Hindawi
Security and Communication Networks
Volume 2020, Article ID 8891183, 13 pages
https://doi.org/10.1155/2020/8891183

Research Article

PyRos: A State Channel-Based Access Control System for a Public Blockchain Network

Siwan Noh[1], **Sang Uk Shin**[2], **and Kyung-Hyune Rhee**[2]

[1]*Department of Information Security, Graduate School, Pukyong National University, Busan 48513, Republic of Korea*
[2]*Department of IT Convergence and Application Engineering, Pukyong National University, Busan 48513, Republic of Korea*

Correspondence should be addressed to Kyung-Hyune Rhee; khrhee@pknu.ac.kr

Received 7 August 2020; Revised 17 October 2020; Accepted 16 November 2020; Published 27 November 2020

Academic Editor: Zhe-Li Liu

Blockchain is a technology that enables the implementation of a decentralized system by replacing the role of the centralized entity with the consensus of participants in the system to solve the problem of subordination to the centralized entity. Blockchain technology is being considered for application in numerous fields; however, the scalability limitation of a public blockchain has led many researchers to consider private blockchains, which reduce the security of the system while improving scalability. A state channel represents a leading approach among several scalability solutions, intended to address public blockchain scalability challenges while ensuring the security of the blockchain network. Participants in the channel perform the process of updating the state of the channel outside the blockchain. This process can proceed very quickly because it does not require the consensus of the blockchain network, but still, like on-chain, it can guarantee features such as irreversibility. In this paper, we propose the PyRos protocol, an access control system that supports the trading and sharing of data between individuals on a public blockchain based on the state channel. As far as we know, the research using the off-chain state channel for access control has not been proposed yet, so PyRos is a new approach in this field. In PyRos, user-defined access control policies are stored off-chain, and policy updates are always rapid regardless of the performance of the blockchain network. Moreover, PyRos provides means to prevent malicious participants from arbitrarily using the channel's previous state while resolving constraints due to scalability problems, along with privacy guarantees for the transaction content. To evaluate the efficiency and security of PyRos, we provide qualitative analysis of security requirements and analysis in terms of the performance of public blockchain platforms.

1. Introduction

The development of Internet of Things (IoT) technology has enabled us to generate unimaginable quantities of data in the course of our daily lives. A variety of data is produced with smart mobile devices such as smartphones, smart bands, or devices connected to smart home networks (TVs, lights, etc.). According to a recent survey [1], the data generated in this manner is expected to reach 175 zettabytes per year by 2025. Big data is a technology that analyzes such large quantities of data to extract new information. It is used in numerous fields, including healthcare and logistics [2, 3]. However, this requires the collection of extensive user data. Machine data [4] refers to data collected through machines, such as industrial equipment, sensors, or weblogs that record users' behavior on the Web. The amount of data acquired by the dissemination of IoT devices is expected to increase exponentially.

In a traditional IT platform environment, users do not have the proper authority over their data. Global IT companies, such as Google and Facebook, or service providers have taken control of the users' data, which has caused numerous security concerns [5, 6]. The MyData industry [7] presents a paradigm in which the subject of information manages, controls, and utilizes their data based on the right to data portability of individuals instead of companies or governments. In the MyData industry, blockchain is considered as a key technology for decentralized data self-control, and numerous related projects are being proposed [8, 9].

The blockchain node collects data through peer-to-peer networks and stores it in a chain-structured distributed data storage. Characteristically, based on the consensus protocol, it is possible to implement a reliable operation among nontrust nodes without a central authority whom the nodes commonly trust. The blockchain network uses a variety of consensus protocols to solve problems arising from the absence of a central authority. Only data verified through consensus protocols is stored as new data in the blockchain.

Blockchain-based access control [10–12] is one of several blockchain-based applications. Instead of being managed by the centralized access control server for storage and control of resources, access control policies are kept and verified in the blockchain layer built above the storage layer. However, in practical terms, resources are stored outside the blockchain (e.g., cloud storage) and only access control policies are kept in the blockchain, as storing the data itself in the blockchain causes an unaffordable overhead for users on the network. The decentralization, transparency, and irreversibility of the blockchain are expected to enable the delivery of new services by overcoming the limitations of the traditional access control system.

A public blockchain, however, has the disadvantage of the absence of a system administrator, which limits the processing performance of the system. To ensure reliable operation in a blockchain network composed of only untrusted nodes, Bitcoin blockchain employed a very strong consensus protocol called Proof-of-Works (PoW); however, this resulted in only about seven transactions per second. Numerous blockchain projects have recently solved this problem by limiting the nodes of the blockchain network to authorized users (permissioned blockchain) or organizing only specific groups of users (private blockchain) [13]. This approach remains a problem that is being discussed today, as it abandons decentralization to improve scalability [14]. Scalability, decentralization, and security are called blockchain trilemma as factors that are difficult to satisfy simultaneously on the blockchain. Recently, many solutions have adopted a method that has been recentralized and security-vulnerable to improve efficiency. However, in this paper, we do not consider this approach. Because the motivation behind blockchain-based access control is to eliminate the access control server and implement user-centric access control, it is not desirable to apply recentralization solutions to access control applications. However, if access control applications are implemented on the public blockchain, the limited processing performance of blockchain networks makes it difficult for user-defined policies to be reflected without delay.

To overcome the above-mentioned problem, in this study, we propose the PyRos protocol based on the off-chain state channel, one of the blockchain scalability solutions.

To summarize, our contributions are listed as follows:

(i) We propose the PyRos protocol, an access control application that operates on a public blockchain with limited processing performance.

(ii) PyRos operates based on the off-chain state channel solution and provides a validation method for access control policies recorded on the off-chain channel.

(ii) PyRos does not sacrifice the security or decentralization of the system that operates to improve scalability.

2. Background

We present an overview and related research on blockchain-based access control, blockchain scalability limitations, and the off-chain state channel.

2.1. Blockchain and Blockchain-Based Access Control. Bitcoin [15], the most widely known cryptocurrency, records information on its ownership in the blockchain ledger. Users update the ownership information of the Bitcoin recorded in the blockchain ledger through the creation of transactions, including their digital signatures and new owner information (e.g., blockchain address), to use the Bitcoin they own. If the information contained in the transaction is valid, it will be disseminated to the majority of users of the Bitcoin blockchain network, and it will later be included in the block through mining and reflected in the blockchain ledger. The Bitcoin blockchain selects miners at certain time intervals based on the PoW algorithm to maintain a single blockchain ledger on the network. The PoW algorithm makes only single ledger exist in the network, even if several miners attempt to update their blockchain ledger at the same time. The PoW algorithm adds blocks of users, who first find values that make the cryptographic hash results of the block header exist within a certain range to the blockchain as a new block. Finally, the blockchain takes the form of a hash chain, which ensures the irreversibility and transparency of the blockchain.

The transparency and irreversibility of the blockchain can have a huge impact on improving the reliability of the database management. In particular, applying the blockchain to the access control system makes it possible to manage policies for the requester without a centralized authority. The key element of blockchain-based access control is similar to cryptocurrency. In cryptocurrency blockchain, users manage their cryptocurrency without the help of banks. The blockchain is a ledger that records cryptocurrency ownership information for all users of the network and that has been recording all details since the launch of the cryptocurrency. In contrast, blockchain-based access control records access control policies for digital objects in the blockchain ledger instead of recording ownership information for the cryptocurrency.

In [10, 11], each transaction represents the subject's right to access the object. The rights recorded in the blockchain can be transferred to another user without the help of the owner, and any user can inspect who has the rights at any time through the blockchain. However, it is not desirable for the buyer to resell the seller's data in the data trading model. In [12], Xia et al. proposed blockchain-based data sharing for electronic medical records stored in the cloud. Verifiers can confirm the membership of a user by using cryptographic keys that are generated by the issuer before storing the request to the blockchain. Therefore, all users can efficiently manage their data without the help of a third party. However, the authors do not consider the users' privacy and the limited throughput of the public blockchain.

In the blockchain-based access control system, users' access control policies are open to all participants in the network. This transparency of the blockchain ensures transparent management of the Access Control List (ACL); however, at the same time, it has the disadvantage of making user-defined policies public to all participants in the network. Because the system is affected by security problems within in the blockchain, the relationship between the access control system and the blockchain security concerns must also be considered.

2.2. Blockchain Scalability and State Channel. Blockchain ensures transparency and irreversibility of systems, which have been difficult to achieve for centralized systems. Therefore, many industries are considering converting their operating systems into the blockchain. However, research on the blockchain technology has gradually highlighted unique problems of the blockchain [16, 17], and their evaluation before switching the system to the blockchain is becoming important [18, 19]. Scalability is one of the most representative problems, which means that the speed of transaction processing in the network does not increase even when more resources are put into the blockchain network. This is because the block creation cycle and size are limited for a stable consensus in the blockchain network. Several cryptocurrency developers have attempted to improve transaction throughput in the blockchain network by reducing or eliminating this restriction. However, Croman [20] showed that increasing the block size or decreasing the block generation cycle in the blockchain P2P communication protocol increased the propagation delay in the network [21] and consequently reduced the security of the blockchain network. The blockchain trilemma is the biggest challenge in the blockchain industry due to the difficulty of satisfying all three factors, security, decentralization, and scalability, in the blockchain system. Numerous attempts have been proposed to improve the performance of the blockchain and challenge the trilemma. Currently, there is a private blockchain that is widely used. The private blockchain limits network participants to authorized users and reduces the level of consensus to network administrators to ensure scalability by sacrificing decentralization, thereby failing to solve the trilemma. Hence, the segregated witness (Segwit) [22] of Bitcoin, the sharding and Casper algorithm of Ethereum, and an Algorand's Pure Proof-of-Stake (PPoS) protocol [23] have been proposed as solutions to avoid the blockchain trilemma. Another proposed solution is the state channel, which is the focus in this study.

A state channel has been employed in numerous studies [24, 25] as a solution to solve the problem of scalability due to the finality of the blockchain by introducing off-chain processing methods. Finality guarantees that the block will not change after it is added to the blockchain, which means that the blockchain transaction is irreversible. However, in the public blockchain network, there is a possibility that blocks already added to the blockchain will branch out (i.e., fork) and be discarded due to competitive block generation algorithms. When a fork occurs, groups in the network arise which have two or more different blockchains. After the subsequent block generation process, groups that lost the competition discard their blockchain and replace it with the blocks of the group that won the competition instead. In the process, the transactions in the discarded blockchain are likewise canceled and later included in the block again. Thus, the public blockchain cannot guarantee an absolute finality and only provide a probabilistic one [26]. In contrast, a private blockchain can provide absolute finality by applying noncompetitive consensus algorithms, such as Practical Byzantine Fault Tolerance (PBFT). The probabilistic finality of the public blockchain also affects the transaction throughput of the blockchain network. A private blockchain can achieve higher throughput compared to the public blockchain due to absolute finality. However, in the public blockchain, a certain amount of confirmation time is required after the block is included in the chain to ensure that the block is stochastically safe enough (Bitcoin requires an average of approximately 60 minutes, and Ethereum requires approximately 6 minutes for confirmation). As shown in Table 1, the probability that an attacker can invalidate blocks that have already been confirmed increases with the hash rate that the attacker has in the entire network. However, as the number of confirmed blocks increases, the probability of a successful attack decreases, meaning that the block is highly unlikely to be modified in the presence of sufficient confirmed blocks. Consequently, probabilistic finality makes it difficult to apply the public blockchain to systems that require rapid processing in real time.

A state channel can solve the blockchain trilemma and significantly improve transaction throughput by processing transactions between users on off-chain channels and recording only the results on the blockchain. The transaction processing in the state channel is conducted outside of the blockchain (called the off-chain), such that fast transaction throughput can be guaranteed regardless of the probabilistic finality of the public blockchain. Further, state channels have two advantages: First, transaction processing takes place outside the blockchain, such that transaction processing fees are not required, because the blockchain network does not consume resources. Second, when continuous transactions occur among users, privacy protection may be provided by recording only the first state and final state of transactions in the blockchain instead of all of them, as shown in Figure 1.

The state channel is valid from the time when the initial state of the channel, which all channel participants agreed to, is recorded on the blockchain until one of the channels' various states, which was exchanged on the off-chain, is propagated to the blockchain network by one of the

Table 1: Probability of success of a double-spending attack based on the attacker's hash rate (y-axis) and the number of confirmations (x-axis).

q (%)	1 (%)	2 (%)	3 (%)	4 (%)	5 (%)	6 (%)	7 (%)
2	4	0.237	0.016	0.001	0	0	0
10	20	5.6	1.712	0.546	0.178	0.059	0.02
20	40	20.8	11.584	6.669	3.916	2.331	1.401
30	60	43.2	32.616	25.207	19.762	15.645	12.475
40	80	70.4	63.488	57.958	53.314	49.3	45.769
50	100	100	100	100	100	100	100

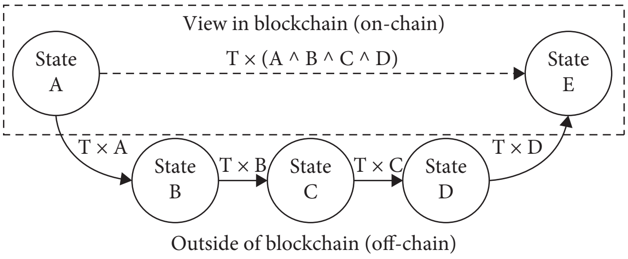


Figure 1: State channel overview.

participants. However, all off-chain states that are generated on the channel are valid states that allow participants in the channel to propagate them to the blockchain network at any time. Thus, participants may propagate the previously agreed past state to the blockchain network as the final state, without the consent of the counterparty and for their own benefit, instead of propagating the final state agreed between the participants.

For example, Alice has a contract to pay Bob a dollar a day for a month. Instead of creating a daily transaction for Bob, Alice can use a state channel to change the balance state of the two participants every day. The initial state of the channel with Alice's balance of \$30 and Bob's balance of \$0 (state_1) will be recorded in the blockchain by Alice. Alice generates a state change transaction $\text{tx}_{1\longrightarrow \text{day}}$ every day which reduces her balance by one dollar and increases Bob's balance by one dollar. Every day, Bob generates and delivers his digital signature to Alice in agreement with the state change transaction that Alice generates. After a month, the off-chain balance state will be \$0 for Alice and \$30 for Bob (state_{30}). Alice or Bob can propagate the last generated state change transaction $\text{tx}_{1\longrightarrow 30}$ to the blockchain network to record it as the final state of the channel and close the channel.

However, on the last day of the contract, Alice could propagate the state change transaction $\text{tx}_{1\longrightarrow 2}$ she created on the first day to the network instead of the last transaction $\text{tx}_{1\longrightarrow 30}$ to avoid paying. As a result, there is no normal transition of state ($\text{state}_1 \longrightarrow \text{state}_{30}$), and only the partial transition of state ($\text{state}_1 \longrightarrow \text{state}_2$) occurs in the blockchain, and the channel is closed.

To prevent the above problem, the use of previous states, except for the most recently agreed state, must be prevented. Decker et al. proposed a method to add a time-lock to the off-chain state, such that it cannot be included in the blockchain until a certain amount of time has passed, even if the previously agreed state is propagated to the network [21]. When generating an off-chain state, participants add a time-lock shorter than the time-lock included in the previous state, such that the most recently agreed off-chain state can be added to the blockchain at any time. However, the interval of the time-lock set on the channel gave rise to the expiration time for the channel to operate. Poon and Dryja proposed a replace-by-revocation [24] that implicitly revokes the previous state and agrees on a new state. In [24], when updating the state of the channel, participants create and exchange transactions that discard the previous state. If one of the participants propagates the channel's previous state (revoked state) to a blockchain network without the counterparty's consent, the counterparty can propagate the previously exchanged revoked transaction to the network within a particular time and eventually consume all deposits that were locked in the channel as a penalty.

3. PyRos System

We propose PyRos, a system that improves the problem of scalability of public blockchain applications. The PyRos system is composed of three layers, as shown in Figure 2.

(i) The **Data Owner (DO)** stores data they want to share with others into the cloud storage. To avoid data exposure by unauthorized users, they must encrypt their data before storing it. DO establishes a state channel to give other users access to these data and manages access to the data based on the off-chain channel's state transition.

(ii) The **Data Requester (DQ)** wants to access DO's data stored in the cloud storage. After obtaining appropriate access rights to DO's data through them, DQ requests the storage keeper to access these data.

(iii) The **Storage Keeper** keeps the stored data securely and provides the requested data only to users with the appropriate permissions. When DQ submits the off-chain state for access to the stored data with the corresponding evidence, they verify the validity of the submitted state and evidence based on the information recorded on-chain.

The first layer is an *application layer*, where the owner of the data and the user requesting access to the data create a state channel to correctly manage data access. In PyRos, the state of the off-chain channel represents the access rights of the channel participants to specific data held by the data owner. The second layer is a *blockchain layer* that records the state of off-chain channels created in the application layer on the blockchain and uses it to validate access authority at the storage layer. The third layer is a *storage layer* that stores data that users want to produce and share with other users. Access to the storage layer is controlled by the storage keeper, who will only provide the requested data to users with appropriate permissions.

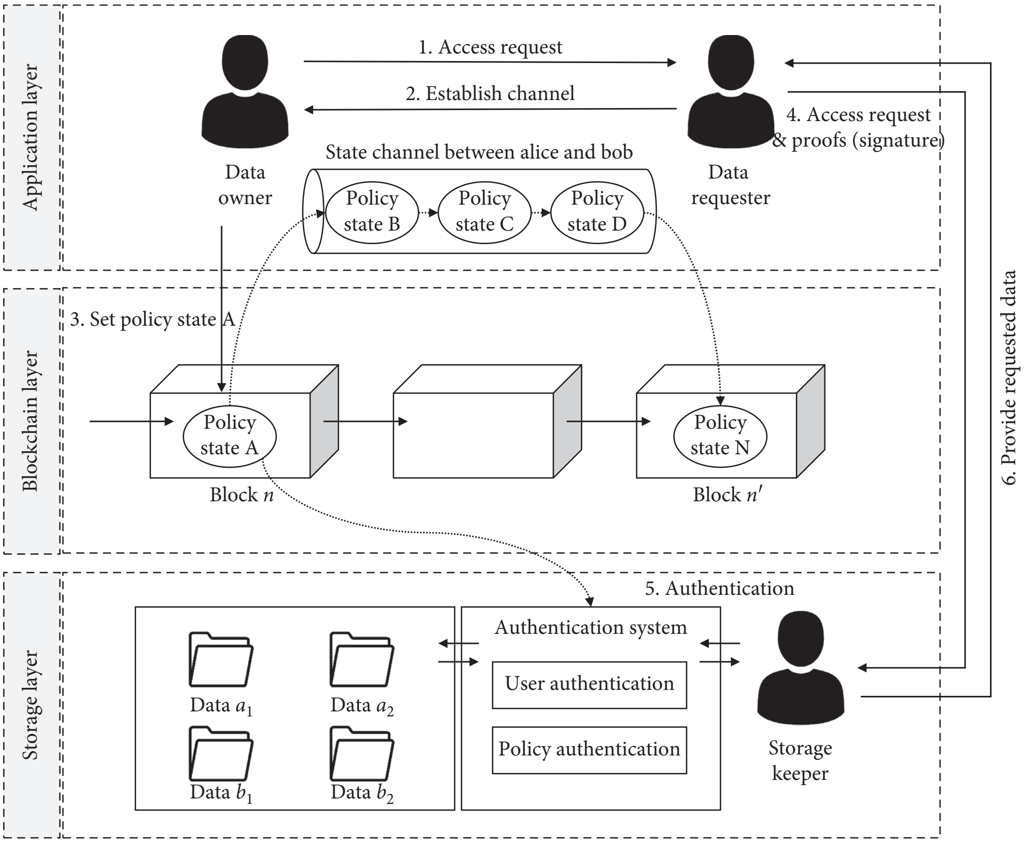


FIGURE 2: Proposed system architecture.

In the remainder of this section, we present the role of participants in the proposed system and the goals that the system seeks to achieve.

3.1. Overview of PyRos. We employ the system architecture shown in Figure 2 to design an access control system over public blockchain networks. PyRos comprises a data owner, data requester, and storage keeper as system participants, and each participant's role in the system is as follows:

The proposed PyRos system consists of *setup*, *channel management*, and *close channel* phases. In the setup phase, the data owner and the data requester create an off-chain channel to control access to the data stored in the external storage by the data owner. In the *channel management* phase, the data owner creates a transaction that changes the state of the off-chain channel created in the setup phase, and the two participants store it individually. In PyRos, the state of the off-chain channel represents the access control policy for specific data stored in the storage layer. The storage keeper validates the data requester's request based on transactions that transform the initial state of the channel stored on-chain into the proper access control policy. Each time a channel's state is changed, the data owner executes an implicit revocation that prevents the data requester from using the channel's previous state in the request. Finally, the *close channel* phase deals with closing the off-chain channel when access control is no longer required between the data owner and the data requester. To perform data access control in PyRos, users create three transactions as follows:

(i) Funding transaction (T_{State_0}): As the first transaction to create an off-chain channel, both users (i.e., DO and DQ) create a funding transaction to deposit their cryptocurrency on the channel. The funding transaction consists of two types of transactions in which users' deposits are transferred to a 2-of-2 multisignature address (the initial state) and in which the channel's deposits are returned to their original owners after a certain time t_{settle} (the refund transaction).

(ii) State transaction (T_{State_n}): By creating state transactions, participants in the channel can change the state of the channel (i.e., redistribution of deposits recorded in the initial state State_0) until the refund transaction recorded on the on-chain is included in the blockchain after t_{settle}. A valid state transaction contains the digital signatures of all participants in the channel and the blockchain addresses of the data owner and requester, such as the standard transaction structure of cryptocurrency (e.g., Bitcoin, Ethereum). However, unlike the standard transaction structure, the signature in the state transaction contains a hash value of the data that users want to share as a message of the signature. Therefore, even if state transactions are propagated over a blockchain network without the consent of the other party, the propagated transaction cannot be included in the blockchain (because it has an invalid structure). After obtaining the valid state transaction, the data requester generates and submits their digital signature to the corresponding storage keeper with the storage transaction generated on their off-chain channel. When the data owner wants to modify the access control policy, they create a new state transaction that changes the hash value contained in the signature to the hash value of the new data without having to establish a new channel.

(iii) Revoked state transaction (RT_{State_n}): Unlike the state transaction, signatures in the revoked state transaction do not contain a hash value of shared data. Consequently, the revoked state transaction can be propagated to the blockchain network and be included in the blockchain, which is used to prevent the data requester from using the channel's out-of-date status in access requests.

4. Security Goals

We consider two threat models for the proposed system, and to design a more realistic and practical system, we adopt several assumptions. First, we assume that a platform exists for the matching of data owners and data requesters. Our proposed system focuses on the sharing of data stored in external repositories which takes place between two users after this matching. Second, we assume that the storage keeper is a trusted entity. The storage keeper honestly verifies the request of the data requester and provides the requested data only to the requester who has presented the valid permissions. Because the focus in this study is the proposal of a decentralized approach control method, centralization of the storage layer is assumed. Finally, we assume that users participating in our system have generated a parent private key/public key pair, with child private key/public key pairs derived from it, and that corresponding addresses are generated from their child public key using BIP 0032 HD Wallets [27]. These child key pairs and addresses are denoted as $Kx = \{sk_{x,1}, pk_{x,1}, addr_{x,1}, \ldots, sk_{x,l}, pk_{x,l}, addr_{x,l}\}$, where x denotes the user's identity and l is the number of indexes.

(i) *Threats within a channel*: Within the established channel, a malicious data requester can attempt to access unauthorized data by modifying the permissions they have been granted from the data owner or by using states that were revoked by the data owner in the past.

(ii) *Threats outside a channel*: Adversaries outside the channel can inspect a blockchain ledger and extract the information needed for an attack from the public information. In the case of an active attacker, an attack on a blockchain network [16, 17] could pose a threat to the security of not only the proposed system but also of all systems operating on the target blockchain network.

Under the threat model noted above, we consider the following security goals for a decentralized access control system on the public blockchain.

(i) *State privacy*: Third parties in the public blockchain network (except the data owner, data requester, and storage keeper) must not know details of the access control. According to the *need-to-know* principle, user access controls and authorization procedures and its objective is to ensure that only authorized individuals gain access to information or systems necessary to undertake their duties.

(ii) *Scalability*: The reliability of the permission to access objects in the proposed system is based on the features of the public blockchain (i.e., decentralization, transparency, and irreversibility). However, the problem of the public blockchain scalability is a major constraint on these features contributing to the proposed system. Hence, the creation, modification, and disposal of access control policies must be done quickly, regardless of the network performance of the blockchain, even if the system operates on the public blockchain that offers only limited scalability.

(iii) *Revocation*: The data owner and data requester perform access control through the state transition of their off-chain channel state. The data owner manages access control policies by generating transactions that cause the off-chain channel's state transition ($state_1 \longrightarrow state_n$). Until the channel is closed, the data owner creates transactions that can change the off-chain channel's state and shares it with the data requester. The transaction is not propagated to the blockchain network, and it is kept personally by two participants in the off-chain before being used in the authentication process when the data requester requests access to the storage keeper. However, because transactions are shared only between the two participants (off-chain), the data requester may present transactions for change to the channel's past state ($state_1 \longrightarrow state_{n-k}$) for other purposes instead of transactions for the transition to the channel's current state ($state_1 \longrightarrow state_n$). To avoid this problem, the data owner must have a measure that prevents the previous state of the channel from being used by the requester in the data access process.

4.1. Phase 1: Setup. Both parties individually create the funding transaction of the same structure that transfers their funds (predefined amounts in negotiation) to a single 2-of-2 multisignature address as a deposit (except for the counterparty's digital signature). To prevent unauthorized modification of the transaction due to the order of the exchange of signatures [24], both parties do not exchange their signature until they have individually created a refund transaction. Both parties execute the following steps, as shown in Figure 3:

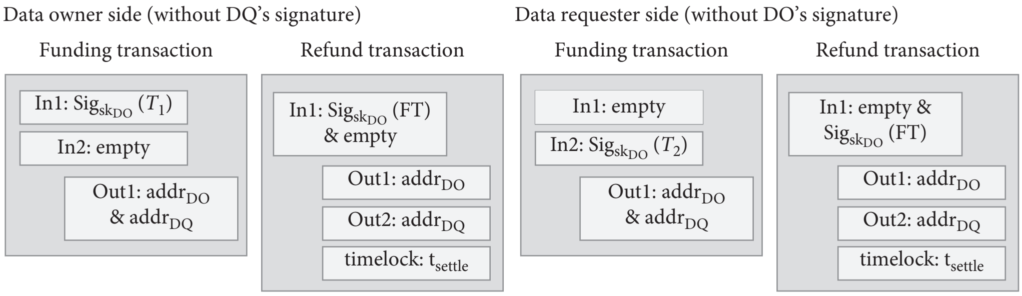

Data owner side (without DQ's signature)
Funding transaction
In1: $Sig_{sk_{DO}}$ (T_1)
In2: empty
Out1: $addr_{DO}$ & $addr_{DQ}$
Refund transaction
In1: $Sig_{sk_{DO}}$ (FT) & empty
Out1: $addr_{DO}$
Out2: $addr_{DQ}$
timelock: t_{settle}
Data requester side (without DO's signature)
Funding transaction
In1: empty
In2: $Sig_{sk_{DO}}$ (T_2)
Out1: $addr_{DO}$ & $addr_{DQ}$
Refund transaction
In1: empty & $Sig_{sk_{DO}}$ (FT)
Out1: $addr_{DO}$
Out2: $addr_{DQ}$
timelock: t_{settle}

FIGURE 3: Setup phase transaction structure.

(1) DQ provides their first address addrDQ,1 derived from their first private key and public key pair <$sk_{DQ,1}$, $pk_{DQ,1}$> to DO
(2) DO and DQ generate the funding transaction FT, which sends their deposits to the channel, and the refund transactions, which return the deposits after t_{settle}
(3) Exchange each other's FT and refund transaction
(4) Add their signature to the incomplete transaction that has been received
(5) Finally, DO and DQ establish their off-chain channel by propagating completed transactions to the blockchain network

4.2. Phase 2: Channel Management. After FT is finalized in the blockchain (i.e., the block depth including FT is six or higher), DO and DQ create a first state transaction off-chain to indicate the new access control policy. In this phase, they create transactions with different structures, as shown in Figure 4. The state transaction redistributes the deposit locked in the off-chain channel to DO and DQ. As described earlier, when a platform exists for matching DO and DQ, we assume that DQ already knows the hash value of the data h_{data}.

(1) DQ sends the hash value of the randomly selected value h_{r_1} to DO
(2) DO and DQ create a state transaction T_{state_1}, as shown in Figure 4 (where signatures contain the hash value of the data $h_{data,1}$ as a digest message)
(3) DO and DQ attach their digital signatures to the state transaction and exchange it with each other
(4) DO and DQ complete the state transaction by adding their digital signature to the incomplete transaction that has been received

DQ requests data from the storage keeper by presenting proofs for user authentication and state transaction T_{state_1}. The storage keeper validates the access request based on proofs presented by DQ and the information shown on the blockchain ledger (Algorithm 1).

(1) DQ sends a request message $m = \{h_{data1}, \text{DO}, addr_{DO}, addr_{DQ}, \text{FT}, T_{state,1}, r_1\}$ with their digital signature Sig_{skDQ} (m)
(2) The storage keeper uses the *stateValidate* algorithm to validate the access request. The *stateValidate* algorithm verifies whether the request meets the following:
   (A) The validity of the off-chain channel
   (B) Whether the signature contained in $T_{state,1}$ can be verified with the address contained in FT
   (C) Whether the signature presented by DQ can be verified using the address included in $T_{state,1}$
   (D) Whether the signature contained in $T_{state,1}$ can be verified using the hash operation results for r_1 presented by DQ as a digest message

A storage keeper can verify whether the message digest of the signatures in the state transaction contains the data requested by the DQ to determine the right of access to the object. However, if DO creates a new state transaction that includes signatures for new data to modify the access control policy of DQ, it cannot guarantee that the DQ does not use the state transaction created in the past. Therefore, we applied the replace-by-revocation used in [24] to PyRos, such that if the DQ used a ticket that had been revoked in the past to access an object whose access rights had been revoked, they would lose the amount deposited on the channel, as shown in Figure 5 and described as follows:

(1) DQ sends the hash value of the randomly selected value h_{r_2} to DO with their new address $addr_{DQ,2}$.
(2) DO and DQ create a new state transaction T_{state_2}.
(3) DQ creates a revoked state transaction RT_{state_1}, which has the same structure as the state transaction T_{state_1}; however, it does not contain hash values of shared data $h_{data,1}$ in the signature message digest.
(4) DO and DQ add their digital signature to the new state transaction and exchange it with each other. Additionally, only DQ performs the same process for $RT_{state,1}$ and generates a signature to claim ownership of their deposits in the $RT_{state,1}$, after which they send it to the DO.

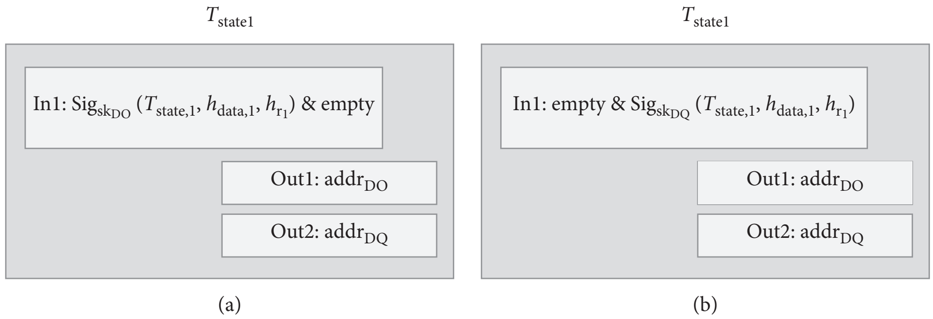

T_{state1}
In1: $\mathrm{Sig}_{sk_{DO}}$ ($T_{state,1}$, $h_{data,1}$, h_{r_1}) & empty
Out1: addr_{DO}
Out2: addr_{DQ}
(a)
T_{state1}
In1: empty & $\mathrm{Sig}_{sk_{DQ}}$ ($T_{state,1}$, $h_{data,1}$, h_{r_1})
Out1: addr_{DO}
Out2: addr_{DQ}
(b)

Figure 4: Channel management transaction structure (grant access). (a) Data owner side (without DQ's signature). (b) Data requester side (without DO's signature).

```
Procedure Channel_Validate
        Search (FT) in Blockchain network
        if FT.output was spent then return False
        else return True
end procedure
Procedure Verify_Signature in $T_{state}$
        If Verify$T_{state}$.input.Sig$_{skx}$, FT.output.addr$_x$ is True then return Ture
        else return False
end procedure
Procedure Verify_Signature in $m$
        If Verify (Sig$_{Skx}$ ($m$), $T_{state}$.output.addr$_x$) is True then return Ture
        else return False
end procedure
Procedure Verify_Hash ($m$)
        HashValue ← Hash ($m.r$)
        $M$ ← Hash ($T_{state}||h_{data}||$HashValue)
        If Verify ($T_{state}$.input.Sig$_{skx}$ ($M$), addr$_x$ is True then return Ture
        else return False
end procedure
```

Algorithm 1: *StateValidate*.

(5) DO and DQ complete received transactions by adding their digital signature to the incomplete transaction that has been received.

4.3. Phase 3: Close Channel. In the proposed system, we consider closing the channel in the following cases:

(A) When there is no further transaction between the data owner and the data requester, they create a closing transaction and propagate it to the blockchain network to apply the channel's final state to the blockchain. If the channel's final state is propagated to the blockchain network, all state transactions previously created on the channel are automatically invalid and return the deposit locked in the setup phase.

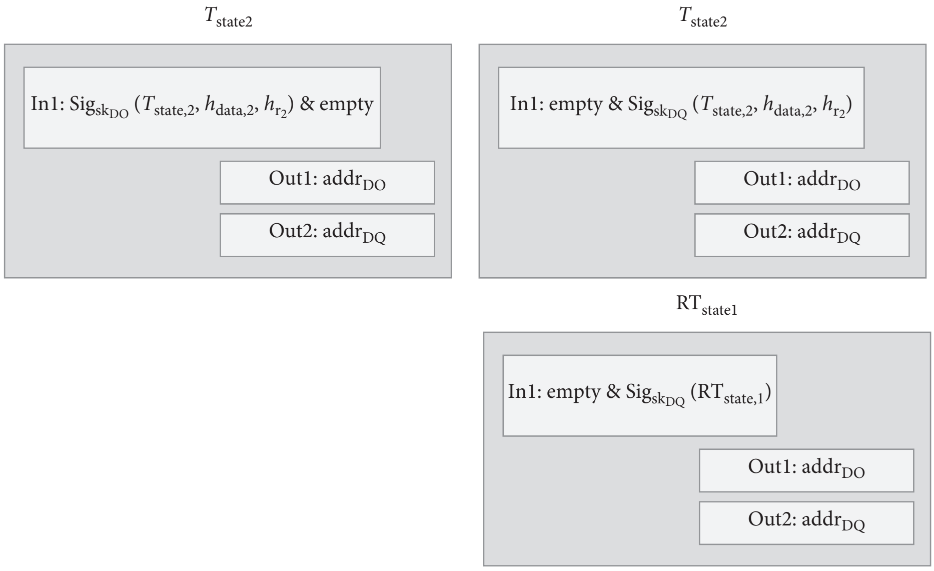
(B) In the case of a nonresponsive counterparty, the deposit in the channel can be returned to participants by automatically closing the channel as the refund transaction that had a time-lock t_{settle} is included in the blockchain after a time t_{settle}.

(C) If the use of the previously revoked state is detected during the verification process, the honest storage keeper will inform the use of the revoked state T_{state_x} that was received from DQ to DO. After the use of the revoked state has been confirmed, DO propagates the revoked state transaction RT_{state_x} to the blockchain network and transfers DQ's deposits in RT_{state_x} to their account using DQ's received signature in the modify permission process.

The state transition of the channel from phase 1 to phase 3 is shown in Figure 6. Figure 6 illustrates a scenario in which the access control policy was updated three times, and the channel was closed normally.

5. Security Analysis

5.1. State Privacy. Our goal is to protect users' privacy by preventing third parties that do not participate in access control for a particular user in PyRos system (i.e., except for the data owner, data requester, and storage keeper) from knowing the content of the transactions. The goal of state privacy is to protect the user's transaction information against the adversaries that monitor the blockchain network. First, the initial state of the channel, in which the participants transfer their deposits to the channel, and the refund


T_{state2}
In1: $\text{Sig}_{\text{sk}_{\text{DO}}}$ $(T_{\text{state,2}}, h_{\text{data,2}}, h_{r_2})$ & empty
Out1: addr_{DO}
Out2: addr_{DQ}
T_{state2}
In1: empty & $\text{Sig}_{\text{sk}_{\text{DQ}}}$ $(T_{\text{state,2}}, h_{\text{data,2}}, h_{r_2})$
Out1: addr_{DO}
Out2: addr_{DQ}
$\text{RT}_{\text{state1}}$
In1: empty & $\text{Sig}_{\text{sk}_{\text{DQ}}}$ $(\text{RT}_{\text{state,1}})$
Out1: addr_{DO}
Out2: addr_{DQ}

(a) (b)

Figure 5: Channel management transaction structure (modify permissions). (a) Data owner side (without DQ's signature). (b) Data requester side (without DO's signature).

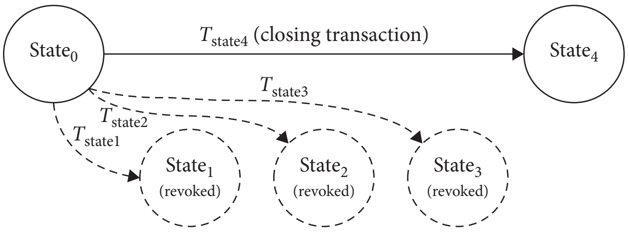

State_0
T_{state4} (closing transaction)
State_4
T_{state3}
T_{state2}
T_{state1}
State_1 (revoked)
State_2 (revoked)
State_3 (revoked)

Figure 6: State transition of off-chain channel.

transaction, in which the channel-locked deposits are returned to the participants, must be known in the blockchain network. As shown in Figure 3, the only information disclosed in funding transactions for channel establishment is the address, deposit of the participants who generate the channel, and their signature. If the participants close the established channel, the redistribution of deposits by the participants will be recorded in the blockchain. In the event of fraudulent use of the state, the channel will record the state transaction that has been revoked in the past by the honest participant and transfer the deposits of malicious participants locked in the revoked state to the other user. However, transactions that contain information about the shared object are not recorded in the blockchain, and all off-chain transactions are communicated only through the personal communication channel between participants. Consequently, it is highly unlikely that adversaries obtain significant transaction information only from transactions recorded in the blockchain. Moreover, because all transactions recorded in the blockchain in PyRos system follow the standard structure of cryptocurrencies, the adversaries cannot distinguish between transactions for payment and PyRos transactions. Even if the adversary chooses to target and monitor all of their packets, it will be difficult to find significant transactions, because the data requester will change their addresses used for each creation of the state transaction.

5.2. Scalability. The purpose behind solving the scalability problem is to ensure that the system will operate without delay, regardless of the throughput of the blockchain network, while assuming that the adversary can attempt various known attacks [16, 27] on the network. In the previous section, we assumed that access control through channels is performed after the FT establishing channels has been finalized in the blockchain. Therefore, the block containing FT cannot be modified in the blockchain after FT has been finalized with sufficient confirmation. In the proposed system, all transactions (except FT) need not be propagated to the blockchain network until the channel is closed. The implementation of existing centralized systems in the decentralized network required a majority of the network consensus instead of trust institutions; however, this resulted in a large transaction processing delay. Attacks on blockchain networks took advantage of these delays to achieve malicious purposes, such as double payments. However, as described in Section 2, a state channel that only requires agreement from the channel participants is free from this delay and can be operated regardless of the availability of the blockchain network, if the integrity of the initial state, which is the basis of the channel's reliability, is guaranteed.

5.3. Revocation. In the PyRos system, the data owner's access control policies can be expressed in the off-chain state of the channel. However, as described in Section 2, the off-chain state of the channel is not recorded in the blockchain, such that explicit revocation of the past state is practically impossible. Therefore, we employed the implicit revocation used in [24] to require the data requester to pay the penalty for fraudulent use, although they may use a state transaction that was revoked in the past. In modifying the permission phase, the data requester creates a revoked state transaction RT_{state} in the form of the standard transaction excluding shared data information in the signature message digest. Further, the data requester generates a digital signature that enables the use of their deposit transferred to the revoked state transaction and sends it to the data owner with the latter. If the data requester attempts to access data using a revoked state, the data owner can propagate RT_{state} to the blockchain network to close the channel and use the data requester's deposit as a penalty.

However, we assumed that all system participants, except the storage keeper, could be malicious. Therefore, a malicious data owner may propagate a revoked state transaction to the blockchain network, regardless of whether the state is used fraudulently or not. To prevent this problem, we have added new conditions for consuming the data requester's deposit in the revoked state transaction. The standard transaction structure records the address of the new owner in the blockchain for the amount used in the transaction. The new owner then attempts to use cryptocurrency by attaching a digital signature, which is generated by the key corresponding to the address recorded in the blockchain, to the new transaction. Only if this digital signature is valid will the transaction be recorded in the blockchain. However, we added the hash value h_{r_k} of the r_k selected randomly by the data requester to the condition for the consumption of the data requester's deposit in the revoked state transaction. Hence, the data owner requires a preimage of the hash result r_k, included in the revoked state transaction RT_{state} with the digital signature of the data requester to consume the data requester's deposit.

6. Evaluation

We evaluate this proposal through comparison with other studies. PyRos implements blockchain-based access control using the off-chain state channel. It exhibits a major characteristic of improving performance by applying off-chain computation processing based on the state channel. The evaluation focused on the delay required for access control.

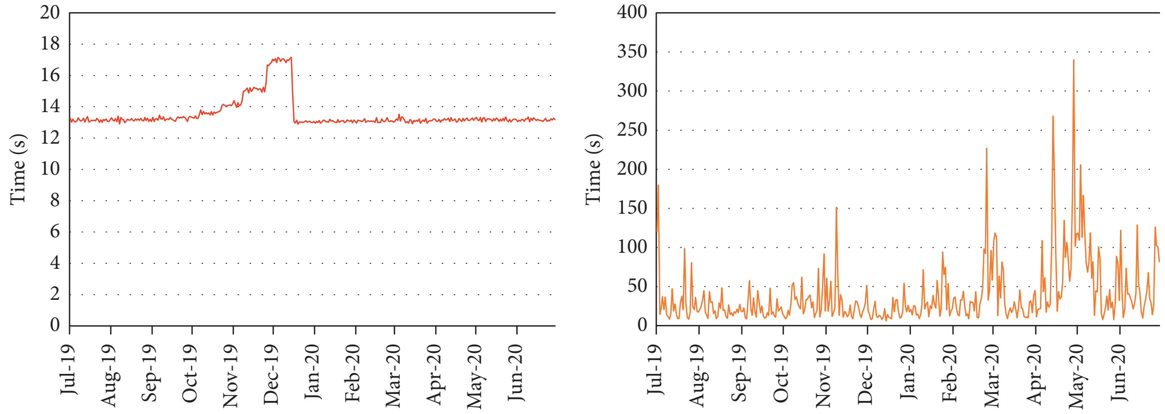
In the related studies on blockchain-based access control [10–12] mentioned in Section 2, a method of recording data related to access control was employed in the irreversible blockchain database. Blockchain technology provided users with key features in access control without the participation of centralized managers, which enabled the implementation of decentralized access control. However, considering a realistic data society environment, when an access control application is implemented in the public blockchain, its problem of scalability has become a major constraint. Access control applications can be implemented on private blockchain networks to solve performance problems. However, if the access control application is implemented on a private blockchain, the presence of the blockchain network administrator will not guarantee decentralization of access control and will not be able to implement dynamic access control services due to a limited pool of network participants. Nonreversive and decentralized databases are highly efficient for the storage and verification of access control policies. However, the probabilistic finality of the public blockchain will require minimal time for the irreversibility of stored access control policies to reach a secure level. Table 2 lists features associated with block generation of known public blockchain platforms.

The public blockchain platform uses a consensus protocol to maintain and manage a single and unique database without the trusted third party among unreliable network members. A consensus protocol ensures that the blockchain network operates even if there is no more than a certain percentage of malicious users (byzantine node) or users who cannot participate in the protocol (fault node) in the network. This algorithm contributes to maintaining a highly secure blockchain network without the trusted third party; however, it causes delays in the database's update process. As shown in Table 1, all public blockchain platforms limit the average block generation cycle through a consensus protocol. Most public blockchain platforms have a limited number of transactions that can be processed per unit time (known as TPS). Considering that this consensus process is required in all processes on the public blockchain application, the recording and updating of access control policies will consistently have the minimum delay required to create blocks on the blockchain platform. Figure 7 shows the average time it takes for blocks to be included in the Ethereum and Bitcoin blockchains. Ethereum takes an average of 14 s, and Bitcoin takes about a minute to connect a block. Hence, new data will not be quickly reflected in the blockchain if transactions that newly register or renew access control policies are excessively concentrated at a specific time, which could have a fatal impact on the availability of access control applications. In contrast, in PyRos, access control policies are represented as an off-chain state, such that the on-chain consensus process is not necessary. The availability of existing proposals depends on the performance of the blockchain network, onto which the application is implemented, whereas in PyRos, the performance of the network does not affect the availability of the access control application at all, except for the setup phase.

Table 3 shows the results of comparing PyRos and other researches [10–12] from a performance perspective. In the blockchain application, the biggest impact on performance is the network topology and consensus mechanism [28]. The public blockchain network enables secure management of access control policies. However, it takes a lot of time before requests for the state transfer in the blockchain state DB are reflected in the majority of network nodes. As shown in Figure 7, the time taken in this process changes to flexible depending on various factors such as the size of the blockchain network and consensus algorithm. However, regardless of the blockchain platform, this time commonly refers to the process of transactions being contained in the block by miners after they are propagated/verified to nodes in the network. References [10–12] are commonly based on the Bitcoin blockchain. Thus, all transactions associated with

Table 2: Information related to block generation of public blockchain platforms.

	Bitcoin	Ethereum	Ripple	Monero
Blockchain type	Permisionless	Permisionless	Permissioned	Permisionless
Block size (average)	1 MB	Variable (1,500,000 gas limit, averages in ∼20–30 KB)	N/A	Variable (twice the median size of the last 100 blocks; the limit is 60 KB)
Block cycle (average)	10 min	10–19 s	N/A	2 min
Consensus algorithm	Proof-of-Work	Proof-of-Work	Ripple Protocol Consensus Algorithm (RPCA)	Proof-of-Work



(a)

(b)

Figure 7: Historical average time required for a block to be included in the Ethereum blockchain (a) and the Bitcoin blockchain (b).

Table 3: Comparison from a performance perspective.

	Ouaddah et al. [10]	Maesa et al. [11]	Xia et al. [12]	PyRos
Transaction/block validation	Majority of full nodes	Majority of full nodes	Majority of full nodes	Channel participant (after the channel is created)
Transaction/block propagation	Majority of full nodes	Majority of full nodes	Majority of full nodes	Channel participant (after the channel is created)
Block mining (consensus)	Required	Required	Required	Not required (after the channel is created)
Scalability	No	No	Customized block structure	State channel

the access control protocol must be propagated and verified by a majority of full nodes in the network. Reference [12] applied a method to reduce the size of block data propagated in the blockchain network through a customized block structure. However, they still had limitations in the process of the propagation and validation of the transaction. In contrast, PyRos can save significant processing time by omitting the propagation and validation process using a state channel. Instead of transferring the blockchain state DB, channel participants' requests that occur after the channel is created transfer only the state of the channel which is shared only among channel participants. Therefore, all access control events that occur in PyRos can be processed quickly without delay due to network processing.

7. Use Case and Future Studies

The proposed system can be applied to a variety of fields; however, we expect its particularly widespread use in the healthcare sector. As an example, we assume a scenario in which sellers and buyers promise periodic data transactions over a period of time, rather than simple data transactions that occur only once. A patient suffering from diabetes and a company studying diabetes drugs may sign a contract, in which the company receives health data from the patient once a week. Patients provide their health data to companies every week, and companies pay cryptocurrency, such as Bitcoin, in return. In this process, the data seller encrypts their data and keeps it in external storage. After the contract is signed with the data requester, the data seller creates channels on the blockchain with the requester instead of transmitting the data directly. Subsequently, the seller periodically grants an access right to the new data and the ability to decode it, and at the same time, the requester pays the seller cryptocurrency for the data.

However, we assume that the storage keeper is a semitrusted entity that is expected to act honestly upon legitimate requests. The storage keeper consistently provides the requested data after user authentication; however, this is an assumption that violates the decentralization aspect within the system's purpose. To solve this problem, we aim to attempt the implementation of a decentralized storage layer in a future study. P2P storage, such as the interplanetary file system (IPFS), serves as a good platform for this study, and we plan to conduct research that will assume control of access to encrypted data stored in distributed repositories across the blockchain.

8. Conclusions

We proposed PyRos, a system that supports data trading and sharing between individuals on top of the public blockchain. The public blockchain is more reliable than the private blockchain, as it is increasingly difficult for more users to manage the blockchain and attackers to attack all blocks. However, the scalability problem in the public blockchain network makes it difficult to quickly synchronize blockchain databases. Therefore, we proposed a system that supports the data sharing application between individuals by combining access control service based on the off-chain state channel on the public blockchain. In PyRos, the user's access control policy is represented by the state of the off-chain channel. The state of the off-chain channel can be changed by the agreement of the channel's participants, which can greatly reduce the costs required for agreement compared to the on-chain. Moreover, this approach is easy to implement in existing systems and does not require the addition of any new elements. We hope that this proposed system will contribute as a step toward a user-centric data society.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2020-2015-0-00403), supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation), and partially supported by Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Education (no. 2019R1I1A3A01060652).

References

[1] D. Reinsel, J. Gantz, and J. Rydning, *The Digitization of the World from Edge to Core*, IDC White Paper, 2018.

[2] M. Viceconti, P. Hunter, and R. Hose, "Big data, big knowledge: big data for personalized healthcare," *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 4, pp. 1209–1215, 2015.

[3] G. Wang, A. Gunasekaran, E. W. T. Ngai, and T. Papadopoulos, "Big data analytics in logistics and supply chain management: certain investigations for research and applications," *International Journal of Production Economics*, vol. 176, pp. 98–110, 2016.

[4] "Sources of big data: where does it come from?," 2020, https://www.cloudmoyo.com/blog/data-architecture/what-is-big-data-and-where-it-comes-from/.

[5] D. Rushe, *Facebook Sorry–Almost–for Secret Psychological Experiment on Users*, The Guardian, 2014.

[6] C. Cadwalladr and E. Mraham-Harrison, *Revealed: 50 Million Facebook Profiles Harvested for Cambridge Analytica in Major Data Breach*, The Guardian, 2018.

[7] *About–MyData.org*, https://mydata.org/about/, 2020.

[8] *Ocean Protocol—A Decentralized Data Exchange Protocol to Unlock Data for AI*, The Ocean Protocol Whitepaper, 2018, https://oceanprotocol.com.

[9] M. Anastasiu, S. Giacomelli, D. Hanson, C. Pennachin, and M. Argentieri, *SingularityNET: A Decentralized, Open Market and Inter-network for AIs*, The SingularityNET Whitepaper, 2020, https://public.singularitynet.io/whitepaper.pdf.

[10] A. Ouaddah, A. A. Elkalam, and A. A. Ouahman, "Towards a novel privacy-preserving access control model based on blockchain technology in IoT," in *Europe and MENA Cooperation Advances in Information and Communication Technologies*Springer, Berlin, Germany, 2017.

[11] D. D. F. Maesa, P. Mori, and L. Ricci, "Blockchain based access control," *IFIP International Conference on Distributed Applications and Interoperable Systems*, Springer, Berlin, Germany, 2017.

[12] Q. Xia, E. Sifah, A. Smahi, S. Amofa, and X. Zhang, "BBDS: blockchain-based data sharing for electronic medical records in cloud environments," *Information*, vol. 8, no. 2, p. 44, 2017.

[13] K. Wüst and A. Gervais, "Do you need a blockchain?" in *Proceedings of the Crypto Valley Conference on Blockchain Technology (CVCBT)*, Zug, Switzerland, 2018.

[14] K. Qin and A. Gervais, *An Overview of Blockchain Scalability, Interoperability and Sustainability*, Hochschule Luzern Imperial College London Liquidity Network, London, UK, 2018.

[15] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008, http://bitcoin.org/bitcoin.pdf.

[16] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on bitcoin's peer-to-peer network," in *Proceedings of the USENIX Security Symposium*, pp. 129–144, Washington, DC, USA, 2015.

[17] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *proceedings of the International conference on financial cryptography and data security*, pp. 436–454, Christ Church, Barbados, March 2014.

[18] *Evaluation Forms for Blockchain-Based System Ver 1.0*, Ministry of Economy, Trade and Industry in Japan, 2020, http://www.meti.go.jp/press/2016/03/20170329004/20170329004.html.

[19] G. Fridgen, F. Guggenmoos, J. Lockl, A. Rieger, and A. Schweizer, "Developing an evaluation framework for blockchain in the public sector: the example of the German Asylum process," in *Proceedings of the 1st ERCIM Blockchain Workshop*, Amsterdam, Netherlands, July 2018.

[20] K. Croman, "On scaling decentralized blockchains," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, pp. 106–125, Christ Church, Barbados, February 2016.

[21] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *Proceedings of the IEEE Thirteenth International Conference on Peer-To-Peer Computing (P2P)*, pp. 1–10, Trento, Italy, 2013.

[22] E. Lombrozo, J. Lau, and P. Wuille, *BIP 0141: Segregated Witness (Consensus Layer)*, 2015, https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki.

[23] J. Chen and S. Micali, "Algorand," 2017, http://arxiv.org/abs/1607.01341.

[24] J. Poon and T. Dryja, "The bitcoin lightning network: scalable off-chain instant payments," *Medium*, vol. 9, p. 14, 2016 (Draft Version) 0.5.

[25] A. Miller, I. Bentov, R. Kumaresan, C. Cordi, and P. McCorry, "Sprites and state channels: payment networks that go faster than lightning," 2020, http://arxiv.org/abs/1702.05812.

[26] A. Gauba, "Finality in Blockchain Consensus," *Medium*, https://medium.com/mechanism-labs/finality-in-blockchain-consensus-d1f83c120a9a, 2018.

[27] P. Wuille, *BIP 0032: Hierarchical Deterministic Wallets*, 2012, https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki.

[28] P. W. Eklund and R. Beck, "Factors that impact blockchain scalability," in *Proceedings of the 11th International Conference on Management of Digital EcoSystems*, pp. 126–133, Limassol, Cyprus, November 2019.