sensors

MDPI

Article

Combining IOTA and Attribute-Based Encryption for Access Control in the Internet of Things †

Yuanyu Zhang [1,2,*], Ruka Nakanishi [2], Masahiro Sasabe [2] and Shoji Kasahara [2]

[1] School of Computer Science and Technology, Xidian University, Xi'an 710071, China
[2] Graduate School of Science and Technology, Nara Institute of Science and Technology, 8916-5 Takayama-Cho, Ikoma, Nara 630-0192, Japan; nakanishi.ruka.nm0@is.naist.jp (R.N.); m-sasabe@ieee.org (M.S.) kasahara@is.naist.jp (S.K.)
* Correspondence: yy90zhang@gmail.com or yyzhang@is.naist.jp

† This paper is an extended version of our paper published in IOTA-based access control framework for the Internet of Things. In Proceedings of the 2nd Conference on Blockchain Research and Applications for Innovative Networks and Services (BRAINS 2020), Paris, France, 28–30 September 2020.

Abstract: Unauthorized resource access represents a typical security threat in the Internet of Things (IoT), while distributed ledger technologies (e.g., blockchain and IOTA) hold great promise to address this threat. Although blockchain-based IoT access control schemes have been the most popular ones, they suffer from several significant limitations, such as high monetary cost and low throughput of processing access requests. To overcome these limitations, this paper proposes a novel IoT access control scheme by combining the fee-less IOTA technology and the Ciphertext-Policy Attribute-Based Encryption (CP-ABE) technology. To control the access to a resource, a token, which records access permissions to this resource, is encrypted by the CP-ABE technology and uploaded to the IOTA Tangle (i.e., the underlying database of IOTA). Any user can fetch the encrypted token from the Tangle, while only those who can decrypt this token are authorized to access the resource. In this way, the proposed scheme enables not only distributed, fee-less and scalable access control thanks to the IOTA but also fine-grained attribute-based access control thanks to the CP-ABE. We show the feasibility of our scheme by implementing a proof-of-concept prototype system using smart phones (Google Pixel 3XL) and a commercial IoT gateway (NEC EGW001). We also evaluate the performance of the proposed scheme in terms of access request processing throughput. The experimental results show that our scheme enables object owners to authorize access rights to a large number of subjects in a much (about 5 times) shorter time than the existing access control scheme called Decentralized Capability-based Access Control framework using IOTA (DCACI), significantly improving the access request processing throughput.

Keywords: Internet of Things; access control; IOTA; Ciphertext-Policy Attribute-Based Encryption (CP-ABE); blockchain

check for updates

Citation: Zhang, Y.; Nakanishi, R.; Sasabe, M.; Kasahara, S. Combining IOTA and Attribute-Based Encryption for Access Control in the Internet of Things. *Sensors* **2021**, *21*, 5053. https://doi.org/10.3390/s21155053

Academic Editors: Hong-Ning Dai, Jiajing Wu and Hao Wang

Received: 27 June 2021
Accepted: 23 July 2021
Published: 26 July 2021

1. Introduction

Thanks to the rapid advancement of the Internet of Things (IoT), an unprecedented number of devices are connected to the Internet today, from sensors to home appliances, as well as automobiles [1,2]. The tremendous amount of data these devices collect from the physical world has opened up new opportunities for smart applications in various fields, including logistics, healthcare and manufacturing [3–5]. On the other hand, IoT devices are vulnerable to security attacks represented by unauthorized access [6–9], because IoT devices, especially sensors and actuators, are usually not equipped with strong security measures due to their limited computing power. Since IoT devices often handle personal information and play an important role in various task executions, unauthorized access to them can result in serious issues such as information leakage and malfunction, greatly threatening our safety and privacy. Therefore, enforcing appropriate *access control* is essential for the security of IoT systems.

Access control is the process of ensuring that only the authorized users (i.e., subjects) can access a resource (i.e., object), e.g., device and data. To enforce access control, the access rights of subjects, i.e., information about which subject can access which resource, have to be recorded in a certain format and referred to when the subjects access the resource. In conventional access control systems, such access right information is usually stored in a centralized server for ease of management. However, this leads to various limitations, such as a single point of failure, performance bottleneck and vulnerability against tampering by malicious users, especially in large-scale environments such as the IoT.

To achieve reliable and distributed access control, various access control schemes have been proposed recently based on the blockchain technology [10–27]. Blockchain is the underlying technology of cryptocurrency systems such as Bitcoin [28] and Ethereum [29], which is a distributed database that is managed over Peer-to-Peer (P2P) networks. Peers sync, verify and reach consensus on the data to be recorded to the blockchain, making the database tamper resistant. In addition, the data are stored in a distributed manner because every peer keeps its own copy of the entire blockchain. Owing to these attractive properties, blockchains have been regarded as suitable platforms for storing access rights. Another promising feature of the blockchain technology is smart contract, which is a piece of executable codes stored in the blockchain. Smart contracts transform the blockchain system from a mere database to a distributed and trustworthy computing platform [30], making them appropriate for the computation tasks in access control such as managing access rights and processing access requests. This has given rise to various smart-contract-based access control schemes for the IoT [10–23].

Although blockchain-based access control schemes have overcome the limitations in conventional access control schemes, they have also given rise to two main drawbacks deriving from the underlying blockchains. First, they incur monetary cost to users, since users need to pay some fee to the peers who manage and update the blockchain [31], i.e., verify the validity of the access rights/policies stored on the blockchain and execute smart contracts for processing access requests. Second, they suffer from low throughput of access request processing. In blockchain, newly incoming data are never considered valid unless they are verified and appended to the blockchain. Although new blocks are added to the blockchain at regular intervals, each block can store only a limited amount of data, restricting the capability of processing access requests [32]. These two shortcomings have to be addressed in order to adapt to large-scale IoT systems.

To address the limitations in blockchain-based schemes, researchers began to seek alternative distributed ledger technologies (DLTs) for IoT access control. One of them is IOTA, which is a next-generation DLT designed specifically for the IoT. IOTA aims to overcome the drawbacks of blockchains, i.e., low throughput and high transaction fee, by adopting a different data structure for the ledger and changing the consensus mechanism. These features will be introduced in greater detail in Section 3. Although IOTA is criticized for its current centralized-like implementation due to the introduction of coordinators, such an issue will be addressed completely in the upcoming IOTA 2.0, which is a fully decentralized version, as announced by the IOTA foundation [33].

The fee-less and high-throughput features of IOTA have made it highly promising for implementing access control for IoT systems. This is why the IOTA foundation has launched the IOTA Access project in September 2020 with Jaguar Land Rover, STMicroelectronics, EDAG, RIDDLE & CODE, NTT DATA Romania, ETO GRUPPE and BiiLabs [34]. IOTA Access aims to build an open-source DLT framework for customizing policy-based access control that is able to manage billions of IoT devices/machines. Unfortunately, IOTA Access is still in progress currently. An access control scheme called the Decentralized Capability-based Access Control framework using IOTA (DCACI) has been proposed in [35]. In the scheme, the distributed ledger of IOTA, called the Tangle, is used to store the subjects' access rights in the form of tokens. This enables the access rights to be stored in a distributed and tamper-resistant fashion, similar to blockchain-based schemes. In addition to this, DCACI achieves fee-less access control with high throughput thanks to IOTA. However, the DCACI scheme faces some limitations in terms of scalability and security, such as heavy token management and the lack of security considerations, which will also be described in detail in Section 3.

The main contribution of this paper is to propose a fine-grained, fee-less and scalable IoT access control that supports easy token management. This is achieved by combining the IOTA technology and Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [36]. More specifically, we encrypt access tokens using CP-ABE and distribute them to the subjects through the IOTA Tangle. More flexible and fine-grained access control can be enforced by finely specifying the policies in CP-ABE, which are rules describing subjects with what attributes can access objects with what attributes under what conditions. We also show the applicability of our access control scheme in commercial smart phones (Google Pixel 3 XL) and IoT gateways (NEC EGW001), and evaluate the performance of the proposed scheme in terms of access request processing throughput. Compared with DCACI, the proposed scheme provides stronger security, such as secure communication between the subject and object owner when sending access requests as well as the authentication of subjects when using their tokens. The concrete methods to implement such security measures are also shown, whereas in [35] the security measures are not provided. The proposed scheme provides easier token management than the DCACI by introducing one-to-many access control, i.e., one token can be used for multiple subjects, whereas tokens have to be issued for every subject in DCACI. This greatly reduces the burden of token management for object owners and improves the scalability. In addition, the introduction of CP-ABE enables attribute-based access control, which is more fine-grained than that achieved by the DCACI.

A preliminary version of the proposed scheme has been published as a conference paper in [37]. Compared with our previous work, this paper includes enhanced security by introducing an additional security measure, i.e., authentication of the subjects in order to prevent the use of illegally-obtained tokens. In addition, a more practical prototype of the proposed scheme is implemented with actual IoT devices. Moreover, detailed evaluation and analysis of the proposed scheme, such as the relationship between the granularity of access control and execution time are carried out, both of which have not been discussed in our previous work.

The remainder of this paper is structured as follows. We first introduce some related work in Section 2. We then describe the preliminaries including the DCACI scheme in Section 3. We illustrate our scheme in Section 4 and show its implementation in Section 5. We then evaluate the performance of our scheme in Section 6. Finally, we summarize the paper in Section 7.

2. Related Work

2.1. Conventional Access Control Schemes

Various access control models to decide the access rights of subjects and access control policies have been proposed, among which Access Control List (ACL), Role-Based Access Control (RBAC), Attribute-Based Access Control (ABAC) and Capability-Based Access Control (CapBAC) are representative ones. ACL is a table associated with an object that describes what subject can perform what kind of actions on the object [38]. In RBAC, subjects are first assigned roles (e.g., guest and administrator), and the access rights are determined for each role [39]. In ABAC, a set of rules, called policies, are defined using the subjects' attributes (e.g., age and affiliation) and the objects' attributes (e.g., identifier and location) [40]. In CapBAC, some kind of tokens (e.g., keys and tickets) are issued to subjects as proof of authority (i.e., capabilities) [41]. The subjects present their tokens to the object owners when accessing the objects.

Based on these models, a variety of conventional access control schemes have been proposed [42–44], most of which fail to cope with large-scale access control for two main reasons. First, the access rights and policies are stored and managed on a centralized server, which turns out to be a single point of failure. This makes the system vulnerable to disorder caused by disasters and attacks by malicious users [45,46]. In the case of attacks, the tampered access rights can lead to an illegal, unintended access control, e.g., unauthorized subjects gain access to some resource. Second, the central server can be a performance bottleneck, failing to process the increasing amount of access requests in large-scale systems. Therefore, access control schemes must be distributed, reliable and scalable to cope with the explosively-growing IoT era.

2.2. Blockchain and Smart Contract

Blockchain is the underlying technology of cryptocurrencies such as Bitcoin [28] and Ethereum [29]. It is a distributed and tamper-resistant database that is managed over P2P networks. In blockchain, pieces of data representing remittance between peers, called transactions, are collected and encapsulated as blocks and appended to the blockchain at regular intervals. One important property of blockchains is their tamper resistance, which is supported by two main features. One is that every block includes the hash value of the previous block, which makes the blockchain tamper evident [31]. When a peer tampers with the content of a block, the hash value of this block changes, which leads to inconsistency in the hash values of all succeeding blocks. This means that all succeeding blocks have to be tampered with as well, in order to maintain consistency. However, recording new data to the blockchain, i.e., adding a block to the blockchain, requires heavy computation and a certain amount of time, which is the second feature [47]. The process of adding a new block to the blockchain is called mining, which is driven by a mathematical puzzle called Proof-of-Work (PoW). Given a block, PoW is the process of finding a random value called nonce, such that the hash value of the block resulting from the nonce can meet a pre-defined difficulty requirement, e.g., starting with a certain number of zeros. It has been known that there is no efficient way to solve PoW, although it is easy to verify. Another fascinating property of blockchains is that the data are distributed. Since every peer stores and updates a copy of the blockchain, the data will not be lost even if some peers stop operating. Because of these attractive properties, blockchains are suitable for storing access rights and policies in terms of access control.

In addition to storing data, recent blockchain systems including Ethereum can handle computation, which is powered by a functionality called smart contract [30]. The core idea is to store executable codes on the blockchain and make peers execute them in a decentralized manner. The peers reach consensus on the execution result of the code and also store the results in the blockchain. This enables distributed and reliable computing, which is suitable for processing access requests in terms of access control.

2.3. Blockchain-Based Access Control Schemes

In [10–13], Ethereum-based distributed CapBAC schemes were proposed, whose idea is to manage the subjects' tokens on the Ethereum blockchain using smart contracts. Thanks to the Ethereum blockchain, the tokens can be stored in a distributed and tamper-resistant manner. When the object receives an access request from the subject, it invokes the smart contract responsible for verifying the subject's token. Thanks to the smart contracts, processing the access request, i.e., the decision making of permitting or denying the request, can be performed in a decentralized and reliable manner.

In [14–17], blockchain-based distributed ABAC schemes were proposed, where the attributes of the subjects and objects, as well as the access control policies are managed on the Ethereum blockchain using smart contracts. Thanks to the Ethereum blockchain, the attributes and policies can be stored in a distributed and tamper-resistant manner. When the subject makes an access request to some object, whether or not the subject's and object's attributes satisfy the corresponding policy is verified by using smart contracts, enabling decentralized and trustworthy access control.

Ethereum-based RBAC schemes were also developed in [18,19], whose idea is to manage the association between subjects and roles and also the association between roles and access permissions by smart contracts. In this way, resource owners can decide which subjects can access the resource by referring to the stored associations. Access control schemes based on other models (e.g., ACL) or other blockchain platforms (e.g., Bitcoin) can be found in [20–27]. For a detailed introduction, please refer to these references.

As described above, blockchain technology along with smart contracts can solve the limitations in conventional access control schemes and realize trustworthy and distributed access control. However, the underlying blockchain technology has also introduced new challenges, i.e., low throughput and high transaction fee. These two problems can impose a great burden on both the administrators and users when enforcing access control in large-scale environments with a large number of users and highly frequent access requests.

3. Preliminaries

3.1. IOTA

IOTA is a next-generation distributed ledger technology designed for the IoT. Unlike blockchain-based cryptocurrencies such as Bitcoin and Ethereum, the distributed ledger of IOTA, called the Tangle, forms a Directed Acyclic Graph (DAG) as shown in Figure 1. Transactions (Txs) are linked together directly using one-way hash functions, instead of being encapsulated in blocks. Since the limit of block size is removed, newly incoming data can be recorded with high throughput. Another significant characteristic of IOTA is the removal of mining. In IOTA, the consistency of the Tangle is maintained by requiring every peer to verify and approve two existing transactions to issue a new transaction. This not only eliminates the need of transaction fees but also accelerates the speed at which new transactions are approved, because the increase of the number of incoming transactions leads to more existing transactions being approved. For these reasons, IOTA is considered to have the potential to solve the problems of high transaction fee and low throughput of blockchain.

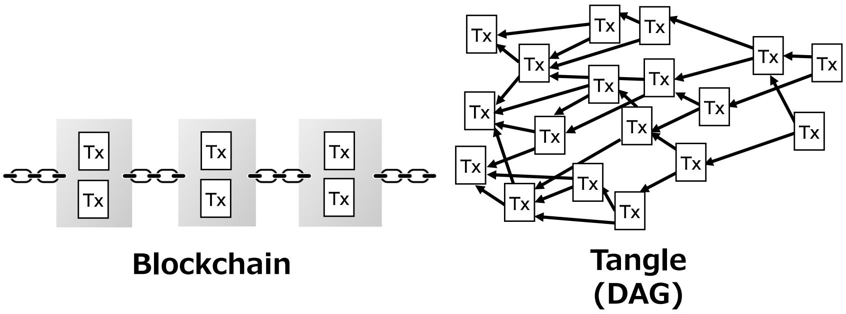


Figure 1. Comparison between blockchain and Tangle.

3.2. Masked Authenticated Messaging (MAM)

Although smart contracts have not been implemented in IOTA, a novel data communication protocol called Masked Authenticated Messaging (MAM) [48] is available, which is a promising solution for access control. Using MAM, peers can record data to and retrieve data from the Tangle in a tamper-resistant fashion. Peers can record data to the Tangle by masking (i.e., encrypting) them and issuing them as special transactions (MAM transactions). Each MAM transaction is associated with an address with which any peer can refer to the transaction. MAM transactions issued by the same peer are linked together chronologically, forming a channel (i.e., a chain of transactions). MAM channels are useful to record and retrieve sequential data, such as periodic recording of temperature data from a smart sensor device, as illustrated in Figure 2. In addition, a signature of the issuer is attached to every MAM transaction, which enables subscribers to verify the authenticity of the issuer (authenticated messaging). A master password that every peer in the IOTA network keeps, called seed, is used to generate the addresses and signatures. Only the owner of the seed can publish messages to his/her channel. With MAM, peers can safely exchange data via the Tangle by subscribing to each other's channel. This enables us to use the Tangle as a database for storing access rights in terms of access control.

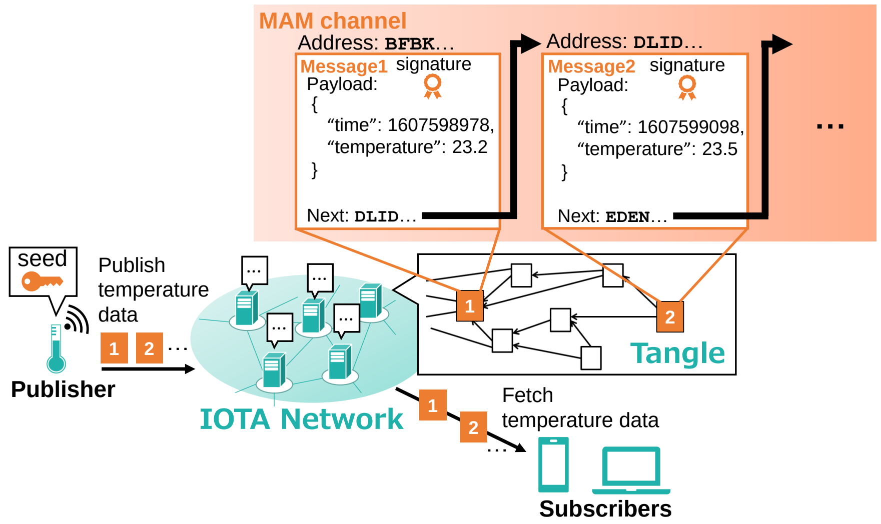


Figure 2. MAM channels.

3.3. DCACI Scheme

In this subsection, we introduce the DCACI scheme in details. The scheme provides four main operations, i.e., GrantAccess , UpdateAccess, DelegateAccess and GetAccess, whose functions are authorizing, updating, delegating and verifying access right, respectively. Since our scheme does not support access right delegation, we focus on the others in this paper.

3.3.1. GrantAccess (Access Right Authorization)

During the initial authorization process, the subject first sends a request for access rights to the object owner (Step 1 in Figure 3). After authenticating the subject, the owner decides the access rights to grant based on local authorization policies and issues the subject an access token (Step 2 in Figure 3). At the same time, the owner records the token to the Tangle as the original copy using MAM (Step 2 in Figure 3). A new MAM channel is generated for every subject to enable access right update as described below.

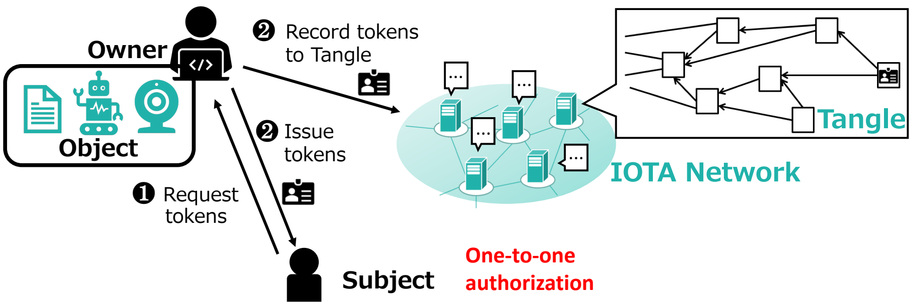


Figure 3. Access right authorization (DCACI).

3.3.2. UpdateAccess (Access Right Update)

When there is a change in the local authorization policies, the owner can update the tokens recorded on the Tangle. The owner issues a new token and attaches it to the corresponding MAM channel as the next message. Therefore, the last message in the MAM channel reflects the latest access rights of the corresponding subject, and the owner will always refer to the last message when validating the subject's access rights.

3.3.3. GetAccess (Access Right Verification)

To access an object, the subject sends the owner an access request along with the token. After authenticating the subject, the owner fetches the original copy of the presented token from the Tangle, i.e., the last message in the MAM channel associated with the subject. Based on the original copy, the owner decides whether or not to grant the requested access. The flow of access right verification of DCACI is shown in Figure 4.

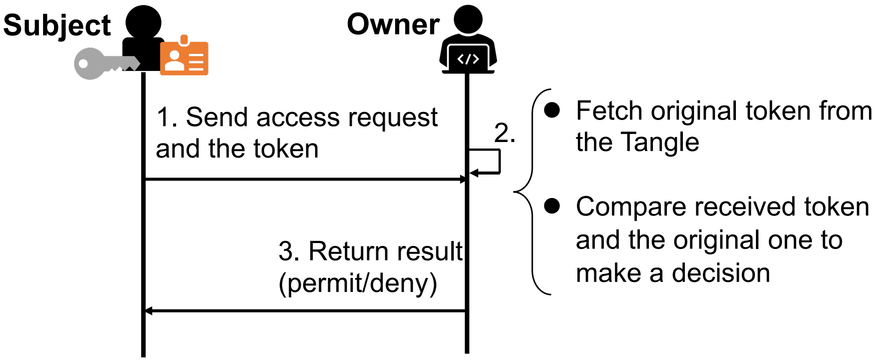

Subject
Owner
1. Send access request and the token
2.
Fetch original token from the Tangle
Compare received token and the original one to make a decision
3. Return result (permit/deny)

Figure 4. Access right verification (DCACI).

3.3.4. Limitations in DCACI

Although DCACI has achieved fee-less distributed access control thanks to IOTA, the framework still suffers from three drawbacks. First, it is assumed that secure communication links are established between the subjects and object owners, while no methods for establishing the links are provided. Thus, requests and tokens are sent without being encrypted, facing the risk of being leaked to malicious users when an insecure channel is used. Second, it supports only one-to-one access control, which means that one token must be recorded for each subject, i.e., one token per subject, increasing the burden of token management for large-scale IoT systems. Finally, it provides no concrete implementation of the authorization process on the owner sides, i.e., the way/model used to decide what access rights should be granted to the subjects before issuing tokens, and the way to authenticate subjects. To overcome these drawbacks, we propose a novel access control framework based on IOTA and the CP-ABE technology to realize more flexible and scalable access control.

3.4. Ciphertext-Policy Attribute-Based Encryption (CP-ABE)

CP-ABE [36] is a type of public-key cryptosystems. Unlike common public-key cryptosystems in which each user possesses a pair of public and private keys, there is only one public key (the master public key) in CP-ABE and private keys are associated with a set of attributes. The master public key and private keys are issued to users by an attribute authority, which associates each private key with the attributes of the corresponding user. For example, the authority may issue a student in the division of Information Science (IS) a private key corresponding to the set of attributes {Division: IS, Role: Student}, and a staff a private key corresponding to the set of attributes {Division: IS, Role: Staff}.

Another difference from common cryptosystems is the introduction of logic formulas called policies into the encryption process. Policies state the conditions of attributes that need to be satisfied to decrypt the ciphertext. The decryption succeeds if and only if the set of attributes associated with the private key satisfies the policy. Figure 5 shows an example. Given a ciphertext encrypted using the policy "Division: IS AND Role: Staff", users with the private key corresponding to the set of attributes {Division: IS, Role: Staff} can decrypt it, while users with the private key corresponding to the set of attributes {Division: IS, Role: Student} cannot.

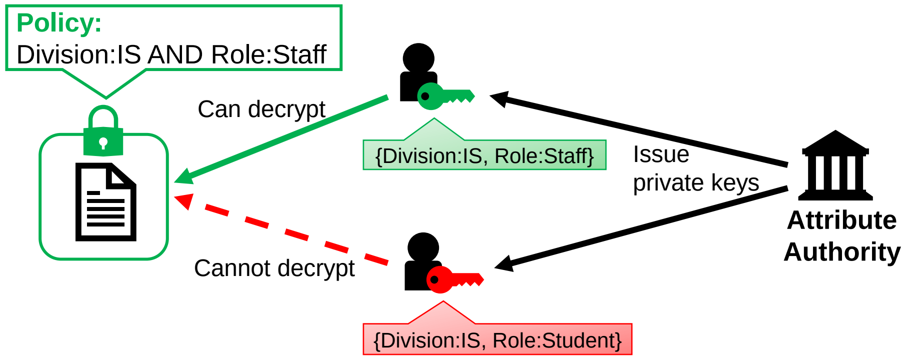

Policy:
Division:IS AND Role:Staff
Can decrypt
Cannot decrypt
{Division:IS, Role:Staff}
{Division:IS, Role:Student}
Issue
private keys
Attribute
Authority

Figure 5. Example of access control to data using CP-ABE.

As seen above, CP-ABE enables fine-grained ABAC to data by restricting the successful decryption to a specific group of users using policies. In our scheme, we encrypt tokens using CP-ABE and store them on the Tangle to achieve flexible access right authorization.

4. Proposed Scheme

We combine CP-ABE with IOTA to solve the issues mentioned in Section 3.3. Access rights are managed on the IOTA Tangle in the form of tokens, which are encrypted using CP-ABE and recorded as MAM transactions. Our scheme is thus a hybrid of CapBAC and ABAC. Please note that a trust authority is required in our scheme to generate and distribute secret keys to the participants of the system. We assume all the secret keys are generated before the deployment of the proposed access control scheme, and thus, we only focus on the access control scheme in this paper. The IOTA here is used as only a database for storing the encrypted tokens, so it can be replaced by the blockchain technology such as bitcoin and/or Ethereum. The only difference is that the blockchain technology may not be able to meet the desired requirements of high throughput and no fee.

4.1. Token Structure

As shown in Figure 6, a token is a JSON object issued by object owners. A token contains a unique identifier (ID), the issuer, the address it is associated with on the Tangle, the policy that must be satisfied to decrypt the token, the current status and a list of access rights. This indicates that only the subjects whose attributes satisfy the policy in the token can decrypt the token and are thus granted the access rights recorded in the token. For instance, the token shown in Figure 6 has been issued by `owner1` and is associated with the address `MKM...ABQ` on the Tangle. Only subjects whose attributes satisfy the policy `Division:IS AND Role:Student` can perform the two actions `TURN_ON` and `TURN_OFF` on the resource `led1/power`. The status field `ACTIVE` can be changed to `INACTIVE` by the owner afterward in order to revocate access rights, which is introduced in detail later. Using this token structure, we design the overall architecture of the proposed scheme.

```
{
    "id":"NMUm6MTRO7",
    "issuer":"owner1",
    "status":"ACTIVE",
    "address":"MKMCRTLVRTTVGIIACHRIWQCOGOKT9TNGEGZBVFMWCBFKFXBAYUMEQKEBJPHYRVHZNECDJCXGCIOBIAABQ",
    "policy":"Division:IS AND Role:Student",
    "rights":[{
        "resource":"led1/power",
        "action":"TURN_ON"
    },{
        "resource":"led1/power",
        "action":"TURN_OFF"
    }]
}
```

Figure 6. Example of a token.

4.2. Access Right Authorization

Figure 7 illustrates how access rights are authorized to the subjects. The object owner first decides the policy to be embedded in the token and the corresponding access rights to specify which group of subjects can perform what actions to the object (Step 1 in Figure 7). Examples of policies and access rights are shown in Table 1. The first example implies that subjects satisfying the policy "Division: IS AND Role: Student" will be authorized to perform two actions "TURN_ON" and "TURN_OFF" on the resource "led1/power". After deciding the policy and access rights, the object owner prepares a token according to a pre-defined structure and encrypts it under the policy using CP-ABE. This means that only those who satisfy the policy can decrypt it. For the first example in Table 1, the object owner will issue a token as shown in Figure 6 and then encrypt it under the policy "Division: IS AND Role: Student".

The owner then records the encrypted token to the Tangle using MAM (Step 2 in Figure 7). A new MAM channel is generated for every policy so that the token can be updated afterward when there is any change in the corresponding access rights, as introduced later. Although the MAM transaction itself is public and visible to any peer, the token can be decrypted only by those with a private key associated with a set of attributes satisfying the policy. Subjects can fetch the encrypted token from the Tangle and decrypt it using their private keys (Step 3 in Figure 7). It is assumed here that the address associated with the MAM transaction containing the encrypted token is made public and available to the subjects. In this way, once the owner has recorded an encrypted token to the Tangle, all subjects satisfying the policy can obtain the token and are authorized the access rights. On the contrary, in DCACI, the owner has to conduct the authorization (although the implementation is not provided) and token issuance processes once for every subject, which increases the burden of object owners. Our scheme not only alleviates the burden of the owner but also enables one-to-many access control. This means that one token is responsible for the access control of a group of subjects, while, in DCACI, one token is only for one subject.

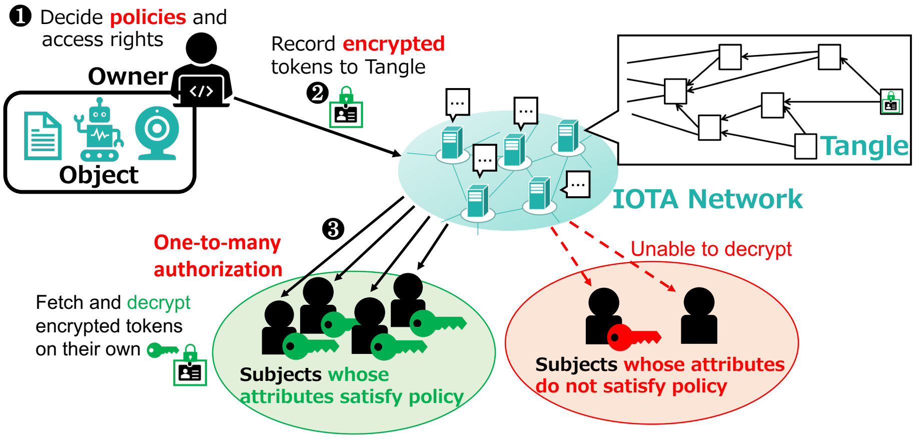

❶ Decide policies and access rights
Owner
Object
Record encrypted tokens to Tangle
❷
Tangle
IOTA Network
❸
One-to-many authorization
Fetch and decrypt encrypted tokens on their own
Unable to decrypt
Subjects whose attributes satisfy policy
Subjects whose attributes do not satisfy policy

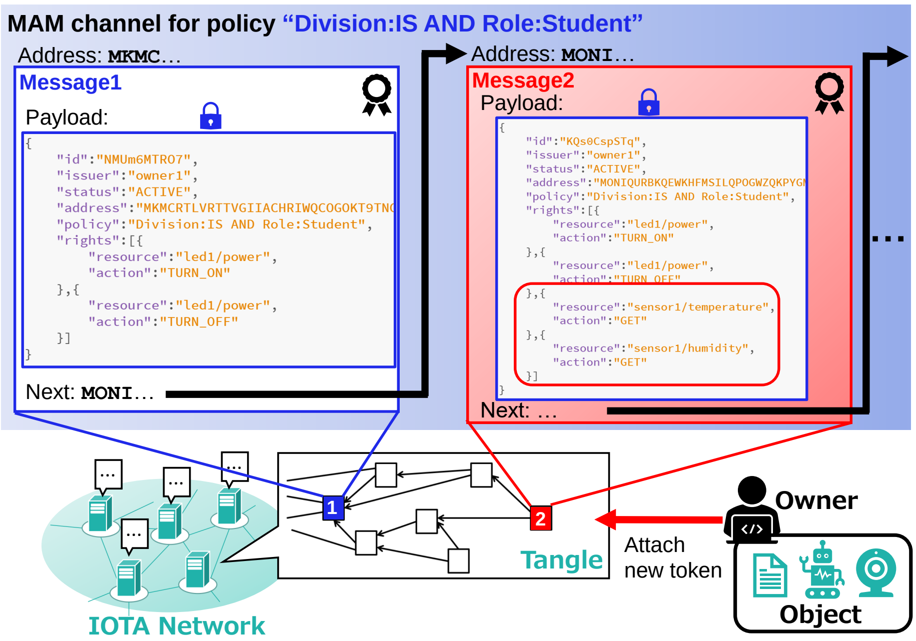
Figure 7. Proposed scheme: access right authorization.

Table 1. Examples of policies and access rights.

Policy	Access Rights
Division: IS AND Role: Student	'TURN_ON' access to 'led1/power' 'TURN_OFF' access to 'led1/power'
Division: IS AND Role: Staff	'TURN_ON' access to 'led1/power' 'TURN_OFF' access to 'led1/power' 'GET' access to 'sensor1/temperature' 'GET' access to 'sensor1/humidity' 'GET' access to 'camera1/snapshpt' 'LOCK' access to 'key1' 'UNLOCK' access to 'key1'

4.3. Access Right Update

Figure 8 illustrates how access rights can be updated by the owner. We use the same approach as in DCACI, i.e., publishing a new token containing the updated access rights as the next message in the corresponding MAM channel. For example, we consider a case where the owner changes the access rights for policy “Division: IS AND Role: Student” from those in Table 1 to those in Table 2. The owner attaches the new token (encrypted using CP-ABE) to the MAM channel corresponding to policy “Division: IS AND Role: Student” as the next message. Since older tokens no longer hold the latest access rights when updates occur, the owner always refers to the last message in the channel when retrieving access rights from the Tangle. Likewise, the owner can also revocate tokens by updating the status field of tokens to “INACTIVE”.



```
{
    "id":"NMUm6MTRO7",
    "issuer":"owner1",
    "status":"ACTIVE",
    "address":"MKMCRTLVRTTVGIIACHRIWQCOGOKT9TNC
    "policy":"Division:IS AND Role:Student",
    "rights":[{
        "resource":"led1/power",
        "action":"TURN_ON"
    },{
        "resource":"led1/power",
        "action":"TURN_OFF"
    }]
}
```

```
{
    "id":"KQs0CspSTq",
    "issuer":"owner1",
    "status":"ACTIVE",
    "address":"MONIQURBKQEWKHFMSILQPOGWZQKPYGM
    "policy":"Division:IS AND Role:Student",
    "rights":[{
        "resource":"led1/power",
        "action":"TURN_ON"
    },{
        "resource":"led1/power",
        "action":"TURN_OFF"
    },{
        "resource":"sensor1/temperature",
        "action":"GET"
    },{
        "resource":"sensor1/humidity",
        "action":"GET"
    }]
}
```

Figure 8. Example of access right update.

Table 2. Policies and access rights after update.

Policy	Access Rights
Division: IS AND Role: Student	’TURN_ON’ access to ’led1/power’ ’TURN_OFF’ access to ’led1/power’ **’GET’ access to ’sensor1/temperature’** **’GET’ access to ’sensor1/humidity’**
Division: IS AND Role: Staff	’TURN_ON’ access to ’led1/power’ ’TURN_OFF’ access to ’led1/power’ ’GET’ access to ’sensor1/temperature’ ’GET’ access to ’sensor1/humidity’ ’GET’ access to ’camera1/snapshpt’ ’LOCK’ access to ’key1’ ’UNLOCK’ access to ’key1’

4.4. Access Right Verification

Although the basic idea in our scheme is similar to GetAccess in DCACI, i.e., subjects can use their tokens to access resources, we introduce an original authentication phase in order to prevent the use of illegally-obtained tokens. This is required because we have to take into account illegal transfer of tokens among subjects and token stolen by malicious subjects. Although it is mentioned in [35] that such cases can be detected by authenticating the subject since tokens are unique to each subject in the scheme, the concrete method to authenticate subjects is not provided.

The proposed verification process consists of two phases, i.e., the authentication phase and the access request phase. The former is required to make sure that the subject indeed satisfies the corresponding policy, and the latter is required to verify that the requested action can be executed using the corresponding access rights.

4.4.1. Authentication Phase

In the authentication phase, the owner imposes a One-Time Password (OTP) authentication on the subject in order to verify that the subject, who is about to use some token, satisfies the corresponding policy embedded inside the token. We design the OTP authentication process such that only subjects with an appropriate private key can obtain the OTP, and thus, those who illegally obtained tokens will not be able to use them.

The flow of the OTP authentication is depicted in Figure 9. Subjects who want to use a token to access some resource first make an authentication request by declaring the policy corresponding to the token (Step 1 in Figure 9). What the owner wants to verify here is that the subject really satisfies the required policy, i.e., holds a private key associated with a set of attributes satisfying the policy. To do so, the owner sends back an OTP encrypted under the policy using CP-ABE (Steps 2 to 3 in Figure 9), which means that only those who satisfy the policy can decrypt it. At the same time, the owner registers the policy-OTP pair to a list for later confirmation.

On receiving the encrypted OTP, the subject decrypts it using his/her private key and presents it to the owner through an encrypted access request (Steps 4 to 6 in Figure 9), along with the resource to access, the action to perform and the token to present. The request is encrypted using CP-ABE, ensuring secure communication between the subject and owner. The subject encrypts the request under some policy that allows only the object owner to decrypt it (e.g., "Role: Owner").

On receiving the encrypted access request, the owner decrypts it using his/her private key issued by the authority. The owner extracts the OTP from the access request and the policy from the presented token, and then checks the pair against the policy-OTP pair list registered in Step 2 (Step 7 in Figure 9). If the pair exists in the list, the OTP is considered valid and cleared from list, after which the owner proceeds to the request phase. Otherwise, the OTP is invalid and the request is thus rejected.

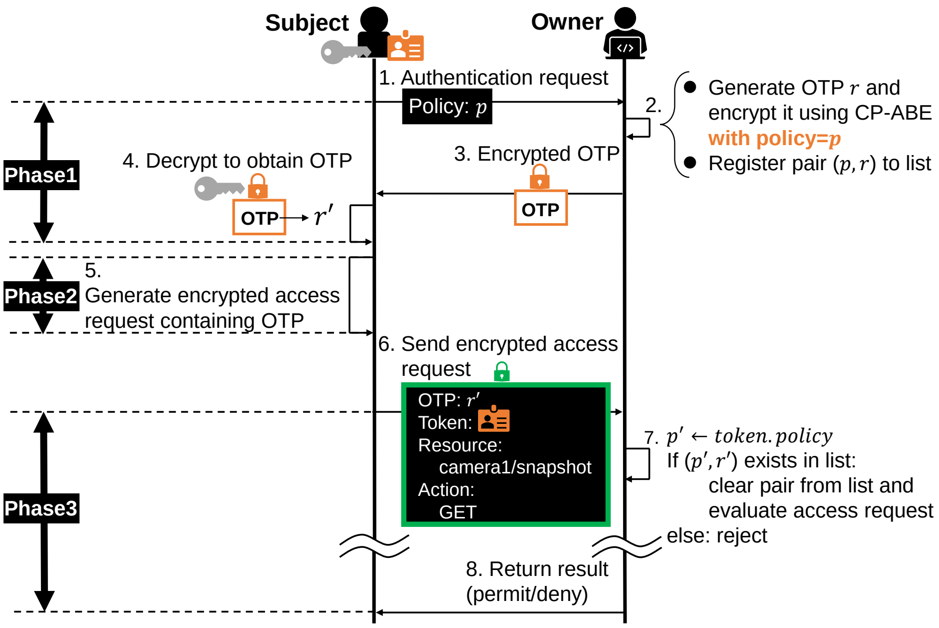

Subject
Owner
1. Authentication request
Policy: p
2.
Generate OTP r and encrypt it using CP-ABE with policy=p
Register pair (p, r) to list
Phase1
4. Decrypt to obtain OTP
3. Encrypted OTP
OTP
OTP → r'
Phase2
5. Generate encrypted access request containing OTP
6. Send encrypted access request
OTP: r'
Token:
Resource: camera1/snapshot
Action: GET
Phase3
7. $p' \leftarrow token.policy$
If (p', r') exists in list:
clear pair from list and evaluate access request
else: reject
8. Return result (permit/deny)

Figure 9. Proposed scheme: access right verification (authentication phase).

In this way, subjects who illegally obtained the token will not be able to obtain the valid OTP and thus cannot generate a valid access request, meaning that they cannot use the token.

4.4.2. Access Request Phase

After the authentication phase, the owner evaluates the access request based on the presented token. The owner first fetches the original copy of the corresponding token (i.e., the one issued during authorization) from the Tangle. The presented token is checked against the original token to verify its authenticity. Since the Tangle is tamper-proof, any modification in the token can be detected in this process. The access request is rejected if the token verification fails. If the token is valid, the resource and action are evaluated based on the list of access rights contained in the token. If the requested action does not exist in the list, the request is rejected because it is an attempt of unauthorized access.

5. Implementation

To show the feasibility of our scheme, we have implemented a proof-of-concept prototype using the IOTA Mainnet [49] and IoT devices.

5.1. System Configuration

As shown in Figure 10, we used a Microsoft Surface Laptop 3 (1.5 GHz Intel Core i7, 16 GB RAM) as the object owner and a Google Pixel 3 XL (2.5 GHz Qualcomm Snapdragon 845, 4 GB RAM) as the subject. The owner manages three objects, "camera1" (IODATA TS-WRLP/E), "sensor1" (OMRON 2JCIE-BU01) and "led1" (TP-LINK KL130). The objects can be accessed through the IoT gateway (NEC EGW001: 1.46 GHz Intel Atom, 2 GB RAM) which performs access right verification. The official JavaScript API [50] was used to issue and fetch MAM transactions and an implementation based on [51] was used to handle the CP-ABE encryption and decryption. The gateway service was implemented by an HTTP server which listens to requests from the subject. At the subject side, we developed a native Android application which can fetch MAM transactions from the Tangle, handle CP-ABE and fire access requests. All entities participate in the IOTA Mainnet as clients and communicate with a full node (https://nodes.thetangle.org (accessed on 25 June 2021)) to issue and fetch MAM transactions.

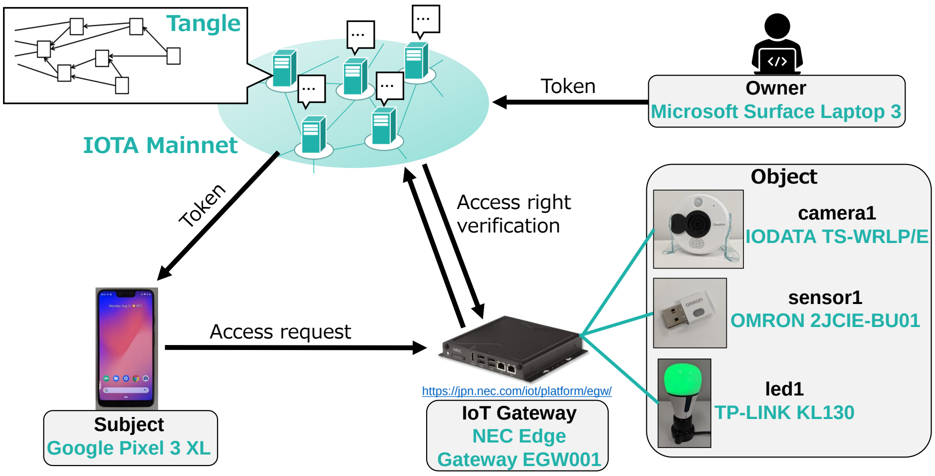
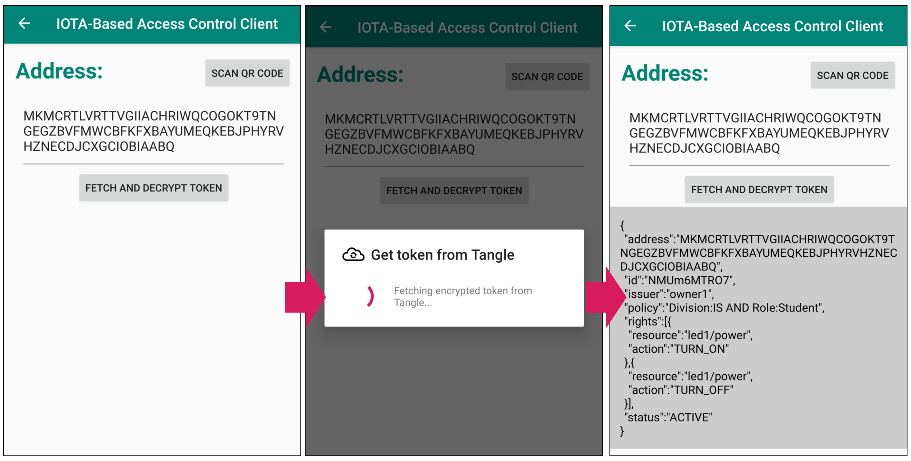


Figure 10. Configuration of the prototype system.

5.2. Access Right Authorization

For simplicity, we limited the tokens to the two illustrated in Table 2 (i.e., the student token and the staff token) and issued the subject a private key associated with the set of attributes {Division: IS, Role: Student}. Figure 11 shows the result of fetching and decrypting the student token using the private key. We can see that the access right is successfully authorized via the Tangle to the subject. Decrypted tokens will be stored on local storage as text files, which can be used to access resources later.



```
MKMCRTLVRTTVGIIACHRIWQCOGOKT9TN
GEGZBVFMWCBFKFXBAYUMEQKEBJPHYRV
HZNECDJCXGCIOBIAABQ
```

Get token from Tangle

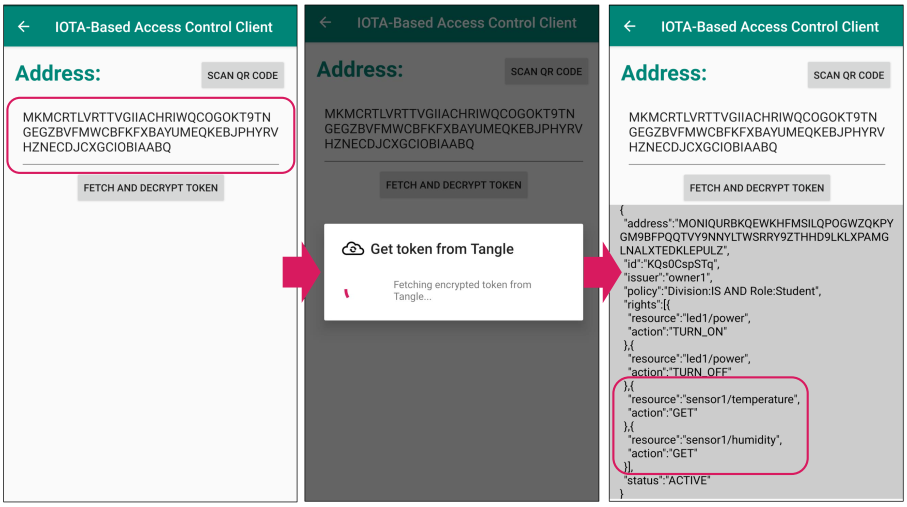
Fetching encrypted token from Tangle...

```
{
 "address":"MKMCRTLVRTTVGIIACHRIWQCOGOKT9T
NGEGZBVFMWCBFKFXBAYUMEQKEBJPHYRVHZNEC
DJCXGCIOBIAABQ",
 "id":"NMUm6MTRO7",
 "issuer":"owner1",
 "policy":"Division:IS AND Role:Student",
 "rights":[{
  "resource":"led1/power",
  "action":"TURN_ON"
 },{
  "resource":"led1/power",
  "action":"TURN_OFF"
 }],
 "status":"ACTIVE"
}
```

Figure 11. Token fetching from Tangle and decryption.

5.3. Access Right Update

Figure 12 shows the result of fetching and decrypting the student token *after* the access right update mentioned in Section 4.3. The application walks through the MAM channel starting from the first message which contains the first token shown in Figure 11 (note that the address is the same as in Figure 11) and fetches the latest message, i.e., the second message, which contains the updated token. We can see that the access right is successfully updated (the access right pertaining to "sensor1" has been added) and that the subject can always obtain the latest token via the Tangle.



```
MKMCRTLVRTTVGIIACHRIWQCOGOKT9TN
GEGZBVFMWCBFKFXBAYUMEQKEBJPHYRV
HZNECDJCXGCIOBIAABQ
```

Get token from Tangle

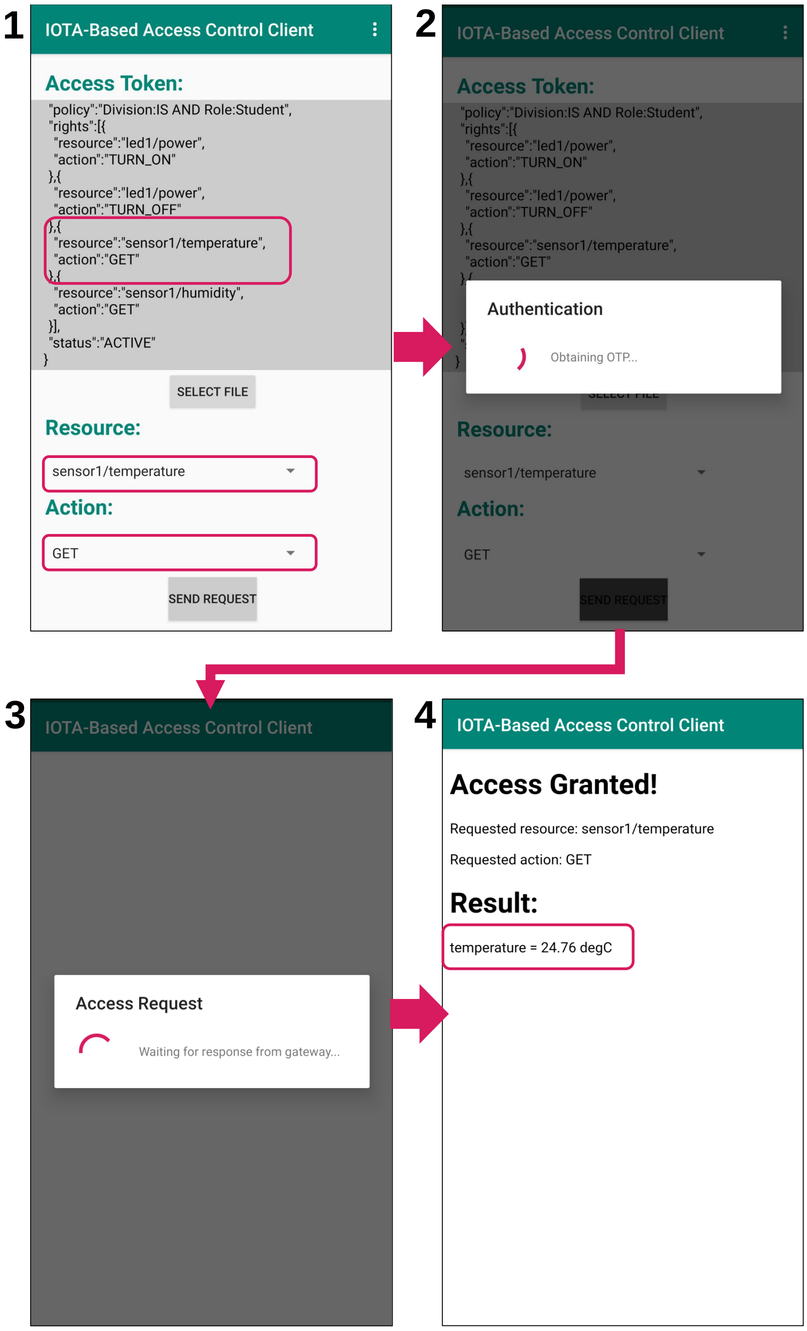
Fetching encrypted token from Tangle...

```
{
 "address":"MONIQURBKQEWKHFMSILQPOGWZQKPY
GM9BFPQQTVY9NNYLTWSRRY9ZTHHD9LKLXPAMG
LNALXTEDKLEPULZ",
 "id":"KQs0CspSTq",
 "issuer":"owner1",
 "policy":"Division:IS AND Role:Student",
 "rights":[{
  "resource":"led1/power",
  "action":"TURN_ON"
 },{
  "resource":"led1/power",
  "action":"TURN_OFF"
 },{
  "resource":"sensor1/temperature",
  "action":"GET"
 },{
  "resource":"sensor1/humidity",
  "action":"GET"
 }],
 "status":"ACTIVE"
}
```

Figure 12. Token fetching from Tangle and decryption after access right update.

5.4. Access Right Verification

Figure 13 shows the result of a student accessing a resource using the student token and his/her CP-ABE private key. The subject (i.e., the student) first selects the token to present (the one saved to local storage during authorization), the resource to access ("sensor1/temperature") and the action to perform ("GET") (Image 1 in Figure 13). By pressing the "SEND REQUEST" button, the authentication phase is initiated, where an authentication request with the corresponding policy "Division:IS AND Role:Student" is sent to the owner (Image 2 in Figure 13). In this case, authentication succeeds since the subject holds a private key associated with the set of attributes {Division: IS, Role: Student} as mentioned earlier, and is thus able to decrypt the encrypted OTP sent back from the owner. Therefore, the subject can proceed to the access request phase (Image 3 in Figure 13). In the access request phase, the presented token is authentic and it contains the right to perform the action "GET" on the resource "sensor1/temperature". Therefore, access is granted (Image 4 in Figure 13).



```
"policy":"Division:IS AND Role:Student",
"rights":[{
 "resource":"led1/power",
 "action":"TURN_ON"
},{
 "resource":"led1/power",
 "action":"TURN_OFF"
},{
 "resource":"sensor1/temperature",
 "action":"GET"
},{
 "resource":"sensor1/humidity",
 "action":"GET"
}],
"status":"ACTIVE"
}
```

Resource: sensor1/temperature

Action: GET

Authentication

Obtaining OTP...

Access Request

Waiting for response from gateway...

Access Granted!

Requested resource: sensor1/temperature

Requested action: GET

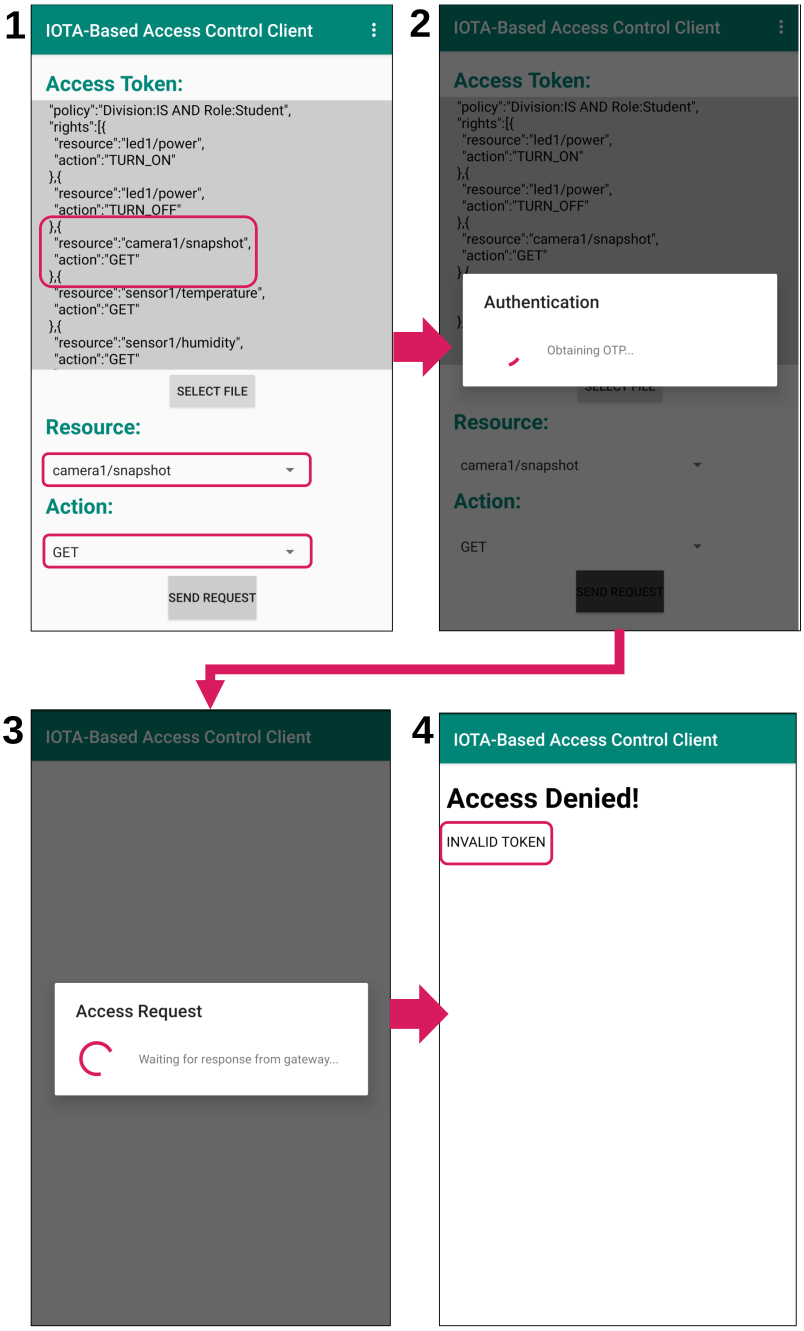
Result:

temperature = 24.76 degC

Figure 13. Granted access.

On the other hand, Figure 14 shows the result of an attempt by a student to perform the action "GET" on the resource "camera1/snapshot" (which is exclusive to the staff) using a tampered token. Based on the student token, a new "rights" clause containing "camera1/snapshot" and "GET" as the resource field and action field, respectively, has been added to the token (Image 1 in Figure 14). In the authentication phase, an authentication request with the corresponding policy "Division:IS AND Role:Student" is sent to the owner

(Image 2 in Figure 14). Although the subject can pass the authentication in the same manner as the granted case (Image 3 in Figure 14), the owner detects the modification on the token in the request phase by checking the presented token against the original token fetched from the Tangle. Therefore, access is rejected (Image 4 in Figure 14).



1

IOTA-Based Access Control Client

Access Token:

```
"policy":"Division:IS AND Role:Student",
"rights":[{
 "resource":"led1/power",
 "action":"TURN_ON"
},{
 "resource":"led1/power",
 "action":"TURN_OFF"
},{
 "resource":"camera1/snapshot",
 "action":"GET"
},{
 "resource":"sensor1/temperature",
 "action":"GET"
},{
 "resource":"sensor1/humidity",
 "action":"GET"
```

SELECT FILE

Resource: camera1/snapshot

Action: GET

SEND REQUEST

2

IOTA-Based Access Control Client

Access Token:

```
"policy":"Division:IS AND Role:Student",
"rights":[{
 "resource":"led1/power",
 "action":"TURN_ON"
},{
 "resource":"led1/power",
 "action":"TURN_OFF"
},{
 "resource":"camera1/snapshot",
 "action":"GET"
},{
```

Authentication

Obtaining OTP...

Resource: camera1/snapshot

Action: GET

SEND REQUEST

3

IOTA-Based Access Control Client

Access Request

Waiting for response from gateway...

4

IOTA-Based Access Control Client

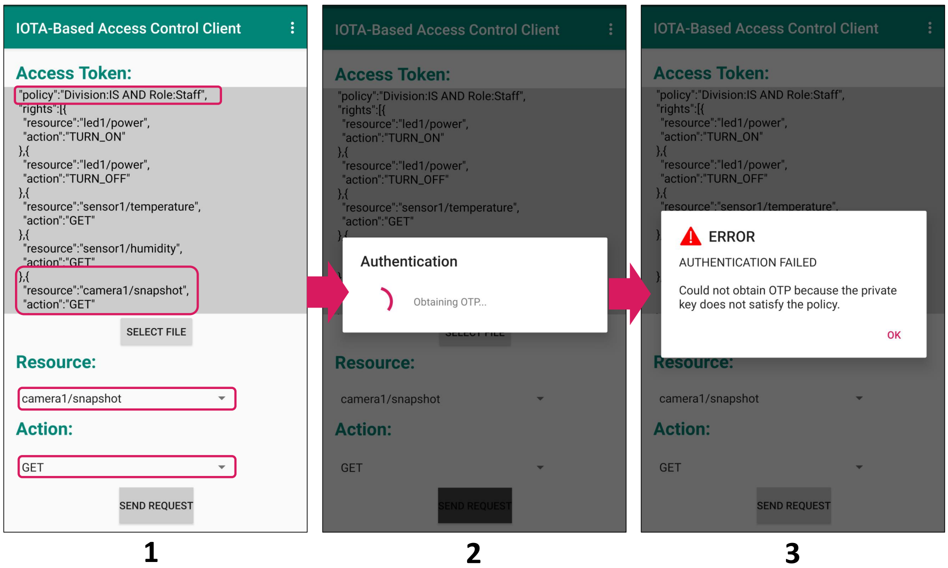
Access Denied!

INVALID TOKEN

Figure 14. Denied access due to tampered token.

Moreover, in Figure 15, we consider the case of the unauthorized access with a leaked/stolen token, where a student obtains the staff token illegally and attempts to use it for resource access. The student first selects the staff token as the token to present, "camera1/snapshot" as the resource (which is exclusive to the staff) and "GET" as the action (Image 1 in Figure 15). In the authentication phase, an authentication request for policy "Division:IS AND Role:Staff " is sent to the owner (Image 2 in Figure 15), since the selected token is corresponding to the policy "Division:IS AND Role:Staff". This makes the owner send back an OTP encrypted using CP-ABE under the policy "Division:IS AND Role:Staff".

However, the student's private key is associated with the set of attributes {Division: IS, Role: Student} and is thus unable to decrypt the encrypted OTP. Therefore, the student cannot proceed to the access request phase and the attempt to use the token is prevented (Image 3 in Figure 15). As can be seen here, both the right token and an appropriate private key satisfying the corresponding policy are needed to access resources.



IOTA-Based Access Control Client

Access Token:

```
"policy":"Division:IS AND Role:Staff",
"rights":[{
 "resource":"led1/power",
 "action":"TURN_ON"
},{
 "resource":"led1/power",
 "action":"TURN_OFF"
},{
 "resource":"sensor1/temperature",
 "action":"GET"
},{
 "resource":"sensor1/humidity",
 "action":"GET"
},{
 "resource":"camera1/snapshot",
 "action":"GET"
```

SELECT FILE

Resource: camera1/snapshot

Action: GET

SEND REQUEST

Authentication

Obtaining OTP...

ERROR

AUTHENTICATION FAILED

Could not obtain OTP because the private key does not satisfy the policy.

OK

Figure 15. Denied access due to stolen token.

6. Performance Evaluation

In this section, we first evaluate the scalability/throughput performance of our scheme in terms of execution time and then compare our scheme with DCACI in terms of both single-operation execution time in the case with one subject and total execution time in the case with multiple subjects.

6.1. Scalability/Throughput

We first discuss the scalability/throughput (i.e., the ability of processing access requests per unit time) of our scheme from the point of an ABAC scheme by measuring the execution time of each operation.

Intuitively, the smaller the execution time is, the higher the scalability will be. As mentioned in Section 1, our scheme enables flexible and fine-grained access control by properly fining the policies, i.e., setting complicated logic formulas of attributes, which leads to an increased number of attributes contained in the policies. In addition, in large-scale environments, there can be a large number of policies to describe the complex local authorization policies. Given these points, we investigate how the execution time changes (1) when the number of attributes in the policies increases and (2) when the number of policies increases.

6.1.1. Execution Time vs. Number of Attributes

As shown in Table 3, we consider four cases in which the policy consists of 3, 6, 9 and 12 attributes, respectively, and the number of the corresponding access rights was fixed to three. The average execution time was measured over 100 executions for each operation, using the prototype shown in Figure 10.

Figure 16 shows the results for access right authorization at the owner side, i.e., issuing a token, encrypting it using CP-ABE and attaching it to the Tangle. We can see that the overall execution time is proportional to the number of attributes in the policy. A careful observation indicates that attaching the token to the Tangle accounts for the majority of the execution time, which is about 98% for all the four cases. This is mainly because the encrypted token is stored on the Tangle in the form of MAM transactions and we need to perform time-consuming PoW for each transaction in IOTA. The percentage of the time consumed by attaching the token to the Tangle is proportional to the number of attributes as well, since more attributes in the policy leads to more transactions being issued. More specifically, more attributes in the policy leads to a larger encrypted token (this is because the policy is embedded into the ciphertext) and each transaction can contain only a constant amount of data. Therefore, a large token has to be split into multiple transactions and these transactions are stored together on the Tangle [52]. It should be noted that the node we used in the experiment (https://nodes.thetangle.org (accessed on 25 June 2021)) supports remote PoW (i.e., the node performs the PoW on behalf of the client so that client devices with limited computing power can issue transactions) [53], and the execution time of attaching the token to the Tangle is thus highly dependent on the condition of the node, such as computation power and the amount of load at the time of request. This also means that the execution time is highly dependent on the performance of the client device when connecting to a node that does not support remote PoW. As for CP-ABE encryption, it has been shown by the authors in [36] that the execution time is proportional to the number of attributes in the policy, and we can see the same results from Figure 16. The remaining part of the time (i.e., "Others" in Figure 16) apart from those for attaching the token to the Tangle and CP-ABE encryption includes time for string operations pertaining to token issuance and conversion between `JSON` objects and Java objects, which are almost independent of the number of attributes.

Table 3. Policies and access rights after update (a_i: attribute i).

Policy	Access Rights
a_1 AND a_2 AND a_3	'action1' access to 'device1'
a_1 AND a_2 AND $a_3 \cdots$ AND a_6	'action2' access to 'device2'
a_1 AND a_2 AND $a_3 \cdots$ AND a_9	'action3' access to 'device3'
a_1 AND a_2 AND $a_3 \cdots$ AND a_{12}	

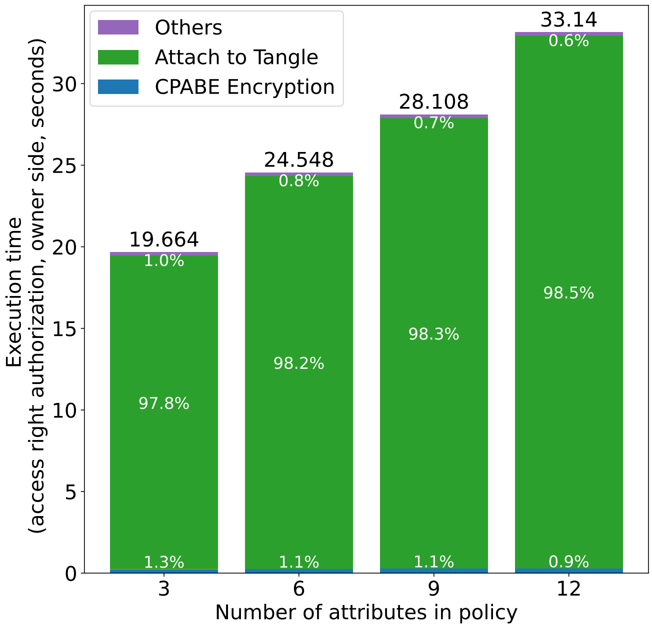


Figure 16. Access right authorization time (owner) vs. number of attributes.

Figure 17 shows the results for access right authorization at the subject side, i.e., fetching an encrypted token from the Tangle and decrypting it using CP-ABE. We can see that the execution time is proportional to the number of attributes in the policy. We can see from the breakdown of the execution time that fetching the token from the Tangle accounts for the majority of the execution time (more than 85% in all experiments). This is because the subject has to fetch all the transactions over which the encrypted token is fragmented, and then concatenates the transactions to reconstruct the encrypted token (i.e., the inverse operation of attaching the token to the Tangle). As more attributes lead to more transactions (as mentioned above in the authorization at the owner side), the execution time is proportional to the number of attributes. As for CP-ABE decryption, it has been shown by the authors in [36] that the execution time is proportional to the number of attributes in the policy and the same conclusion can be drawn from Figure 17. An interesting finding from Figure 17 shows that the percentage of the time for fetching the token from the Tangle decreases as the number of attributes in the policy increases. This is because the time required for CPABE decryption also increases as the number of attributes increases and this increase rate is larger than that of the time for fetching a token from the Tangle, which thus results in the reduction of the proportion of time for token fetching.

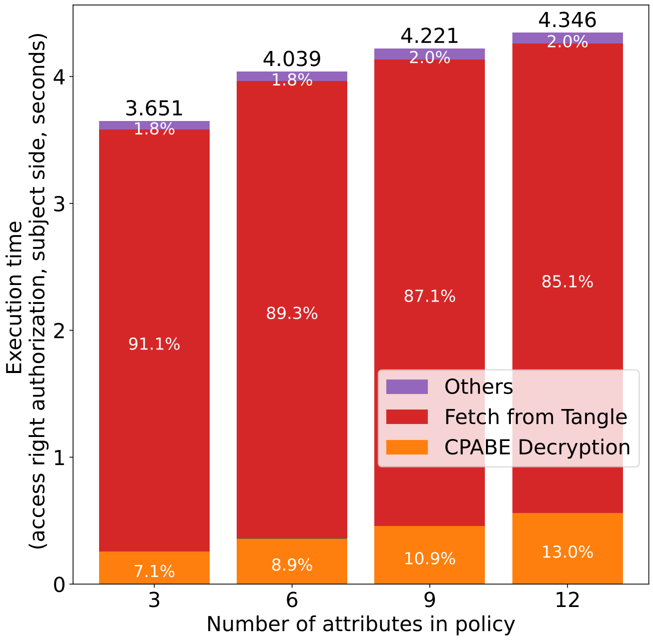


Figure 17. Access right authorization time (subject) vs. number of attributes.

Figure 18 shows the results for access right update, i.e., issuing a new token, encrypting it using CP-ABE and attaching it to the Tangle by the owner. The updated access rights were selected randomly from three cases, i.e., reduced rights (one right), increased rights (five rights) and token inactivation (i.e., setting the status field to "INACTIVE"). Since it consists of the same operations as the authorization at the owner side, we can see similar results from both figures (i.e., Figures 16 and 18).

Next, we evaluate how the number of attributes affects the execution time of access right verification, which is divided into three phases as shown in Figure 9.

Figure 19 shows the execution time results for obtaining and decrypting an encrypted OTP from the owner in the first phase. We can see that the execution time in this phase is proportional to the number of attributes in the policy. The two main processes, CP-ABE encryption at the owner side (OTP encryption) and CP-ABE decryption at the subject side (OTP decryption) are both proportional to the number of attributes in the policy, as mentioned earlier. The remaining time is consumed by other processes including the generation of the authentication request, simple string operations pertaining to CP-ABE and communication between the subject and the owner. We can see from Figure 19 that, as the number of attributes in the policy increases, the percentage of time for encryption increases while that for decryption remains nearly constant. In addition, the percentage of time for other operations decreases. The main reason for this phenomenon is that the increase rate of the time for CPABE encryption is larger than that for CPABE decryption, resulting in the increase of the proportion of the time for CPABE encryption in the total time.

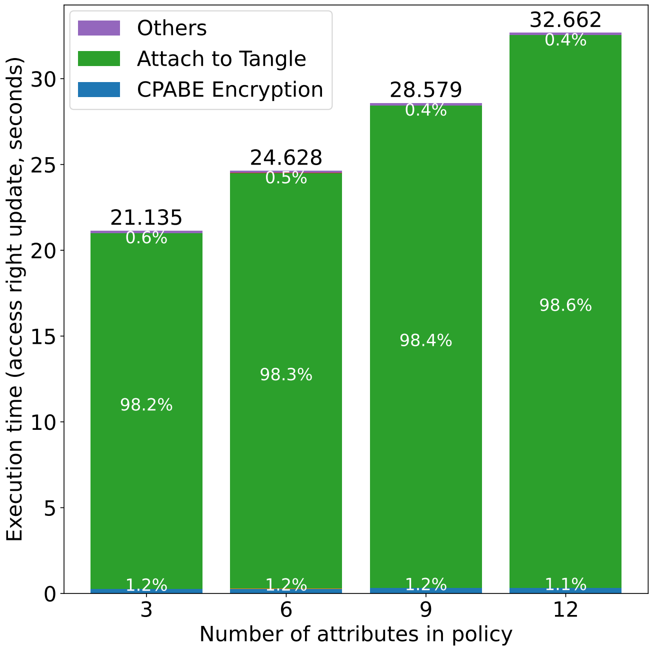


Figure 18. Access right update time vs. number of attributes.

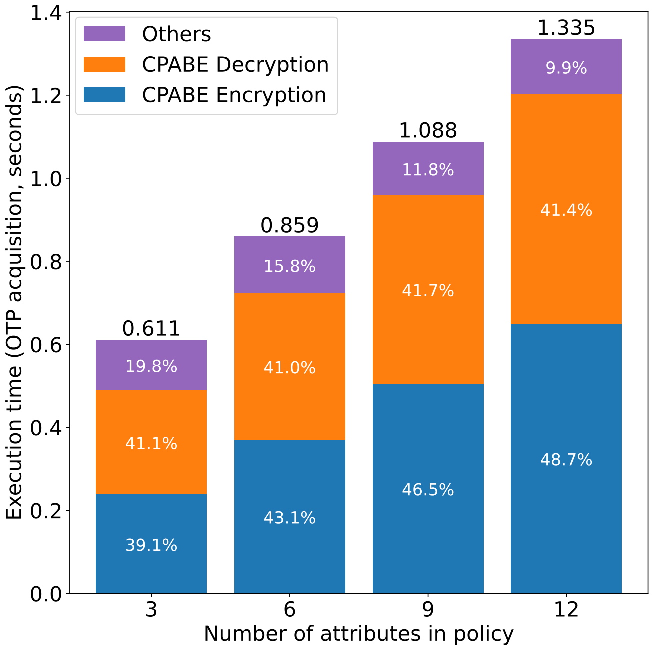


Figure 19. Access right verification time (Phase 1) vs. number of attributes.

Figure 20 shows the average execution time (measured at the subject side) of the second phase, i.e., generating an encrypted access request using the OTP obtained in the first phase. We can see that the execution time is almost constant regardless of the number of attributes. This is because the policy used to encrypt the access request is “Role: Owner” (so that only the owner can decrypt the request, as mentioned in Section 4), which is irrelevant to the “policy” (i.e., the policy for encrypting the token during authorization) in the x axis. The “Others” corresponds to the time used for simple string operations in CP-ABE.

Figure 21 shows the average execution time (owner side) of evaluating the encrypted access request received from the subject in the third phase. We can see that the execution time is proportional to the number of attributes in the policy. Similar to the authorization at the subject side, the owner has to fetch the MAM transactions containing the original copy of the presented token, which accounts for the majority of the execution time and consumes time in proportion to the number of attributes as argued before. CP-ABE decryption includes decrypting both the access request and the original copy of the token and consumes time in proportion to the number of attributes, which has also been argued above. Similar to Figure 17, Figure 21 also shows that the percentage of the time for fetching the token from the Tangle decreases as the number of attributes in policy increases. This is due to the different increase rates of the time for token fetching and CPABE encryption as explained in the discussion of Figure 17.

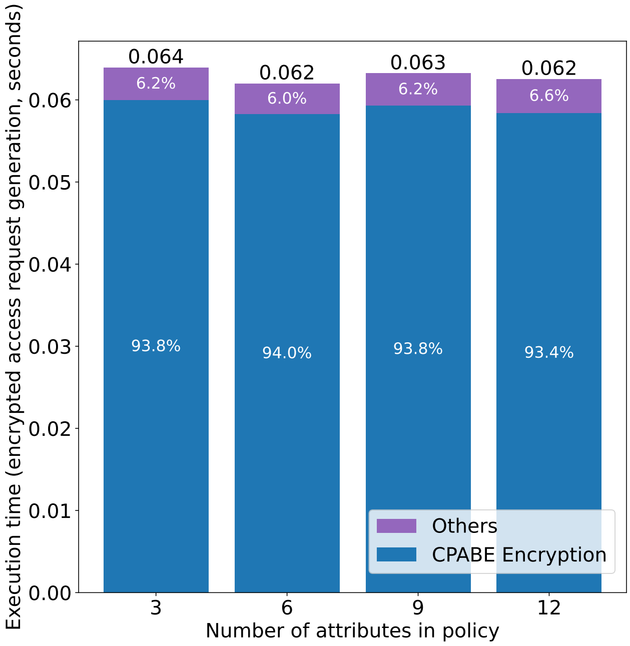


Figure 20. Access right verification time (Phase 2) vs. number of attributes.

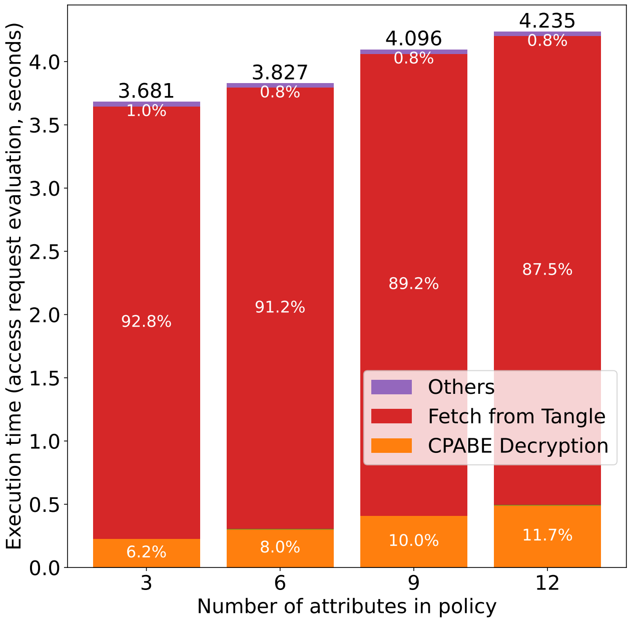


Figure 21. Access right verification time (Phase 3) vs. number of attributes.

6.1.2. Execution Time vs. Number of Policies

In this section, we qualitatively analyze how the number of policies affects the execution time of each operation.

- Access Right Authorization
  - Owner Side
    Since tokens are issued and recorded to the Tangle for each policy, the execution time of access right authorization is proportional to the number of policies and is equal to the sum of the execution time of issuing and recording the tokens for all policies, each of which depends on the number of attributes included in the policy, as shown in Section 6.1.1. Although this is expected to take a considerable time due to attaching the tokens to the Tangle, it is done only once because all subjects can be authorized once all the tokens are recorded to the Tangle.
  - Subject Side
    As introduced in Section 4, a MAM channel is generated for each policy when the encrypted tokens are recorded to the Tangle. Subjects can directly refer to the channels of their interests using the addresses associated with the tokens. Therefore, obtaining tokens from the Tangle is independent of other policies, and thus, the number of policies does not affect the execution time.
- Access Right Update
  As introduced in Section 4, the tokens of the same policy are linked together in a MAM channel. The owner simply attaches the new token to the corresponding channel, without any information about other policies. Therefore, the token update can be performed individually and is independent of the number of policies.
- Access Right Verification
  We discuss each phase illustrated in Figure 9.
  - OTP Acquisition (Phase 1 in Figure 9)
    When making an authentication request, the subject can extract the policy from his/her token and information about other policies is unnecessary. CP-ABE encryption and decryption, which are the main operations, do not require other policies either. Therefore, the increase of the number of policies does not affect the time of OTP acquisition.
  - Encrypted Access Request Generation (Phase 2 in Figure 9)
    The information needed to generate an encrypted access request is the OTP obtained through the authentication request (i.e., in the OTP acquisition phase), the token, the resource to access and the action to perform, all of which are independent of other policies. Therefore, the increase of the number of policies does not affect the time of access request generation.
  - Access Request Evaluation (Phase 3 in Figure 9)
    The main operation here is fetching the original copy of the presented token. Similar to the authorization process at the subject side, this can be performed independently thanks to MAM channels and is thus irrelevant to the number of policies.

As argued above, only the initial token recording involves iterating over all policies and operations that are invoked frequently (e.g., fetching encrypted tokens from the Tangle) are independent of the number of policies. Therefore, we can conclude that the increase of the number of policies has limited impact on the overall execution time.

6.2. Comparison with DCACI

In this subsection, we compare our scheme with DCACI in terms of both single-operation execution time and total execution time. For comparison, we have implemented the DCACI scheme based on the paper [35], using the IOTA Mainnet. The prototype shown in Figure 10 was used for both schemes, except that the subject side was executed on the Surface Laptop 3 for DCACI. As for the evaluation scenario we suppose an educational institution consisting of students and staff members, and the access rights will be authorized to them according to Table 1.

We consider two scenarios: (1) one object owner and one subject (one-to-one scenario) for comparing single-operation execution time and (2) one object owner and multiple subjects (one-to-many scenario) for comparing the total execution time. We show that our scheme can greatly reduce the execution time of authorizing access rights to a large number of subjects by one-to-many access control.

6.2.1. One Owner and One Subject (One-to-One Scenario)

We first consider one owner and one subject who is a member of the staff. The average execution time for each operation has been measured for both schemes and compared.

Figure 22 shows the results for access right authorization. As can be seen in the figure, our scheme requires extra time, especially at the subject side, which takes about 8.5% of the total time. This is because the subject has to fetch and decrypt the encrypted token from the Tangle in our scheme, while, in DCACI, the owner directly issues the token to the subject in response to the token issuance request of the subject.

Moreover, the breakdown of the execution time is shown in Figure 23. We can see that attaching the token to the Tangle accounts for the majority of the execution time, as also discussed previously. Attaching the token to the Tangle takes longer in our scheme, because the token to attach in our scheme is larger than that in DCACI due to CP-ABE encryption. Although the increased token size also leads to an increased CP-ABE encryption time in our scheme compared with DCACI, this increase is much smaller than that caused by attaching the token to the Tangle. Therefore, we can conclude that CP-ABE encryption has limited impact on the overall execution time. The same phenomenons can be observed at the subject side, where fetching the token from the Tangle consumes most of the time.

Figure 24 shows the comparison results for access right update. The updated access rights were selected randomly from three cases, i.e., adding rights, reducing rights and setting status to "INACTIVE". Similar to the results in the authorization, attaching the token to the Tangle is dominant and CP-ABE encryption has little impact.

Finally, Figure 25 shows the comparison results for access right verification. We can see that fetching the token from the Tangle accounts for the majority of the execution time. In the proposed scheme, the "Others" takes a longer time (about 14.9%) than that of DCACI, because the subject and owner have to communicate two times, i.e., one in the authentication phase and the other in the request phase.

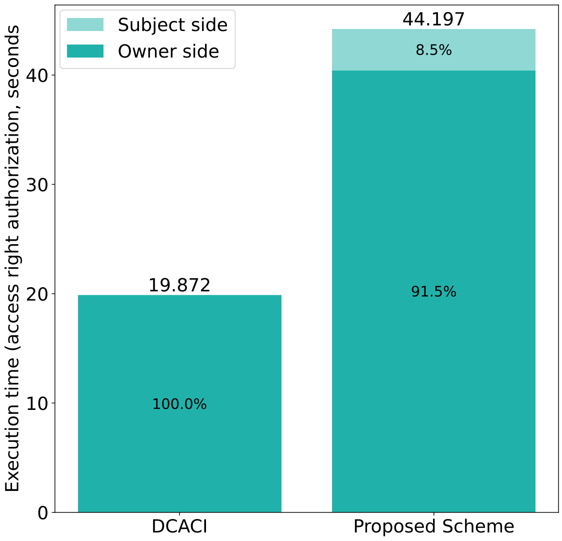


Figure 22. Comparison with DCACI: access right authorization.

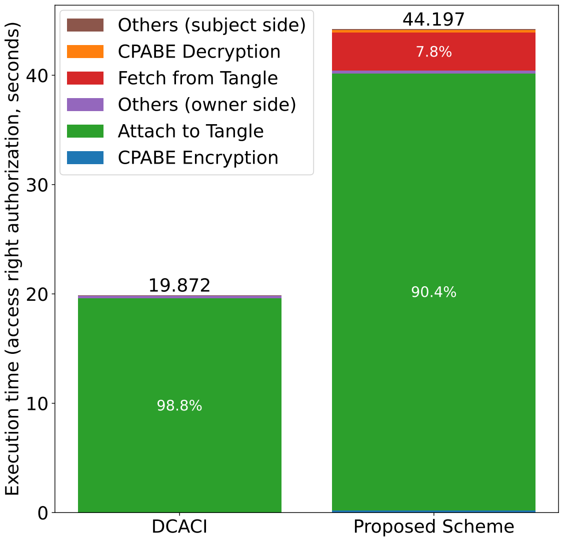


Figure 23. Comparison with DCACI: access right authorization (with breakdown).

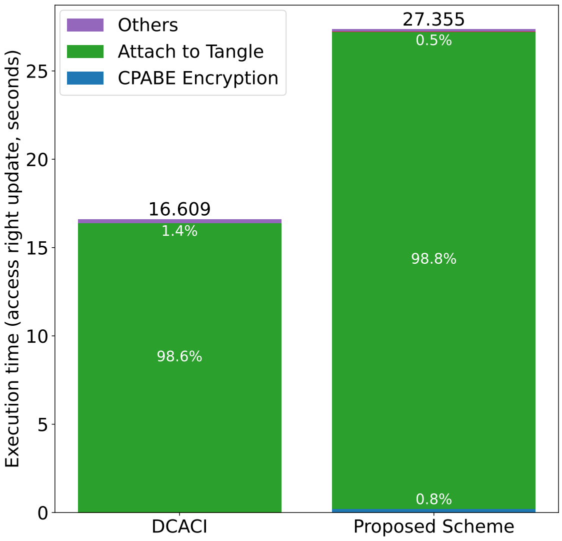


Figure 24. Comparison with DCACI: access right update.

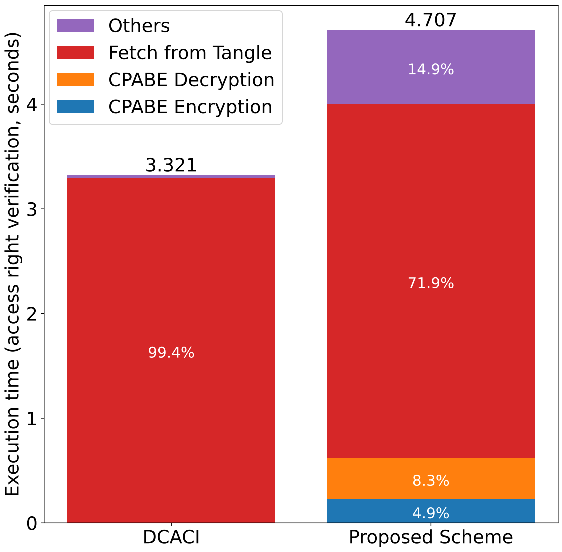


Figure 25. Comparison with DCACI: access right verification.

As can be seen above, experimental results show that our scheme is slower than DCACI for every operation. However, it should be noted that the results for DCACI do not include the security measures, while the results for our scheme do (e.g., establishing secure channels and authentication of subjects).

6.2.2. One Owner and Multiple Subjects (One-to-Many Scenario)

Taking into account that our scheme supports one-to-many access control, i.e., one token can be used to authorize multiple subjects, we can expect that our scheme outperforms DCACI when authorizing a large number of subjects. To demonstrate this, we consider the case in which the owner authorizes multiple subjects (more specifically, 1200 subjects consisting of 1000 students and 200 staff members), and compare the total execution time of authorizing all subjects.

Table 4 shows the comparison of the operations needed to authorize all the subjects, and Table 5 shows the average execution time of each operation over 100 executions. Based on this, the total execution time is calculated and compared in Figure 26. We can see that our scheme can authorize the subjects in a significantly shorter time (about 5 times shorter than the DCACI scheme), as expected. This is because the number of times the owner has to attach data to the Tangle is much less in our scheme, which is equal to the number of policies. Since there are two policies in this scenario, the owner attaches encrypted tokens to the Tangle for a mere two times, while in DCACI original copies of tokens are attached to the Tangle for every subject, i.e., 1200 times. Although fetching tokens from the Tangle has to be performed by every subject in our scheme, it takes much less time than attaching tokens to the Tangle, leading to the overall reduction in execution time.

As can be seen here, our scheme provides faster authorization for a large number of subjects than DCACI. In addition, the number of operations object owners have to perform is greatly reduced, alleviating the burden of the owners.

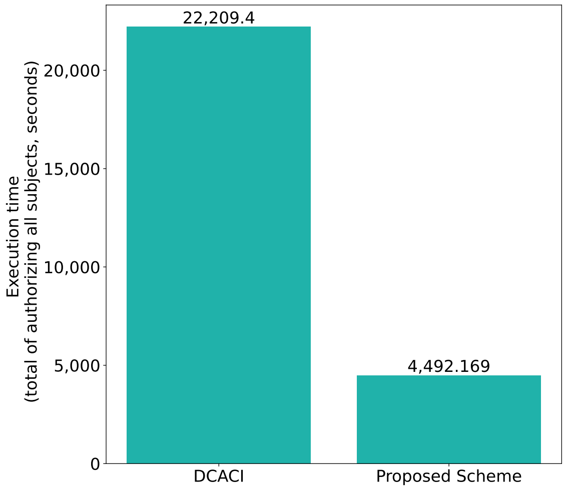


Figure 26. Comparison with DCACI: total execution time of authorizing 1000 students and 200 staff members.

Table 4. Operations to authorize 1000 students and 200 staff members.

Scheme	Operations
DCACI	1000× GrantAccess to student +200× GrantAccess to staff member
Proposed scheme	1× publish student token +1× publish staff token +1000× obtain token by student +200× obtain token by staff member

Table 5. Comparison with DCACI: average execution time of each operation.

Operation	Average Execution Time (s)
GrantAccess to student (DCACI)	18.235
GrantAccess to staff member (DCACI)	19.872
Publish student token (proposed scheme)	32.926
Publish staff token (proposed scheme)	40.443
Obtain student token (proposed scheme)	3.668
Obtain staff token (proposed scheme)	3.754

7. Conclusions

7.1. Summary

In this paper, we have proposed an IOTA-based access control framework in which ABAC and CapBAC are combined by leveraging the CP-ABE technology. Thanks to CP-ABE, our scheme overcomes the drawbacks that exist in the previous framework named DCACI and provides more secure, fine-grained and scalable access control. We have shown the feasibility of our scheme by implementing a practical proof-of-concept prototype using the IOTA Mainnet and IoT devices. We have also evaluated the performance of our scheme in terms of execution time and compared it with DCACI. Experimental results show that our scheme enables object owners to authorize access rights to a large number of subjects in about 5 times shorter time than the DCACI scheme. Furthermore, we have discussed the scalability of our scheme by considering an increased number of attributes and policies. The results show that the execution time for each operation is proportional to the number of attributes involved, and that the number of policies has little effect on the overall execution time for most operations.

7.2. Discussions on Future Work

First, it may not be enough to compare the proposed scheme with DCACI only. In our future work, we will address this issue by comparing our scheme with other access control schemes based on other DLTs, such as Ethereum and Hyperledger Fabric. Second, the main limitation of the proposed method is that the adoption of the CP-ABE scheme introduces a central entity, i.e., the trust authority, which is responsible for secret key generation and distribution. As a result, the proposed scheme is not completely decentralized and some performance loss may occur. In our future work, we will carefully investigate the performance loss caused by the trust authority and also address this issue by adopting a blockchain-based trust authority, such as the one in [54].

Author Contributions: Scheme Design, R.N., Y.Z., M.S. and S.K.; implementation, R.N.; writing—original draft preparation, R.N.; writing—review and editing, Y.Z., M.S. and S.K.; supervision, S.K.; project administration, S.K.; funding acquisition, Y.Z., M.S. and S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Japan Society for the Promotion of Science (JSPS) grant number 19K14983, 19H01103 and Support Center for Advanced Telecommunications Technology (SCAT) Research Foundation. The APC was funded by JSPS grant 19K14983.

Acknowledgments: This work was supported in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant 19K14983, 19H01103 and the Support Center for Advanced Telecommunications (SCAT) Technology Research Foundation.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Gartner Identifies Top 10 Strategic IoT Technologies and Trends. Available online: https://www.gartner.com/en/newsroom/press-releases/2018-11-07-gartner-identifies-top-10-strategic-iot-technologies-and-trends/ (accessed on 25 June 2021).

2. Zikria, Y.B.; Ali, R.; Afzal, M.K.; Kim, S.W. Next-Generation Internet of Things (IoT): Opportunities, Challenges, and Solutions. *Sensors* **2021**, *21*, 1174. [CrossRef]
3. Ben-Daya, M.; Hassini, E.; Bahroun, Z. Internet of things and supply chain management: A literature review. *Int. J. Prod. Res.* **2019**, *57*, 4719–4742. [CrossRef]
4. Qadri, Y.A.; Nauman, A.; Zikria, Y.B.; Vasilakos, A.V.; Kim, S.W. The Future of Healthcare Internet of Things: A Survey of Emerging Technologies. *IEEE Commun. Surv. Tutor.* **2020**. *22*, 1121–1167. [CrossRef]
5. Yang, H.; Kumara, S.; Bukkapatnam, S.T.; Tsung, F. The Internet of things for smart manufacturing: A review. *IISE Trans.* **2019**, *51*, 1190–1216. [CrossRef]
6. HaddadPajouh, H.; Dehghantanha, A.; Parizi, R.M.; Aledhari, M.; Karimipour, H. A survey on Internet of things security: Requirements, challenges, and solutions. *Internet Things* **2019**, *14*, 100129. [CrossRef]
7. Ande, R.; Adebisi, B.; Hammoudeh, M.; Saleem, J. Internet of Things: Evolution and technologies from a security perspective. *Sustain. Cities Soc.* **2020**, *54*, 101728. [CrossRef]
8. Butun, I.; Österberg, P.; Song, H. Security of the Internet of Things: Vulnerabilities, attacks, and countermeasures. *IEEE Commun. Surv. Tutor.* **2019**. *22*, 616–644. [CrossRef]
9. Neshenko, N.; Bou-Harb, E.; Crichigno, J.; Kaddoum, G.; Ghani, N. Demystifying IoT security: An exhaustive survey on IoT vulnerabilities and a first empirical look on Internet-scale IoT exploitations. *IEEE Commun. Surv. Tutor.* **2019**. *21*, 2702–2733. [CrossRef]
10. Xu, R.; Chen, Y.; Blasch, E.; Chen, G. BlendCAC: A blockchain-enabled decentralized capability-based access control for IoTs. In Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018; pp. 1027–1034.
11. Xu, R.; Chen, Y.; Blasch, E.; Chen, G. Exploration of blockchain-enabled decentralized capability-based access control strategy for space situation awareness. *Opt. Eng.* **2019**, *58*, 041609. [CrossRef]
12. Nakamura, Y.; Zhang, Y.; Sasabe, M.; Kasahara, S. Capability-based access control for the Internet of things: An Ethereum blockchain-based scheme. In Proceedings of the IEEE GLOBECOM 2019, Big Island, HI, USA, 9–13 December 2019.
13. Nakamura, Y.; Zhang, Y.; Sasabe, M.; Kasahara, S. Exploiting Smart Contracts for Capability-Based Access Control in the Internet of Things. *Sensors* **2020**, *20*, 1793. [CrossRef]
14. Dukkipati, C.; Zhang, Y.; Cheng, L.C. Decentralized, BlockChain Based Access Control Framework for the Heterogeneous Internet of Things. In Proceedings of the 3rd ACM Workshop on Attribute-Based Access Control, Tempe, AZ, USA, 19–21 March 2018; pp. 61–69.
15. Maesa, D.D.F.; Mori, P.; Ricci, L. A blockchain based approach for the definition of auditable Access Control systems. *Comput. Secur.* **2019**, *84*, 93–119. [CrossRef]
16. Yutaka, M.; Zhang, Y.; Sasabe, M.; Kasahara, S. Using Ethereum blockchain for distributed attribute-based access control in the Internet of things. In Proceedings of the IEEE GLOBECOM 2019, Big Island, HI, USA, 9–13 December 2019.
17. Zhang, Y.; Yutaka, M.; Sasabe, M.; Kasahara, S. Attribute-Based Access Control for Smart Cities: A Smart Contract-Driven Framework. *IEEE Internet Things J.* **2020**, *8*, 6372–6384. [CrossRef]
18. Cruz, J.P.; Kaji, Y.; Yanai, N. RBAC-SC: Role-based access control using smart contract. *IEEE Access* **2018**. *6*, 12240–12251. [CrossRef]
19. Rahman, M.U.; Guidi, B.; Baiardi, F.; Ricci, L. Context-aware and dynamic role-based access control using blockchain. In Proceedings of the International Conference on Advanced Information Networking and Applications, Caserta, Italy, 15–17 April 2020; pp. 1449–1460.
20. Zhang, Y.; Kasahara, S.; Shen, Y.; Jiang, X.; Wan, J. Smart Contract-Based Access Control for the Internet of Things. *IEEE Internet Things J.* **2019**. *6*, 1594–1605. [CrossRef]
21. Sultana, T.; Almogren, A.; Akbar, M.; Zuair, M.; Ullah, I.; Javaid, N. Data sharing system integrating access control mechanism using blockchain-based smart contracts for IoT devices. *Appl. Sci.* **2020**, *10*, 488. [CrossRef]
22. Novo, O. Blockchain meets IoT: An architecture for scalable access management in IoT. *IEEE Internet Things J.* **2018**. *5*, 1184–1195. [CrossRef]
23. Ouaddah, A.; Abou Elkalam, A.; Ait Ouahman, A. FairAccess: A new Blockchain-based access control framework for the Internet of Things. *Secur. Commun. Netw.* **2016**, *9*, 5943–5964. [CrossRef]
24. Maesa, D.D.F.; Mori, P.; Ricci, L. Blockchain based access control. In Proceedings of the IFIP International Conference on Distributed Applications and Interoperable Systems, Neuchâtel, Switzerland, 19–22 June 2017; pp. 206–220.
25. Pinno, O.J.A.; Gregio, A.R.A.; De Bona, L.C. ControlChain: Blockchain as a central enabler for access control authorizations in the IoT. In Proceedings of the IEEE GLOBECOM 2017, Singapore, 4–8 December 2017.
26. Ding, S.; Cao, J.; Li, C.; Fan, K.; Li, H. A novel attribute-based access control scheme using blockchain for IoT. *IEEE Access* **2019**. *7*, 38431–38441. [CrossRef]
27. Zhu, Y.; Qin, Y.; Gan, G.; Shuai, Y.; Chu, W.C.C. TBAC: Transaction-based access control on blockchain for resource sharing with cryptographically decentralized authorization. In Proceedings of the 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), Tokyo, Japan, 23–27 July 2018; Volume 1, pp. 535–544.
28. Bitcoin—Open Source P2P Money. Available online: https://bitcoin.org/en/ (accessed on 25 June 2021).

29. Home | Ethereum. Available online: https://ethereum.org/ (accessed on 25 June 2021).
30. Introduction to Smart Contracts. Available online: https://ethereum.org/en/developers/docs/smart-contracts/ (accessed on 25 June 2021).
31. Blockchain Technology Overview. Available online: https://nvlpubs.nist.gov/nistpubs/ir/2018/NIST.IR.8202.pdf. (accessed on 25 June 2021).
32. Conoscenti, M.; Vetro, A.; De Martin, J.C. Blockchain for the Internet of Things: A systematic literature review. In Proceedings of the 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), Agadir, Morocco, 29 November–2 December 2016.
33. Fully Decentralized IOTA 2.0 Explained in Under 3 Minutes. Available online: https://blog.iota.org/fully-decentralized-iota-explained-in-under-3-minutes/ (accessed on 25 June 2021).
34. Introducing IOTA Access. Available online: https://blog.iota.org/introducing-iota-access-686a2f017ff/ (accessed on 25 June 2021).
35. Pinjala, S.K.; Sivalingam, K.M. DCACI: A Decentralized Lightweight Capability Based Access Control Framework using IOTA for Internet of Things. In Proceedings of the 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), Limerick, Ireland, 15–18 April 2019; pp. 13–18.
36. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-Policy Attribute-Based Encryption. In Proceedings of the IEEE Symposium on Security and Privacy (SP '07), Berkeley, CA, USA, 20–23 May 2007; pp. 321–334.
37. Nakanishi, R.; Zhang, Y.; Sasabe, M.; Kasahara, S. IOTA-Based Access Control Framework for the Internet of Things. In Proceedings of the 2nd Conference on Blockchain Research and Applications for Innovative Networks and Services (BRAINS), Paris, France, 28–30 September 2020; pp. 87–95.
38. Sandhu, R.S.; Samarati, P. Access Control: Principle and Practice. *IEEE Commun. Mag.* **1994**, *32*, 40–48. [CrossRef]
39. Sandhu, R.S.; Coyne, E.J.; Feinstein, H.L.; Youman, C.E. Role-based access control models. *Computer* **1996**, *29*, 38–47. [CrossRef]
40. Hu, V.C.; Kuhn, D.R.; Ferraiolo, D.F.; Voas, J. Attribute-based access control. *Computer* **2015**, *48*, 85–88. [CrossRef]
41. Gusmeroli, S.; Piccione, S.; Rotondi, D. A capability-based security approach to manage access control in the Internet of things. *Math. Comput. Model.* **2013**, *58*, 1189–1205. [CrossRef]
42. Bhatt, S.; Patwa, F.; Sandhu, R. Access control model for AWS Internet of things. In Proceedings of the International Conference on Network and System Security, Helsinki, Finland, 21–23 August 2017; pp. 721–736.
43. Gusmeroli, S.; Piccione, S.; Rotondi, D. IoT access control issues: A capability based approach. In Proceedings of the 2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, Palermo, Italy, 4–6 July 2012; pp. 787–792.
44. Liu, J.; Xiao, Y.; Chen, C.P. Authentication and access control in the Internet of things. In Proceedings of the 2012 32nd International Conference on Distributed Computing Systems Workshops, Macau, China, 18–21 June 2012; pp. 588–592.
45. Ouaddah, A.; Mousannif, H.; Elkalam, A.A.; Ouahman, A.A. Access control in the Internet of Things: Big challenges and new opportunities. *Comput. Netw.* **2017**, *112*, 237–262. [CrossRef]
46. Weber, R.H. Internet of Things—New security and privacy challenges. *Comput. Law Secur. Rev.* **2010**, *26*, 23–30. [CrossRef]
47. Pilkington, M. Blockchain technology: Principles and applications. In *Research Handbook on Digital Transformations*; Edward Elgar Publishing: Cheltenham, UK, 2016.
48. Introducing Masked Authenticated Messaging—IOTA. Available online: https://blog.iota.org/introducing-masked-authenticated-messaging-e55c1822d50e/ (accessed on 25 June 2021).
49. IOTA Networks—IOTA Documentation. Available online: https://docs.iota.org/docs/getting-started/1.1/networks/overview (accessed on 25 June 2021).
50. Masked Authentication Messaging Wrapper for Javascript (Browser and Node). Available online: https://github.com/iotaledger/mam.client.js/ (accessed on 25 June 2021).
51. Zlwen/Cpabe-Java: The Implementation of Ciphertext Policy Attribute Based Encryption in Java. Available online: https://github.com/zlwen/cpabe-java/ (accessed on 25 June 2021).
52. Transaction Fields—IOTA Documentation. Available online: https://docs.iota.org/docs/getting-started/1.1/references/transaction-fields (accessed on 25 June 2021).
53. Sending Transactions—IOTA Documentation. Available online: https://docs.iota.org/docs/getting-started/1.1/first-steps/sending-transactions (accessed on 25 June 2021).
54. Singla, A.; Bertino, E. Blockchain-Based PKI Solutions for IoT. In Proceedings of the 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC), Philadelphia, PA, USA, 18–20 October 2018; pp. 9–15. [CrossRef]