IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 2 March 2023, accepted 9 March 2023, date of publication 14 March 2023, date of current version 17 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3257027

APPLIED RESEARCH

DABAC: Smart Contract-Based Spatio-Temporal Domain Access Control for the Internet of Things

FEIFEI GUO[1], GUOHUA SHEN[1,2,3], ZHIQIU HUANG[1,2,3], YANG YANG[1], MENGNAN CAI[1], AND LINLIN WEI[1]

[1]College of Computer Science and Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China
[2]Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 210093, China
[3]Key Laboratory of Safety-Critical Software, Ministry of Industry and Information Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

Corresponding author: Guohua Shen (ghshen@nuaa.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant U2241216 and Grant 61772270, in part by the National Key Research and Development Program under Grant 2018YFB1003902, and in part by the Opening Fund of Key Laboratory of Civil Aviation Emergency Science and Technology (CAAC) under Grant NJ2022022.

ABSTRACT With the advent of IoT technology, the dynamic nature of IoT devices has introduced new obstacles to access control. It is essential to consider the security requirements of the actual physical environment, rendering the traditional access control approach centered on the information space. In the IoT ecosystem, there are several issues such as the dynamics of devices frequently entering and leaving, the lack of computing and storage capacity, and distributed deployment. To address these challenges, this paper proposes the Domain Attribute Based Access Control(DABAC) that incorporates domain elements to implement the physical location limitation of dynamic devices. Moreover, an intelligent gateway is utilized to divide the physical area and act as a proxy to achieve regional device management, automatic networking of devices in the domain, and the dynamic expansion of the sensor network resulting from device entry or exit. Then, given the distributed deployment of devices, smart contracts are employed to deploy access control mechanisms and construct a trusted environment to mitigate threats such as single points of failure. Finally, the DABAC is implemented on the Ethereum platform, simulating a smart medical situation. The experimental results demonstrate that the proposed solution effectively addresses the problem of access control of device dynamics in an untrusted IoT environment while maintaining system security.

INDEX TERMS Access control, blockchain, dynamics, the Internet of Things, spatio-temporal constraints.

I. INTRODUCTION

The Internet of Things (IoT) refers to a network of physical objects or ''things,'' which are embedded with sensors, software, and other technological elements that enable them to connect and exchange data with other devices and systems over the internet. Owing to the multi-source heterogeneity and openness of the IoT ecosystem, coupled with the diversity and complexity of terminal devices and applications, security concerns associated with IoT systems have become increasingly prominent. A significant risk of unauthorized access and potential compromise of sensitive data stored on IoT devices is an important security challenge that must be addressed. In this context, access control plays a critical role in safeguarding the security and privacy of IoT devices. By preventing unauthorized access by malicious users and improper use by authorized users, access control ensures that the resources of the entire IoT system are used appropriately and reasonably.

The IoT environment is characterized by three features: lightweight, distribution, and dynamics. Firstly, IoT devices typically have limited computing and storage resources dedicated to their operations, making access control an additional burden. Secondly, IoT devices are often deployed in a distributed manner. However, the traditional access control approach relies on a central trusted entity that stores all relevant data and makes decisions based on this information [2]. The distributed object devices provide the central entity with

The associate editor coordinating the review of this manuscript and approving it for publication was Liang-Bi Chen.

the requested data, which may lead to concurrency problems and create a single point of failure. While the distributed design of capability-based access control (CapBAC) mitigates the single point of failure issue, it is insufficient for deploying access control decisions on lightweight devices that cannot guarantee their security and are vulnerable to attacks. In an untrusted environment, distributed CapBAC is insufficient for IoT access control. Thirdly, IoT devices may frequently enter and leave, which is not accounted for by traditional access control approaches that focus primarily on the information space. For the dynamic nature of devices, access control must consider both the information space and physical space limitations.

The integration of blockchain technology with IoT access control provides a reliable and efficient approach to address security concerns. Blockchain technology is a shared, tamper-evident distributed ledger that combines several technologies, including distributed data storage, peer-to-peer transmission, consensus mechanisms, and cryptographic algorithms, among others. Furthermore, blockchain is characterized by decentralization, open transparency, and security. By combining blockchain with IoT access control, the security issues caused by centralized decision-making entities can be eliminated. Instead, blockchain-based access control sets rules that are automatically implemented via smart contracts, providing more dependable and effective control. This approach ensures that access control mechanisms are decentralized, transparent, and tamper-proof while maintaining the privacy and security of IoT devices and data. Therefore, blockchain-based access control is a promising solution for addressing the security challenges in IoT environments.

A novel access control model, DABAC, is proposed, which incorporates domain elements to address the physical location limitation of dynamic IoT devices. This model provides more flexible and accurate control by integrating spatial and temporal information. The main contributions of this paper are summarized as follows:

1) For the lightweight of IoT devices, the intelligent gateway is utilized to realize the connection of Internet of Things devices. Through the gateway, it is simple to establish regional device administration, automatic networking of devices in the domain, and the dynamic expansion of the sensor network.

2) For the distributed deployment of IoT devices, ABAC is combined with blockchain, and the distributed properties of blockchain are leveraged to administer the device by accessing the gateway and to realize automatic decision-making through smart contracts.

3) For the dynamics of IoT devices, the smart gateway is utilized to partition physical regions, convert the physical locations of IoT devices into relative locations, and bind them with spatio-temporal attributes using the blockchain timestamp mechanism.

4) Implemented on Ethereum [23] platform, the solution uses the proof-of-work consensus mechanism to ensure operations such as trusted computing in a distributed untrustworthy IoT environment. To ensure the security of IoT, the attribute-based access control model (ABAC) is implemented through smart contracts to make automatic authorization decisions for access control.

The rest of this paper is organized as follows. Section II focuses on the introduction of relevant technologies and a review of related work. Section III describes the system architecture, including the system model, a description of the smart contract content, and the workflow. Section IV describes a use case. Section V a smart healthcare usage scenario, implements smart contracts on the Ethereum platform for access control experiments, and assesses the performance of these smart contracts.

II. BACKGROUND AND RELATED WORK

A. ATTRIBUTE BASED ACCESS CONTROL

Access control is a crucial component of information security that involves managing who has access to resources and what actions they are permitted to perform. Access control can be defined as a triple AC:=<S, O, P>, where S denotes the subject, an entity that issues a request or requirement, usually a person, process, or device, etc.; O denotes the object, a passive entity that receives access from other entities; P denotes policies for subject-to-object access, which directly defines the behavior and constraints of subjects' action on objects. ABAC is a logical access control methodology where authorization to perform a set of operations is determined by evaluating the attributes associated with the subject, object, requested operations, and, in some cases, environmental conditions against policy, rules, or relationships that describe the allowable operations for a given set of attributes [3]. The main idea of role-based access control [4] (RBAC) is that access rights are associated with roles. A role is merely an attribute in ABAC, therefore RBAC can be viewed as an extension of ABAC. However, in the IoT environment where a large number of roles are involved, RBAC suffers from role explosion and is incapable of meeting the fine-grained, multi-level access control requirements. ABAC, however, supports a context-aware environment by combining a variety of richer attribute information to form access control conditions and is no longer simply user-centric, allowing it to adapt flexibly to varied resource access circumstances. Figure 1 shows a fundamental ABAC access situation.

B. BLOCKCHAIN TECHNOLOGY

Originating from Bitcoin [5], blockchain is a shared, tamper-evident ledger designed to facilitate the process of recording transactions and tracking assets in a business network. A blockchain consists of numerous interconnected blocks, with each block including a block header and a block body. As illustrated in Figure 2, the block header stores the characteristic values of the current block, such as the hash of the previous block, the hash of the current block, a random number, a timestamp, and a Merkle root, whereas the block

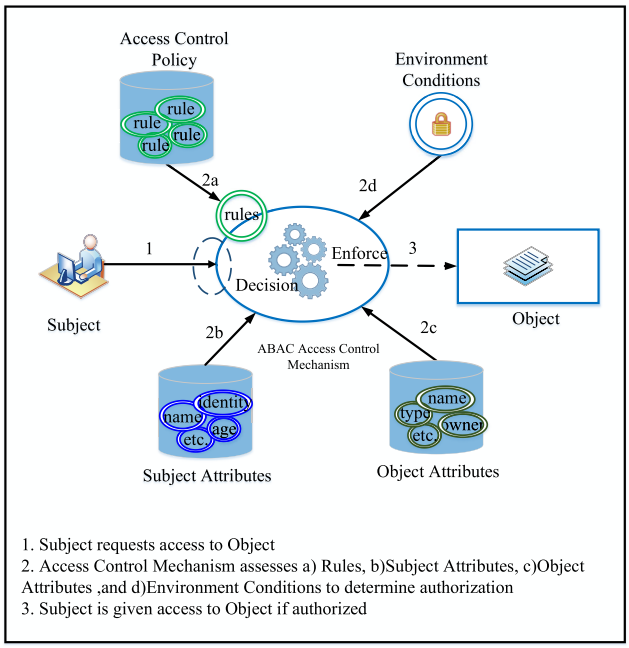


FIGURE 1. Basic ABAC Access Control Scenario.

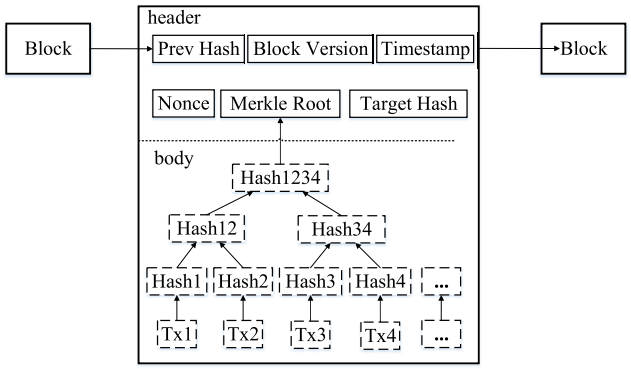


FIGURE 2. Block Structure.

body stores all transaction data and is represented as a Merkle tree.

Moreover, blockchain is decentralized based on a P2P(peer-to-peer) network. All participating nodes manage the network collaboratively by storing and validating new transactions and blocks, using consensus mechanisms to ensure data consistency and tamper resistance. Ethereum is an open-source blockchain platform with a Turing-complete scripting language and the Proof-of-Work consensus process. Smart contracts are executable programs on the Ethereum network, in which Ethereum users deploy business logic that is automatically invoked and executed by the Ethereum Virtual Machine (EVM) [10]. Smart contracts enable complex business logic which expands the scope of blockchain applications and enables access control, supply chain management, etc. to begin integrating with blockchain.

C. RELATED WORK

In previous studies, the researchers' proposed access control scheme was primarily intended for centralized environments. The conventional access control scheme is composed of a central entity that maintains all information and makes all choices based on the stored information. Hemdi and Deters [6] used the server to apply an attribute-based access control mechanism and verified the access request through the cloud server. Yavari et al. [7] proposed a lightweight but highly scalable data obfuscation that limits access based on roles and collects IoT device data through server authorization. Das and Namasudra [9] proposed an ECC-based CP-ABE technique for achieving fine-grained access control to the data or resources of IoT devices. However, most of the CP-ABE schemes use bilinear pairing operations for internal working, which is expensive for any resource constraint device. Simultaneously, the decryption process must rely on external entities to cut costs for end users and enhance certain possible threats.

The above solution uses a centralized decision-making method to solve the problem of lightweight equipment in the Internet of Things environment, however, it introduces a single point of failure and ignores the distribution of equipment deployed in the Internet of Things. Some scholars then recommended using a distributed way to tackle this issue. The distributed CapABAC access control model [12] was proposed, and an IoT-device-based distributed platform was used to implement access control. However, the computing and storage capacities of IoT devices are limited, making them vulnerable to malicious attacks. Pure IoT devices cannot be used as a secure entity for decision-making. Consequently, IoT access control needs a fully dependable third party to supply storage and computing capabilities.

As a trusted third party, the blockchain was initially combined with access control in 2016, and the solution was designed based on Bitcoin. Ouaddah et al. [14] proposed FairAccess applied to the Internet of Things scenario utilizing Bitcoin to give, obtain, delegate, and cancel access rights. Maesa et al. [19] developed a system for the release of resource access rights based on blockchain technology that enabled the decentralized transfer of resource access permits between users. On the blockchain, the exchange of policies and permits is open and transparent. And a deployment example based on the XACML technique is implemented using Bitcoin.

Nevertheless, each of the aforementioned research views the blockchain as a database for storing access control policies. With the arrival of blockchain 2.0, the creation of smart contracts has enhanced the blockchain's applications. The blockchain is no longer used solely as a distributed database, but also facilitates trustworthy computing. Cruz et al. [13] proposed a role-based access control scheme (RBAC-SC), implementing a platform for cross-organizational role utilization using Ethereum's smart contract technology. Zhang et al. [18] proposed an ACL access control framework based on smart contracts, which consists of multiple access control contracts (ACCs), a judgment contract (JC), and a registration contract (RC), which need to be The client specifies the ACL, therefore the storage cost on the

chain will increase linearly with the increase of the number of clients, and the scalability and flexibility are severely limited in the large-scale IoT environment. Xu et al. [20] proposed the BlendCAC scheme, which is a decentralized, federated capabilities-based access control mechanism, and designed a federated capabilities-based delegation model (FCDM) to support hierarchical and multi-hop delegation. The mechanism of delegated authorization and revocation is discussed. Smart contracts are utilized to register, propagate, and revoke access authorization as part of a strong identity-based capability token management strategy. However, none of the above studies considered the dynamics of devices in the IoT environment, ignored the spatio-temporal variables in the access control process, and tended to pre-defined access control list forms for policy administration, which lacked scalability and flexibility.

Yavari et al. [7] combined context with digital watermarking to support the contextualization of IoT data for IoT applications. Liu et al. [8] proposed a resource access control model based on RBAC to solve the cross-service, cross-organization, and cross-region resource sharing of the Industrial Internet of Things. To achieve automated authorization, the RBAC management system calculates the authorization routing optimization problem automatically. Xin-Fang et al. [11] proposed an indoor location-based access control system, which achieves location acquisition based on embedded active tags and designed an authentication protocol to ensure tag anonymity and protect data confidentiality. However, the application scenarios are limited, not Applicable to dynamic distributed scenarios. Grummt and Muller [15] proposed a new rule-based, context-aware policy language for data access control. Kim et al. [16] proposed an architecture that seamlessly integrates heterogeneous protocols and vendor-specific devices and services to solve the problem of inconsistent communication standards for smart home devices, and simultaneously designed a new smart home-specific access control model. Rivera et al. [17] proposed the application of User Managed Access (UMA) to provide a unified access control model for heterogeneous hybrid architectures of IoT devices and intelligent agents.

As stated in Table 1, the design of an IoT access control solution must simultaneously solve the following issues: First, due to the lightweight of IoT devices, certain devices have the inadequate computational capacity and cannot support certain computer functions. Second, Second, given the widespread deployment of IoT devices, the number of distributedly deployed devices is enormous. Third, given the dynamics of IoT devices, the design of access control mechanisms should consider spatio-temporal restrictions.

III. SYSTEM ARCHITECTURE

This section discusses the system architecture and workflow of the proposed attribute-based access control system that combines domain elements with blockchain smart contracts, as well as how to automate the decision-making process using smart contracts and the pertinent algorithms.

A. SYSTEM MODEL

The access control model with the introduction of domain elements DABAC: = <S, O, SA, OA, P>, SA: =<D, T, R, Others>, OA: =<D, T, Others>. In the proposed DABAC model, there are two types of attributes: subject attributes (SA) and object attributes (OA). SA includes domain attribute (D), time attribute (T), role (R), and other attributes, while OA includes domain attribute (D), time attribute (T), and other attributes. The policy set (P) determines the access control rules and relationships for the subject, object, and their attributes. Specifically, the domain attribute (D) refers to the attribute that distinguishes objects or subjects in different domains. The time attribute (T) refers to the attribute that represents the time dimension and allows access control policies to be based on the temporal context. The role (R) is an attribute that describes the position or function of the subject in the organization or system. Other attributes refer to additional attributes that can be used to describe the subject or object. The DABAC model is a spatio-temporal domain attribute-based access control model that considers the physical location and time of IoT devices in the access control decision-making process. By incorporating domain and time attributes into access control policies, the DABAC model can provide fine-grained access control in a dynamic and changing IoT environment.

To expand on the system model, DABAC utilizes the blockchain network to provide a tamper-proof and decentralized storage of subject and object attributes. The subject and object attributes are registered on the blockchain using smart contracts, which act as programmable self-executing contracts that enable the automatic enforcement of access control policies. The smart contracts contain the rules for authorization decision-making, and they are triggered when a subject requests access to an object.

The IoT network consists of intelligent gateways that act as intermediaries between the IoT devices and the blockchain network. The intelligent gateway manages the IoT devices within its domain and communicates with the blockchain network to enforce access control policies. The gateway also partitions the physical regions, converts physical locations to relative locations, and binds them with spatio-temporal attributes using the blockchain timestamp mechanism.

The DABAC system model ensures the security and integrity of access control policies by utilizing blockchain's distributed and tamper-proof nature, providing a decentralized and trustworthy environment for IoT devices. It also provides flexibility in attribute-based access control by allowing for rich attribute information and context-aware access control.

i. Subject: The entity that issues access control requests or requirements. It might be a person or a process, service, or device initiated by the user, but in this article, it is primarily a user.
ii. Object: A manipulable information, resource, or object; an entity that takes requests for access control from

TABLE 1. Comparison with existing solutions.

Scheme	Blockchain	Decentralization	Security	Temporal Constraint	Spatial Constraint
Hemdi [6]	×	×	×	×	×
Yavari [7]	×	×	×	✓	✓
Liu [8]	×	×	×	×	✓
Zhang [11]	×	×	×	×	✓
Hernández-Ramos [12]	×	✓	×	×	×
Cruz [13]	✓	✓	✓	×	×
Ouaddah [14]	✓	✓	✓	×	×
Zhang [18]	✓	✓	✓	×	×
Ours	✓	✓	✓	✓	✓

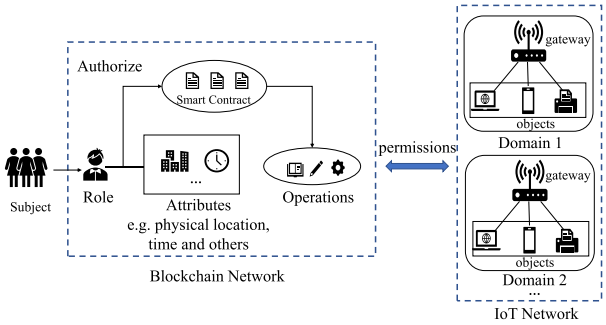


FIGURE 3. The system Framework of the proposed DABAC.

other entities. It primarily refers to IoT devices and the resource data they carry, such.

iii. IoT gateway: Use the Internet of Things gateway to connect various Internet of Things devices, solve the protocol conversion between different types of device communication, enable various heterogeneous devices to realize interconnection, and manage the device nodes at the bottom of the Internet of Things.

iv. Blockchain Network: The P2P network composed of various blockchain nodes stores and maintains access control policy sets and subject and object attributes, implements access control decisions through smart contracts, and provides security features including tamper-proof and traceability.

v. IoT Network: The intelligent gateway divides the physical region, and IoT devices (objects) in distinct areas are connected to their respective gateways to realize networking and build an IoT network.

B. SMART CONTRACT

Figure 4 depicts the contents of the three primary contracts designed in DABAC: User Management Contract (UMC), Device Management Contract (DMC), and Access Control Contract (ACC).

The purpose of User Management Contract is to manage users, including user registration, maintenance of user information, logout, etc. The purpose of Device Management Contract is to manage devices, including device registration, device information maintenance, logout of devices, etc. The purpose of Access Control Contract is to manage access control policies and verify access requests by offering functions such as adding resources, adding access control policies for corresponding resources, and making access control decisions. Table 2 lists the primary responsibilities of the three contracts.

TABLE 2. Main function functions in smart contracts.

Contract	Function	Description
UMC	addUser()	Register a new user
	updateUser()	Update users information
	removeUser()	Logout a user
	updateDomain()	Update users physical location
	addAttr()	Add users attributes and values
	updateAttr()	Update users attributes and values
	removeAttr()	Delete users attributes and values
DMC	addDevice()	Register a new device
	updateDevice()	Update devices information
	removeDevice()	Logout a device
	updateDomain()	Update devices physical location
	addAttr()	Add devices attributes and values
	updateAttr()	Update devices attributes and values
	removeAttr()	Delete devices attributes and values
ACC	addResource()	Add resources to the device
	addPolicy()	Add resource access control policies
	updatePolicy()	Update resource access control policies
	removePolicy()	Delete resource access control policies
	accessControl()	Make decisions and return results

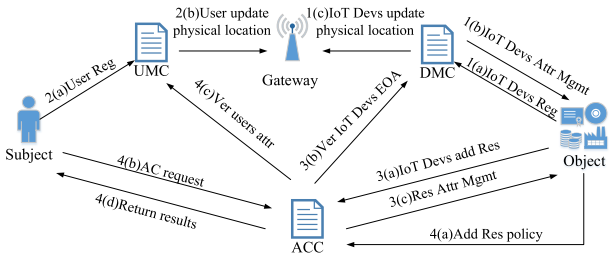


FIGURE 4. The key processes in the access control process.

C. MAIN PROCESS

Figure 4 depicts the unique flow of the access control system, which is supported by UMC, DMC, and ACC contracts, and the essential steps of the flow are described below.

Step 1 User Management a) New users register a new user account before use, as in Algorithm 1. b) Manage user attributes, including adding, updating, and removing attributes. c) After the user device is connected to the gateway, the domain attributes of the user are updated to the gateway domain, as in Algorithm 3.

Step 2 Device Management a) Registering a new IoT device when adding an IoT device, as in Algorithm 2. b) Manage IoT device attributes, including adding, updating, and removing attributes. c) Update the domain attribute of the device to the gateway domain when the IoT device is connected to the gateway, as in Algorithm 3.

Step 3 Resource management a) Add the resources owned by the IoT device to the corresponding device account, as in Algorithm 4. b) Check if the device is registered. c) Manage the resource attributes, including adding, updating, and removing attributes. d) Set access control policies for the corresponding resources.

Step 4 Access control a) The user initiates an access control request. b) Verifying that the user attributes and spatio-temporal constraints are met, as in Algorithm 5. c) Return the result of the access control decision.

Algorithm 1 User Management

Input: address of user A and attributes $Attr$

```
mapping(address =>attributes) users;
function addUser(A, Attr)
    require(msg.sender == owner);
    users[A] = Attr
end function
function deleteUser(A)
    require(msg.sender == owner);
    delete users[A];
end function
```

Algorithm 2 Device and Resource Management

Input: address of device D, devices' resource R and attributes $Attr$

```
mapping(address =>attributes) devices
mapping(address =>resource) deviceresource
function addDevice(D, Attr)
    require(msg.sender == owner)
    devices[D] = Attr
end function
function deleteDevice(D)
    require(msg.sender == owner)
    delete devices[D]
end function
function addResource(D, R)
    require(msg.sender == owner)
    deviceresource[D] = R
end function
function deleteResource(D, R)
    require(msg.sender == owner)
    delete deviceresource[D].R
end function
```

IV. USE CASE

As depicted in Figure 6, this section offers a use case involving smart healthcare as an experimental scenario. IoT

Algorithm 3 Update Domain

Input: the Externally Owned Account of user or device EOA and physical location Dom

```
function updateDomain(EOA, Dom)
    EOA.location = Dom
    time = block.timestamp
end function
```

Algorithm 4 Policy Management

Input: target resource R and access policy P

```
mapping(address =>mapping(resource => action))policies
function addPolicy(R, P)
    require(msg.sender == owner)
    policies[R] = P
end function
function updatePolicy(R, P)
    require(msg.sender == owner)
    policies[R] = P
end function
function deletePolicy(R, P)
    require(msg.sender == owner)
    delete policies[R]
end function
```

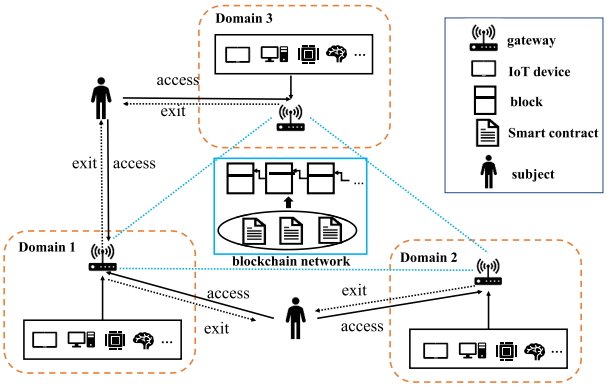


FIGURE 5. Experimental scenarios for smart healthcare.

devices are connected to the gateway, which serves as a proxy for the devices while managing the IoT devices. Through blockchain, a trusted execution environment is created, and access control mechanisms are established utilizing smart contracts to provide safe and secure permission management of the system.

Table 3 provides instances of people and devices participating in the smart healthcare scenario. Alice and Bob are users, and their attributes include age, role, department affiliation, registration time, and physical location domain, among others. Computer and Bracelet are IoT devices with attributes such as registration time, physical location domain, contained resources, etc.

Table 4 presents several pertinent access control policies that describe the constraints that must be met for a subject to initiate an access control request to an object resource, such

Algorithm 5 Access Control

Input: target resource R, user attribute set S
Output: result, error message

```
mapping(address =>mapping(resource => action))policies
function accessControl(R, S)
    assert(DomainCheck(R, S))
    assert(TimeCheck(R, S))
    assert(PolicyCheck(R, S))
    emit ResultEvent(msg.sender, R, "allow/deny")
end function
function DomainCheck(R, S)
    P = policies[R]
    if P.domain is satisfied by S then
        return true
    else
        return false
    end if
end function
function TimeCheck(R, S)
    P = policies[R]
    if P.time is satisfied by request time then
        return true
    else
        return false
    end if
end function
function PolicyCheck(R, S)
    P = policies[R]
    if P.attributes is satisfied by S then
        return true
    else
        return false
    end if
end function
```

TABLE 3. Selected subjects and objects in the experiment.

Alice	Bob	Computer	Bracelet
Id:EOAdoca	Id:EOAnurb	Id:EOAcom	Id:EOAbrac
Type:subject	Type:subject	Type:object	Type:object
Role:docter	Role:nurse	Domain:	Domain:
Gender:male	Gender:male	(area1,2019.	(area2,2019.
Age:35	Age:50	5.2.10:00)	5.2.10:00)
Dep:Surgery	Dep:Surgery	Reg:2019.5.2	Reg:2019.5.2
Domain:	Domain:	Validity:5yr	Validity:3yr
(area1,2022.	(area1,2022.	Res:file...	Res:heart...
6.8.10:00)	6.8.10:00)		
Reg:2020.5.2	Reg:2019.6.17		
Validity:3yr	Validity:5yr		

as physical location constraints, time constraints, etc., as well as the subject's specific permissions over the object.

V. IMPLEMENTATION OF THE PROPOSED SOLUTION

A. EXPERIMENTAL ENVIRONMENT

The IoT gateway plays a crucial role in connecting IoT devices and enabling communication between them. It acts as a central hub that bridges different communication protocols used by IoT devices. For example, IoT perception layer devices can use various short or long-distance wireless transmission modes such as Bluetooth, Zigbee, Z-wave, LTE, LTE-M, and WIFI to access the IoT gateway. The gateway then converts the communication protocols to allow access to the internet. While the HTTP protocol is commonly used on the mobile internet, many other protocols are more suitable for IoT applications, such as MQTT, DDS, AMQP, XMPP, JMS, REST, and CoAP. The application layer leverages the above-mentioned various communication protocols to perform data interaction and communication with the perception layer devices. In this paper, the Raspberry Pi is used as the IoT gateway, and other devices use WIFI to access and communicate using the HTTP protocol. However, it's worth noting that other application layer protocols, such as MQTT, can also be set up and used.

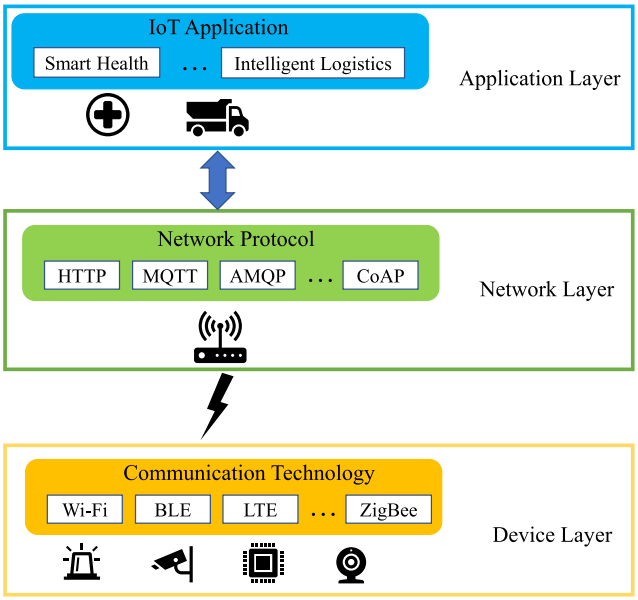


FIGURE 6. The architecture of Internet of Things.

Figure 7 depicts the hardware used in the experiments, which consists of a desktop computer, a laptop, and two single board computers. The desktop computer acts as an IoT device, the laptop as a user device, and the single board machines as IoT gateways. Table 5 lists the specific hardware specifications. To deploy the DABAC architecture, Ethereum nodes are installed on several devices using the geth client [21] to establish an Ethereum blockchain network. Using Truffle [24] framework, smart contracts are deployed while all Ethereum nodes are linked to the Ropsten test network. Using web3.js package [22], JavaScript scripts are constructed that may be used to send transactions and see the subject and object side access results.

Experiments are conducted using attributes similar to those in Table 3 to establish subject and object attributes and access control policies comparable to those in Table 4. According to

TABLE 4. Some of the access control policies used in the experiment.

Number	Content	Spatial Constraints	Temporal Constraints	Actions
P1	During working hours in the internal medicine area 1 doctor a has the right to read and write patient information on the departmental computer.	$user.spatial$ = area 1	$user.temporal$ in (9:00-17:00)	read, write
P2	In area 3 doctor a can only read patient information on the computer.	$user.spatial$ = area 3	none	read
P3	Doctor a are authorized to read patient health information on the bracelet during working hours.	none	$user.temporal$ in (9:00-17:00)	read
P4	During working hours doctor b can operate the bracelet in area 2 such as read, write and execute.	$user.spatial$ = area 2	$user.temporal$ in (9:00-17:00)	read, write, execute

FIGURE 7. Hardware used in the experiment.

TABLE 5. Specifications of the experimental environment.

Number	Hardware	CPU	Memory	Hard Disk
①	Computer	Intel Core i7-6700, 3.4GHz	8GB	1TB
②	Laptop	Intel Core i5-1135G7, 2.40GHz	16GB	256GB
③	Raspberry Pi 3 Model B	ARM Cortex A53, 1.2GHz	1GB	16GB
④	Raspberry Pi Zero W	ARM11 1GHz	512MB	16GB

Tables 3 and 4, a subject can only read, write, and perform actions on an object.

B. EXPERIMENTAL RESULT

In the system, there are two types of return results: permission consent and authorization denial. The denial of authorization contains three cases: (1) the resource does not provide remote access control outside the physical scope specified by the access control policy; (2) outside the access time specified by the access control policy; and (3) other attributes of the subject do not satisfy the requirements of the object access control policy. Figures 8, 9, and 10 depict cases 1, 2, and 3 of denial of authorization, while Figure 11 depicts the successful authorization case.

C. PERFORMANCE ANALYSIS

1) CONTRACT PERFORMANCE ANALYSIS

This section analyzes the performance of the system, including the amount of gas consumed by the deployment of the three smart contracts, UMC, DMC, and ACC, as shown in Table 6, and the amount of gas consumed by the main functions of the contracts when they are transacted, as shown in Table 7.


```
address: '0xb4cA921861d7508c72537483B751375C7755967D',
blockHash: '0x4739a5694044bed953d6c2a913b67efda7dcb15a5d6ad280ba8767d56a30b781',
blockNumber: 12978199,
logIndex: 11,
removed: false,
transactionHash: '0xd74f9d9c94fb540838720020400e10ba66cd12efe47de36051e27cb9c9ab9b0d',
transactionIndex: 12,
id: 'log_59fd1146',
event: 'ReturnAccessResult',
args: Result {
  '0': '0xA5FFe4E97581DD7C05d73f121B59166d8B0849A6',
  '1': false,
  '2': 'Outside the physical scope of access control!',
  __length__: 3,
  _from: '0xA5FFe4E97581DD7C05d73f121B59166d8B0849A6',
  _result: false,
  _errmsg: 'Outside the physical scope of access control!'
```


FIGURE 8. Refusal: failure to meet spatial constraints.


```
address: '0xb4cA921861d7508c72537483B751375C7755967D',
blockHash: '0x936cde5f0529f2d2f26bc575e67a201081e841a8be0710af2c3587986aaf0133',
blockNumber: 12978257,
logIndex: 10,
removed: false,
transactionHash: '0x2577c85a19551803e27c9a74983109eba246db8b51194dc271cc1e6289808ae8',
transactionIndex: 16,
id: 'log_3f06fa94',
event: 'ReturnAccessResult',
args: Result {
  '0': '0xA5FFe4E97581DD7C05d73f121B59166d8B0849A6',
  '1': false,
  '2': 'Not within the specified access control time!',
  __length__: 3,
  _from: '0xA5FFe4E97581DD7C05d73f121B59166d8B0849A6',
  _result: false,
  _errmsg: 'Not within the specified access control time!'
```


FIGURE 9. Refusal: failure to meet temporal constraints.


```
address: '0xb4cA921861d7508c72537483B751375C7755967D',
blockHash: '0x8f7c2907f6cde6d4bb38d135ef89026d62b2728504e7725d174cb08edb380570',
blockNumber: 12978283,
logIndex: 1,
removed: false,
transactionHash: '0xb0464dd065e243a4e6227f78bec5a08d11ad6be5d2e3ce14433151cc746be3ae',
transactionIndex: 14,
id: 'log_24912fe3',
event: 'ReturnAccessResult',
args: Result {
  '0': '0xA5FFe4E97581DD7C05d73f121B59166d8B0849A6',
  '1': false,
  '2': 'The identity does not meet the access control policy!',
  __length__: 3,
  _from: '0xA5FFe4E97581DD7C05d73f121B59166d8B0849A6',
  _result: false,
  _errmsg: 'The identity does not meet the access control policy!'
```


FIGURE 10. Refusal: failure to meet other attributes.


```
address: '0xb4cA921861d7508c72537483B751375C7755967D',
blockHash: '0x4739a5694044bed953d6c2a913b67efda7dcb15a5d6ad280ba8767d56a30b781',
blockNumber: 12978199,
logIndex: 13,
removed: false,
transactionHash: '0xd74f9d9c94fb540838720020400e10ba66cd12efe47de36051e27cb9c9ab9b0d',
transactionIndex: 12,
id: 'log_cac9102b',
event: 'ReturnAccessResult',
args: Result {
  '0': '0xA5FFe4E97581DD7C05d73f121B59166d8B0849A6',
  '1': true,
  '2': 'Access authorized!',
  __length__: 3,
  _from: '0xA5FFe4E97581DD7C05d73f121B59166d8B0849A6',
  _result: true,
  _errmsg: 'Access authorized!'
```


FIGURE 11. Authorization successful.

TABLE 6. The cost of contracts.

Contract	Transaction Cost (gas)
UMC	1648152
DMC	1440233
ACC	2060477

TABLE 7. The cost and TPS of contract traction.

Contract	Transaction Cost (gas)
addUser	230307
updateDomain	93056
addDevice	184467
addPolicy	163414
accessControl	67031

TABLE 8. Comparison with metrics from other jobs.

Scheme	Blockchain	Model	Flexibility	Dynamics
Cruz [13]	✓	RBAC	×	×
Zhang [18]	✓	RBAC	×	×
DABAC	✓	RBAC	✓	✓

2) THROUGHPUT ANALYSIS

Throughput, as a reflection of the transaction processing efficiency of the blockchain platform, is defined as Transactions Per Second (TPS), as shown in Equation (1):

$$TPS = \frac{Tractions}{T_{end} - T_{begin}} \tag{1}$$

Among them, *Transactions* represents the transaction volume processed by the blockchain platform within the period from *Tbegin* and *Tend*. In practical applications, the current average throughput of Ethereum is around 15TPS, meaning that 15 transactions are processed per second. The maximum gas limit of the Ethereum block is currently set at 10,000,000 gas, and each block is generated in 15 seconds. Based on these data, the maximum throughput of the system on the Ethereum main chain is estimated to be 2∼10 tps/s.

Hyperledger Fabric, on the other hand, is a permissioned chain that requires participants to be known. It employs the PBFT [25] consensus algorithm and has a high transaction throughput of 100∼300tps/s in the actual application of this paper. Private chains, also permissioned chains, have a small number of participating nodes and a high degree of trust. These chains function similarly to a distributed data ledger with built-in encryption security, where each node is not required to validate transactions. As a result, private chains have a faster transaction speed and lower transaction cost, and their transaction throughput can approach the infinite transaction processing speed of existing databases. In this study, an Ethereum private chain is used for testing, with a relatively low mining difficulty, and the system throughput is observed to be in the range of 400∼1500tps/s. Figure 12 shows the comparison of system throughput in different blockchains.

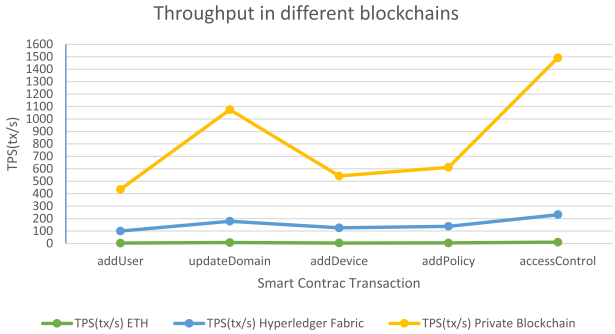


FIGURE 12. System throughput in different blockchains.

3) COMPARISON WITH EXISTING WORK

As shown in Table 8, the above two schemes are based on blockchain access control. The SC-RBAC scheme, as proposed by Cruz et al. [13], utilizes the Role-Based Access Control (RBAC) model for access control and creates a smart contract. However, this approach requires pre-defined role sets and the assignment of roles to users. In complex scenarios, such as a hospital, this can lead to the issue of role eruptions, where a doctor may need to be subdivided into different departments with varying levels of authority, reducing the flexibility of the scheme. In another approach, Zhang et al. [18] proposed an Access Control List (ACL) access control framework based on smart contracts. This approach necessitates the establishment of pre-defined access control lists and consists of several access control contracts (ACCs), a judgment contract (JC), and a registration contract (RC). While this approach enables the assignment of permissions to users, it results in increased storage costs on the blockchain, which scales linearly with the number of clients. As a result, the scalability and flexibility of this scheme are constrained in a large-scale IoT environment.

Furthermore, the aforementioned SC-RBAC and ACL access control schemes do not account for the dynamic nature of nodes in the Internet of Things environment. This includes the dynamic access problem of nodes, node mobility, and real-time modifications of access data items. The constant dynamism in this environment makes it impossible to predict all user information in advance or to accurately understand the structure of users and permissions. Presetting the corresponding relationship between users and permissions in advance becomes unfeasible.

In the context of the intelligent medical care use case described in Section IV, a surgeon named Alice is allowed to operate on surgical patients' diagnoses and treatment information. However, the existing RBAC scheme proposed by Cruz requires Alice to be assigned the role of a surgeon, which is inflexible as it may result in a role explosion due to numerous other departments and workers in the hospital. The scheme proposed by Zhang, on the other hand, relies on the predefinition of access control lists, which can lead to an excessively large access control list, making it difficult to add

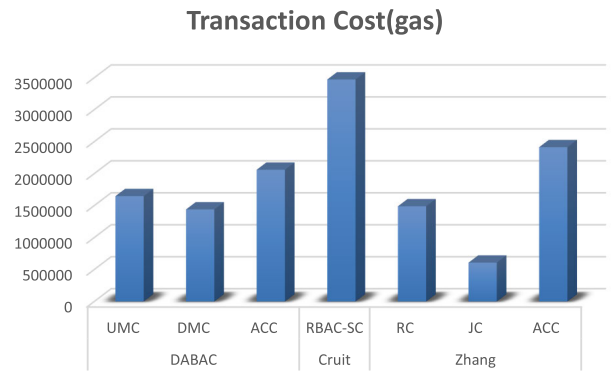


FIGURE 13. Smart contract deployment cost comparison.

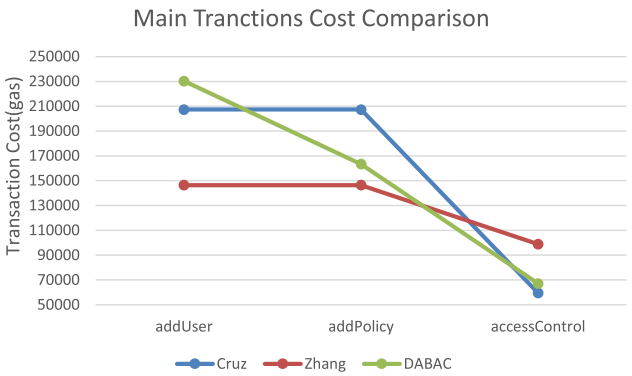


FIGURE 14. Main transactions cost comparison.

new users and not scalable. Furthermore, both schemes do not account for the dynamic nature of user attributes, such as a change in authority when Alice is not in the hospital or at work.

To address these issues, this paper proposes a novel access control scheme based on smart contracts that leverages the inherent attributes of users, such as the department they belong to and their role in the hospital. The proposed scheme matches the user's location and access time to the access control policy set and grants them permission to access surgical patient diagnosis and treatment information. This approach is more flexible, scalable, and dynamic, as it eliminates the need for a predefined set of roles or access control lists, enabling new users to be added easily without having to modify the access control policy set.

In Figure 14, we present a comparison of the resource consumption of the three schemes for deploying smart contracts. It can be observed that Cruz consumes the most resources due to its concentration of all functionalities in a single contract. However, there is no significant difference in the consumption of resources between the other two schemes.

Figure 14 illustrates a comparison of the primary transactions involved in the three schemes. As DABAC introduces the concept of temporal domain and increases the temporal attribute, the consumption of resources increases when adding users. Nonetheless, there is no significant difference between the resource consumption of DABAC and the other two schemes when making authorization decisions.

In summary, this paper uses the ABAC model for access control and authorizes it according to the attributes of users and devices without prior assignment of permissions, which greatly improves flexibility. This work presents the concept of a domain, adds spatio-temporal properties to the system, and provides fine-grained access control to address the dynamic challenge of IoT devices.

D. DISCUSSION

We give a brief discussion on the proposed smart contract-based access control framework as follows.

Distribution Traditional access control frameworks are usually centralized, which lacks transparency. In the framework proposed in this paper, IoT gateways are employed as blockchain nodes to establish a P2P blockchain network in which each node can originate messages and exchange data with surrounding nodes. Each node stores the access control policy of the underlying device, makes access control decisions, and broadcasts the results to the blockchain network. IoT devices engage in the blockchain through access gateways, and the distributed deployment of IoT devices is accomplished through the distributed deployment of gateways. The blockchain acts as a decentralized database that records the input and output of each transaction, hence facilitating the monitoring of asset amount changes and transactional patterns. The immutable data storage approach of blockchain assures that users cannot earn unauthorized access control rights by tampering with their information.

Dynamics In this study, IoT devices are interconnected through smart gateways that are utilized to partition physical areas and serve as proxies. When IoT devices approach gateways, their domain attributes are updated to reflect their most recent physical location. Considering the issue of lightweight, inbuilt GPS locating devices are unnecessary. Using the time-stamping mechanism of blockchain, access control based on spatio-temporal limitations is implemented to address the issue of dynamics in the IoT environment.

Reliability Blockchain is a decentralized system in which all participants (nodes) participate under a consensus process to create a chronological and reliable public ledger and maintain a complete data backup. To prevent the issue of a single point of failure, the data content must be the same at each node. In addition, the blockchain network employs the proof-of-work (PoW) method for consensus. To preserve blockchain transaction data and maintain consistency (consensus) over time, PoW stipulates that all network nodes participating in the blockchain compete for record-keeping. A successful attack on it would require 51% arithmetic power and cost a lot. Therefore, the system has good reliability.

As the number of nodes in the Internet of Things (IoT) environment continues to grow, the application of blockchain in IoT access control presents a series of challenges that need to be addressed. In particular, the massive number of end

nodes in the IoT environment will be the focus of our future research. Specifically, three key issues need to be considered. Firstly, managing the vast number of end nodes in IoT will require careful consideration. Secondly, the storage pressure brought by the exponential growth of nodes on the blockchain is a critical issue that must be addressed. The access control policy and related transactions will increase in volume with the increase of nodes, necessitating scalable and efficient storage solutions. Finally, the impact of the massive number of end nodes on access control performance must be examined to ensure that the system remains functional and efficient. As such, our future research will be focused on developing innovative approaches to address these challenges, ensuring the continued viability and effectiveness of blockchain-based access control in IoT.

VI. CONCLUSION

This paper presents DABAC, an access control scheme that integrates Attribute-Based Access Control (ABAC) with blockchain technology to address the security challenges of access control in an open and untrusted Internet of Things (IoT) environment. By combining spatial-temporal domain components with ABAC, DABAC can provide spatio-temporal limitations that are based on the dynamics of IoT devices. As IoT devices are widely deployed and lightweight, gateways are employed to link and proxy devices to map their physical positions to relative locations, enabling device management. The solution uses smart contracts, namely UMC, DMC, and ACC, to make decisions. The proposed access control scheme is simulated in an intelligent healthcare scenario, and relevant experiments are conducted on the Ethereum test network Ropsten to demonstrate its feasibility in addressing access control in an IoT environment that is both open and dynamic. In summary, DABAC provides a flexible and transparent approach to access control in an open and untrusted IoT environment by combining ABAC with blockchain technology and incorporating spatio-temporal domain components. The results of the experiments conducted in an intelligent healthcare scenario demonstrate the feasibility of DABAC in addressing access control in such an environment.

The proposed solution in this paper is built upon blockchain technology. If the consensus protocol of the public chain is adopted, certain limitations are introduced. Specifically, all node participants in the network must process all transactions, resulting in two limitations. Firstly, the throughput is low as the number of transactions that the blockchain can handle is limited. Secondly, the transaction speed is slow, taking a long time to process a block, leading to a delayed confirmation of transactions. The number of terminal nodes in the Internet of Things (IoT) is constantly increasing and is expected to continue to grow exponentially. The massive number of nodes presents new challenges to the scalability of the blockchain, which will require further research and development efforts to address.

REFERENCES

[1] L. D. Xu, Y. Lu, and L. Li, "Embedding blockchain technology into IoT for security: A survey," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10452–10473, Jul. 2021, doi: 10.1109/JIOT.2021.3060508.

[2] J. Shi and R. Li, "An overview of blockchain access control under the Internet of Things," *J. Softw.*, vol. 30, no. 6, pp. 1632–1648, 2019, doi: 10.13328/j.cnki.jos.005740.

[3] C. V. Hu, D. Ferraiolo, R. Kuhn, R. A. Friedman, J. A. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to attribute based access control (ABAC) definition and considerations," *NIST Special Publication*, vol. 800, no. 162, pp. 1–54, 2013.

[4] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *Computer*, vol. 29, no. 2, pp. 38–47, 1996.

[5] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System, Bitcoin White Paper*. Accessed: Jan. 20, 2022. [Online]. Available: http://www.bitcoin.org/bitcoin.pdf

[6] M. Hemdi and R. Deters, "Using REST based protocol to enable ABAC within IoT systems," in *Proc. IEEE 7th Annu. Inf. Technol., Electron. Mobile Commun. Conf. (IEMCON)*, Oct. 2016, pp. 1–7, doi: 10.1109/IEMCON.2016.7746297.

[7] A. Yavari, A. S. Panah, D. Georgakopoulos, P. P. Jayaraman, and R. van Schyndel, "Scalable role-based data disclosure control for the Internet of Things," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2017, pp. 2226–2233, doi: 10.1109/ICDCS.2017.307.

[8] Q. Liu, H. Zhang, J. Wan, and X. Chen, "An access control model for resource sharing based on the Role-Based access control intended for Multi-domain manufacturing Internet of Things," *IEEE Access*, vol. 5, pp. 7001–7011, 2019, doi: 10.1109/ACCESS.2017.2693380.

[9] S. Das and S. Namasudra, "Multiauthority CP-ABE-based access control model for IoT-enabled healthcare infrastructure," *IEEE Trans. Ind. Informat.*, vol. 19, no. 1, pp. 821–829, Jan. 2023, doi: 10.1109/TII.2022.3167842.

[10] A. E. Gencer, S. Basu, and I. Eyal, "Decentralization in bitcoin and Ethereum networks," in *Proc. 22nd Int. Conf. Financial Cryptogr. Data Secur.*, 2018, pp. 439–457.

[11] Z. Xin-Fang, F. Ming-Wei, and W. Jun-Jun, "An indoor location-based access control system by RFID," in *Proc. IEEE Int. Conf. Cyber Technol. Autom., Control, Intell. Syst.*, Mar. 2011, pp. 43–47, doi: 10.1109/CYBER.2011.6011761.

[12] J. L. Hernández-Ramos, M. P. Pawlowski, A. J. Jara, A. F. Skarmeta, and L. Ladid, "Toward a lightweight authentication and authorization framework for smart objects," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 4, pp. 690–702, Apr. 2015, doi: 10.1109/JSAC.2015.2393436.

[13] J. P. Cruz, Y. Kaji, and N. Yanai, "RBAC-SC: Role-based access control using smart contract," *IEEE Access*, vol. 6, pp. 12240–12251, 2018, doi: 10.1109/ACCESS.2018.2812844.

[14] A. Ouaddah, A. A. Elkalam, and A. A. Ouahman, "FairAccess: A new blockchain-based access control framework for the Internet of Things," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 5943–5964, 2017.

[15] E. Grummt and M. Müller, "Fine-grained access control for EPC information services," in *Proc. 1st Int. Conf. Internet Things-IoT*, vol. 4952. Berlin, Germany: Springer-Verlag, Mar. 2008, pp. 35–49.

[16] J. E. Kim, G. Boulos, J. Yackovich, T. Barth, C. Beckel, and D. Mosse, "Seamless integration of heterogeneous devices and access control in smart Homes," in *Proc. 8th Int. Conf. Intell. Environ.*, Jun. 2012, pp. 206–213, doi: 10.1109/IE.2012.57.

[17] D. Rivera, L. Cruz-Piris, G. Lopez-Civera, E. de la Hoz, and I. Marsa-Maestre, "Applying an unified access control for IoT-based intelligent agent systems," in *Proc. IEEE 8th Int. Conf. Service-Oriented Comput. Appl. (SOCA)*, Oct. 2015, pp. 247–251, doi: 10.1109/SOCA.2015.40.

[18] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, "Smart contract-based access control for the Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1594–1605, Apr. 2019, doi: 10.1109/JIOT.2018.2847705.

[19] D. D. F. Maesa, P. Mori, and L. Ricci, "Blockchain based access control," in *Proc. IFIP Int. Conf. Distrib. Appl. Interoperable Syst.*, 2017, pp. 206–220.

[20] R. Xu, Y. Chen, E. Blasch, and G. Chen, "BlendCAC: A smart contract enabled decentralized capability-based access control mechanism for the IoT," *Comput.*, vol. 7, no. 3, pp. 39–46, 2018.

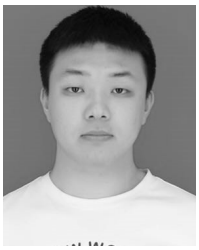
[21] *Geth Client for Building Private Blockchain Networks*. Accessed: Apr. 12, 2020. [Online]. Available: https://github.com/ethereum/go-ethereum/wiki/geth

[22] *Web3 Javascript API to Interact With Ethereum Nodes*. Accessed: Apr. 12, 2020. [Online]. Available: https://github.com/ethereum/wiki/wiki/JavaScript-API
[23] *An Introduction to Ethereum Platform*. Accessed: Aug. 16, 2022. [Online]. Available: http://ethdocs.org/en/latest/introduction/what-is-ethereum.html
[24] *Truffle: The Most Comprehensive Suite of Tools for Smart Contract Development*. Accessed: Oct. 23, 2018. [Online]. Available: https://github.com/trufflesuite/truffle
[25] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proc. 3rd Symp. Operating Syst. Design Implement. (OSDI)*, 1999, pp. 173–186.

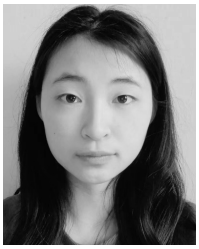
FEIFEI GUO received the B.S. degree in information management and information systems from Jiangsu University of Science and Technology. She is currently pursuing the M.S. degree with the Computer Science Department, Nanjing University of Aeronautics and Astronautics, Nanjing. Her current research interests include blockchain, the Internet of Things, and privacy preservation.

GUOHUA SHEN received the M.S. and Ph.D. degrees in computer science from Nanjing University of Aeronautics and Astronautics, China. He is currently an Associate Professor with the College of Computer Science and Engineering, Nanjing University of Aeronautics and Astronautics. His research interests include requirement traceability, fog computing, description logic, semantic web, and web services and ontology.

ZHIQIU HUANG received the Ph.D. degree in computer science from Nanjing University of Aeronautics and Astronautics. He is currently a Professor with the College of Computer Science and Engineering, Nanjing University of Aeronautics and Astronautics, where he is also the Director of software safety in computer science. His research interests include formal method, cloud computing, edge computing, web security, and privacy preservation.



YANG YANG received the M.S. degree in computer science from Nanjing University of Aeronautics and Astronautics, Nanjing, where he is currently pursuing the Ph.D. degree. His current research interests include blockchain, privacy preservation, and access control.



MENGNAN CAI received the B.S. degree in information security from Anhui University of Science and Technology. She is currently pursuing the M.S. degree with the Computer Science Department, Nanjing University of Aeronautics and Astronautics, Nanjing. Her current research interests include data privacy, differential privacy, and data publishing.

LINLIN WEI received the B.S. degree in computer science (Internet of Things) from Shanxi University of Science and Technology. He is currently pursuing the M.S. degree with the Computer Science Department, Nanjing University of Aeronautics and Astronautics, Nanjing. His current research interests include software engineering, data mining, and text analysis.