail verification. We denote the random number generation algorithm by idGen, it outputs a random number with a range.

3. Proposed scheme

In this section, we describe a blockchain-based access control scheme, which is mainly composed of three smart contracts as follows. Let Alice denote the resource owner, Bob denote the resource visitor, and $N$ denote the blockchain node. The working flow of the proposed scheme is shown in Figure 2.
3.1. Smart contract of registration

The smart contract of registration is denoted by \texttt{Re\_gister}, which consists of a digital signature scheme \( DS = (\text{KeyGen}, \text{Sign}, \text{Verify}) \) and a random number generation algorithm \( \text{idGen} \); it is executed interactively among Alice, Bob, and \( N \) through the following steps:

- Install the smart contract \texttt{Re\_gister} on node \( N \).
- Bob sends a registration request to node \( N \).
- Node \( N \) sends the registration request to Alice.
- Alice calls the smart contract \texttt{Re\_gister} through node \( N \) to obtain the algorithm \( \text{idGen} \), and then runs \( \text{idGen} \) locally to generate a unique identification \( id \).
- Alice calls the smart contract \texttt{Re\_gister} through node \( N \) and sends the \( id \) to Bob.
- After receiving the \( id \), Bob calls the smart contract \texttt{Re\_gister} through node \( N \) to obtain the algorithm \( \text{KeyGen} \), and then runs \( \text{KeyGen} \) locally to generate a public-private key pair \((pk, sk)\).
- Bob calls the smart contract \texttt{Re\_gister} through node \( N \), obtains the algorithm \( \text{Sign} \), and then runs \( \text{Sign} \) locally to generate the signature \( \text{sig}_{id} \).
- Bob calls the smart contract \texttt{Re\_gister} through node \( N \), and sends the public key \( pk \) and signature \( \text{sig}_{id} \) to Alice.
- After receiving the public key \( pk \) and the signature \( \text{sig}_{id} \), Alice calls the smart contract through node \( N \) to execute the algorithm \( \text{Verify} \); if the verification passes, Alice inserts the two-tuple \((id, pk)\) into the local database \text{legal\_member\_DB} ; otherwise, it returns registration failure notification.
- Alice calls the smart contract \texttt{Re\_gister} through node \( N \), and writes the information "(\( id \), \( pk \)) has been registered" into the block.

3.2. Smart contract of access

The smart contract of access is denoted by \texttt{Access} which is composed of the signature algorithm \( \text{Sign} \) and the verification algorithm \( \text{Verify} \). It is used to determine whether the visitor's identity is
legal. If the identity of the visitor is legal, it allows the visitor to access the resource; otherwise, it returns a notification that the identity is illegal. The smart contract Access is executed interactively among Alice, Bob, and \( N \) through the following steps:

- Install the smart contract Access on node \( N \).
- Bob calls the smart contract Access through node \( N \), obtains the algorithm Sign, and then executes Sign locally to generate the signature \( \text{sig}_{id} \).
- Bob calls the smart contract Access through node \( N \), and sends \( id \) and \( \text{sig}_{id} \) to Alice.
- After receiving \( id \) and \( \text{sig}_{id} \), Alice goes to the local database legal_member_DB to find the corresponding public key \( pk \) according to the \( id \).
- Alice calls the smart contract Access through node \( N \) and executes the algorithm Verify; if the verification is passed, it returns to allow access, and writes the information "\( id \) successfully accessed to the database" into the block; otherwise, it returns illegal access.

3.3. Smart contract of revocation
The smart contract of revocation is denoted by Revoke. This smart contract is used to revoke the access rights of registered legitimate visitors to resources. It is executed interactively among Alice, Bob, and \( N \) through the following steps:

- Bob calls the smart contract Revoke through node \( N \), and sends \( id \) and \( pk \) to Alice.
- After receiving \( id \), \( pk \), Alice deletes the two-tuple \((id, pk)\) from the local database legal_member_DB.
- Alice calls the smart contract Revoke through node \( N \), and writes the information "\( (id, pk) \) has been revoked" into the block.

4. Experimental evaluation
In this section, we conduct an experimental evaluation of the proposed scheme from two aspects. The first aspect is the implementation of the proposed scheme; the second aspect is the introduction of experimental methods and the analysis of experimental results.

4.1. Implementation of the proposed scheme
The smart contracts in the proposed scheme are all implemented by golang. The reason for choosing golang is that golang has good cross-platform features and provides a rich library of cryptographic algorithms. In addition, many blockchains support smart contracts implemented by golang such as Hyperledger Fabric.

4.2. Experimental methods and results
In the experiment, we set up three roles: resource owner Alice, resource visitor Bob, and blockchain node \( N \). Furthermore, we set the digital signature scheme as the standard ECDSA scheme and use the random number generation algorithm provided from the standard ECDSA scheme as \( \text{idGen} \). We record the number of \( id \) in legal_member_DB as \( n \), and then explore the impact of changes in \( n \) on the execution time of the three smart contracts Register, Access, and Revoke in the proposed scheme. We have performed the above experiments on representative equipment, and the parameters of the equipment are shown in Table 1.

| Equipment | Operating System | Processor | RAM |
|-----------|------------------|-----------|-----|
| A         | Macbook pro      | macOS Catalina 10.15.2 | Intel Core i5 @ 2.4 GHz | 16 GB |
| B         | Workstation      | Ubuntu 18.04    | Intel Core i7 @ 3.6 GHz  | 32 GB |
The experimental results are shown in Figure 3 and Figure 4 respectively. The result shows that as $n$ increases, the execution time of smart contract `Register` is not affected much; on the contrary, as $n$ increases, the execution time of smart contracts `Access` and `Revoke` also increases. The reason is that during the execution of smart contract `Register`, the main influence on the execution time is the insertion operation. Furthermore, during the execution of smart contracts `Access` and `Revoke`, the time of traverse `legal_member_DB` increases due to the increase of $n$.

![Figure 3. Execution time of smart contract in equipment A](image1.png)

![Figure 4. Execution time of smart contract in equipment B](image2.png)

5. Conclusion
In this paper, we propose a blockchain-based access control scheme, which uses the blockchain as the trusted center in the access control model and implements the access control policy by constructing smart contracts. The resource owner and the resource visitor interact through the nodes on the blockchain by invoking smart contracts to complete the access process. In addition, the proposed scheme has been tested on representative equipment, and the experimental results show that the proposed scheme is efficient and feasible.
Acknowledgments

This paper is personal research.

References

[1] Ferraiolo D, Cugini J, Kuhn D R. Role-based access control (RBAC): Features and motivations[C]//Proceedings of 11th annual computer security application conference. 1995: 241-48.

[2] Yuan E, Tong J. Attributed based access control (ABAC) for web services[C]//IEEE International Conference on Web Services (ICWS'05). IEEE, 2005.

[3] Danwei C, Xiuli H, Xunyi R. Access control of cloud service based on ucon[C]//IEEE International Conference on Cloud Computing. Springer, Berlin, Heidelberg, 2009: 559-564.

[4] Cruz J P, Kaji Y, Yanai N. RBAC-SC: Role-based access control using smart contract[J]. Ieee Access, 2018, 6: 12240-12251.

[5] Alansari S, Paci F, Sassone V. A distributed access control system for cloud federations[C]//2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2017: 2131-2136.

[6] Dorri A, Kanhere S S, Jurdak R, et al. Blockchain for IoT security and privacy: The case study of a smart home[C]//2017 IEEE international conference on pervasive computing and communications workshops (PerCom workshops). IEEE, 2017: 618-623.

[7] Ouaddah A, Abou Elkalam A, Ait Ouahman A. FairAccess: a new Blockchain - based access control framework for the Internet of Things[J]. Security and communication networks, 2016, 9(18): 5943-5964.

[8] Pinno O J A, Gregio A R A, De Bona L C E. Controlchain: Blockchain as a central enabler for access control authorizations in the iot[C]//GLOBECOM 2017-2017 IEEE Global Communications Conference. IEEE, 2017: 1-6.

[9] Ouaddah A, Abou Elkalam A, Ouahman A A. Towards a novel privacy-preserving access control model based on blockchain technology in IoT[M]//Europe and MENA cooperation advances in information and communication technologies. Springer, Cham, 2017: 523-533.

[10] Maesa D D F, Mori P, Ricci L. A blockchain based approach for the definition of auditable Access Control systems[J]. Computers & Security, 2019, 84: 93-119.

[11] Ding S, Cao J, Li C, et al. A novel attribute-based access control scheme using blockchain for IoT[J]. IEEE Access, 2019, 7: 38431-38441.

[12] Yang C, Tan L, Shi N, et al. AuthPrivacyChain: A blockchain-based access control framework with privacy protection in cloud[J]. IEEE Access, 2020, 8: 70604-70615.