Hindawi
Mobile Information Systems
Volume 2020, Article ID 8889192, 19 pages
https://doi.org/10.1155/2020/8889192

Research Article

A^2 Chain: A Blockchain-Based Decentralized Authentication Scheme for 5G-Enabled IoT

Xudong Jia,[1] Ning Hu,[1,2] Shi Yin,[1] Yan Zhao,[1] Chi Zhang,[1] and Xinda Cheng[1]

[1]*Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China*
[2]*Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen 518000, China*

Correspondence should be addressed to Ning Hu; huning@gzhu.edu.cn

Received 13 August 2020; Revised 7 October 2020; Accepted 25 October 2020; Published 21 December 2020

Academic Editor: Vishal Sharma

The fifth-generation mobile communication technology (5G) provides high-bandwidth and low-latency data channels for massive IoT terminals to access the core business network. At the same time, it also brings higher security threats and challenges. Terminal identity authentication is an important security mechanism to ensure the core business network; however, most of the existing solutions adopt a centralized authentication model. Once the number of authentication requests exceeds the processing capacity of the authentication center service, it will cause authentication request congestion or deadlock. The decentralized authentication model can effectively solve the above problems. This article proposes a decentralized IoT authentication scheme called A^2 Chain. First, A^2 Chain uses edge computing to decentralize the processing of authentication requests and eliminate the burden on authentication services and the network. Second, to implement cross-domain identity verification of IoT devices, A^2 Chain uses blockchain, and sidechain technologies are used to securely share the identity verification information of IoT devices. Additionally, A^2 Chain replaces public key infrastructure (PKI) algorithm with identity-based cryptography (IBC) algorithm to eliminate the management overhead caused by centralized authentication model.

1. Introduction

With the rapid development of 5G networks, their fast speed, low latency, and high access will provide a broader platform for the development of IoT technology [1–3]. As defined by 3GPP, 5G supports access to at least 10^6 devices per square kilometre [4]. As 5G powers IoT, it also brings huge challenges to IoT security [5]. The authentication of IoT devices is an important step in ensuring IoT security.

Traditional authentication schemes are usually centralized, which has high latency and untimely response problems in the 5G mass IoT device access scenario [6]. On the one hand, the authentication server or network node will have serious network congestion when massive IoT devices ask for authentication in this era where IoT devices are ubiquitous, and this will seriously affect the service quality of IoT applications [7–11]. On the other hand, centralized authentication usually requires the authentication center to respond to the authentication request of the IoT device. However, due to the long link distance, it cannot satisfy the delay-sensitive applications (for example, the internet of vehicles and unmanned aerial drones) [12]. Also, traditional centralized authentication uses a public key infrastructure-based authentication structure, which carries high computational and communication costs for IoT devices that have limited resources in terms of power, memory, and processing power [13, 14]. Secondly, in traditional public key infrastructure- (PKI-) based authentication models, there is a single point of failure and third-party trustworthiness issues [15].

Decentralized IoT authentication can meet the authentication needs of a large number of IoT devices, and authentication latency issues can be solved by authenticating the device identity through edge nodes. In decentralized authentication scheme of IoT, the decentralized security mechanism is necessary to protect network resources or data, especially to ensure the consistency of authentication data of edge authentication services. In recent years, blockchain technology has gained widespread attention in authentication and access control research due to its decentralized and cryptographic properties. There is a natural fit between blockchain technology as a distributed ledger and the decentralized edge computing model, and researchers have already adopted edge computing to support services in blockchain networks [16–18]. Besides, blockchain's non-falsifiability and fault tolerance make it a good solution to authentication problems [19]. For example, in [20], the feasibility of using blockchain technology for IoT device authentication in edge computing systems is discussed, and blockchain-based smart contracts are introduced to handle the operation of authentication-related certificates; Jia et al. [21] proposed a blockchain-based cross-domain authentication system applied to the authentication process for data access to different IoT application domains; The study [22] was based on blockchain and elliptic curve cryptosystem cross-data center authentication and key exchange programs.

However, in existing blockchain-based authentication schemes, the authentication information of a large number of IoT devices is stored in a single blockchain, which poses a huge storage burden and scalability problem for blockchain nodes. In this paper, we propose a decentralized IoT authentication scheme combining edge computing and sidechain techniques. We named it A^2 Chain since the proposed authentication model includes two types of blockchains: application domain blockchain and alliance blockchain. Compared to past work, the innovations and contributions in this paper are as follows:

(1) Edge computing technology is adopted to decentralize the processing of authentication requests through the authentication service nodes deployed at the edge of the network. On the one hand, it reduces the server burden caused by a large number of IoT authentication requests. On the other hand, the authentication service processing is close to the terminal, which can reduce the network burden and authentication delay and improve communication efficiency.

(2) An IoT authentication structure based on application domain-alliance chains is proposed to deploy blockchains in different application domains as well as between application domains, respectively. The application domain blockchain acts as a sidechain for the alliance blockchain. Each application domain blockchain can run the intradomain authentication process independently within the application domain. The federation chain stores the authentication information index of the application domain devices and proves the existence of the authentication information by simplified payment verification (SPV) proof when cross-domain authentication is required. This structure occupies less storage space and improves the efficiency of searching for target information.

(3) Identity-based cryptography (IBC) is proposed for identity authentication without introducing any trusted third parties. In this case, public key certificates are no longer required, reducing the heavy workload of issuing, maintaining, and revoking digital certificates.

The remainder of this paper is organized as follows: an overview of the related work is given in Section 2. In Section 3, the problem is further stated. Section 4 provides an overview of the proposed solution, Section 5 details the certification process of the solution, and Section 6 verifies the effectiveness and efficiency of the solution through experiments. In Section 7, we discuss the shortcomings of the program and look forward to future research directions. Finally, in Section 8, we conclude the paper.

2. Related Work

2.1. Authentication Scheme Based on Traditional Scheme. At present, IoT usually adopts centralized authentication, which is more costly, prone to a single point of failure, and less efficient in the case of mass device authentication. [23]. Esfahani et al. [24] proposed a lightweight industrial Internet of Things (IIoT) device authentication mechanism, but it stores the authentication data on a local server. Therefore, it is susceptible to a single point of failure. In order to meet the authentication requirements of many IoT devices in the 5G network environment, Ni et al. [25] used fog computing and network slicing technology to propose an efficient and secure service-oriented authentication framework. Users can use the fog node to select the appropriate network slice according to the service type of the access service and efficiently establish a connection with the core network. Gross et al. [26] introduced an authentication method based on IPsec and TLS. However, the higher computational cost required by Gross' scheme is intolerable for resource-constrained IoT devices. Lai et al. [27] proposed the CPAL scheme in order to enable IoT devices to access the mobile Internet all the time. In CPAL, secure roaming authentication can be provided for IoT devices through group signature technology.

2.2. Authentication Scheme Based on Blockchain. Blockchain will play an important role in IoT device management and security due to its characteristics such as decentralized and untamperable. In [28], Hammi et al. discussed the current dilemma in IoT authentication and propose bubbles of trust, a decentralized authentication scheme based on blockchain, to create a secure virtual area in the blockchain, which enables IoT devices to communicate securely. However, due to the closed nature of its virtual region, IoT devices can only communicate with devices belonging to the same region and cannot communicate across domains. In [29], Bao et al. proposed the IoT Chain scheme, which consists of an authentication layer, a blockchain layer, and an application layer to achieve authentication, access control, privacy protection, lightweight features, regional node fault tolerance, denial-of-service resilience, and storage integrity. Khalid et al. [30] proposed a lightweight decentralized IoT based on the fog computing and blockchain technology Internet authentication scheme, IoT devices will be tied to the IoT application system where they are located when they register, and the latter will issue tokens for them, thus enabling secure communication between devices. Zhang et al. [31] proposed a blockchain-based decentralized vehicle authentication scheme and designed collaborative authentication based on secret sharing and blockchain-based data tracking and trust management in a dynamic agent edge computing model. Cui et al. [32], in his study, a hybrid blockchain-based multi-WSN authentication scheme, designed a wireless-aware network hierarchical model and a hybrid blockchain model combining private and public blockchains. For different types of devices, different blockchains were used for authentication. The authentication information of all nodes was stored on the public blockchain at that time, which caused a certain storage burden.

3. Proposed Scheme

In this part, we design an application domain-alliance chain IoT authentication model called A^2 Chain to meet the need for secure authentication in IoT. Firstly, the problem presented in this paper is stated; secondly, reasonable assumptions are made about the scheme; and finally, based on the above assumptions, our proposed authentication scheme is presented.

3.1. Motivation and Basic Idea. Authentication is one of the indispensable means to ensure the security of network communication. 5G enables IoT to have higher transmission speed and capacity and lower transmission delay and can provide high coverage and massive device deployment for the Internet of Things applications [2, 3]. These massively connected terminal devices simultaneously initiate authentication requests, which will have a serious impact on the authentication server [9–11]. In traditional authentication mechanisms, a centralized mechanism is usually used, as in Figure 1(a), where all devices are authenticated through a centrally located authentication server. In the case of massive device access, centralized authentication will bring about a challenge to the availability of legitimate devices, or a weak link in resource exhaustion attacks [15, 33].

As a decentralized and distributed technology, blockchain provides a new solution to the problems that exist in IoT authentication [28, 34, 35]. As in Figure 1(b), blockchain-based authentication schemes decentralize IoT authentication by establishing a blockchain network at a gateway or authentication server in the system to achieve distributed management of the authentication process. These solutions work well to overcome the single point of failure of centralized authentication, third-party trust, and the difficulty of resisting DoS attacks. However, there are still some limitations and challenges of existing blockchain-based authentication schemes as follows:

(1) Low authentication efficiency: the authentication process for IoT applications requires an authentication server to handle authentication requests, although blockchain-based authentication schemes enable distributed management of the authentication process and no longer rely on a single centralized authentication center. However, authentication servers are usually deployed on the side away from the end device or on cloud servers. This imposes higher latency and bandwidth consumption on the authentication process [36]. And when in IoT applications, latency-sensitive applications have strict requirements on response time. In addition, as the number of IoT devices increases, the burden on the authentication server increases significantly, which will also bring bottlenecks and delays in the system communication, thus limiting the quality of the system service.

(2) Scalability problem: with the popularity of IoT, the identity authentication problem in IoT does not only exist in a single IoT application, but also in different IoT applications with the same authentication needs [21, 37], which we call cross-domain authentication. Blockchain-based authentication schemes are usually deployed in a single application domain or intelligent system, and authentication information from different application domains or systems is not interoperable, lacking an effective cross-domain authentication scheme.

(3) Storage overload: even though some schemes solve the cross-domain authentication problem to some extent by forming federated blockchains [38–40], due to the nature of the blockchain, the full node of the blockchain must store every block on the blockchain and the transactions it contains. Therefore, each authentication server has to store all registered IoT endpoint authentication information, which includes not only authentication information from this application, but also information from other application domains. The information from other application domains may include a large number of devices that do not require cross-domain authentication, resulting in a waste of storage space. All device information is stored in the federated blockchain, and frequent authentication operations consume a lot of resources and time, which does not meet the real-time requirements of IoT.

To overcome the aforementioned shortcomings of blockchain-based authentication, we propose to combine blockchain-sidechain technology as well as edge computing technology to authenticate IoT devices. In the decentralized authentication scheme proposed in this paper, the basic ideas include the following aspects. First, in the organization of the authentication architecture, we propose an authentication architecture based on edge computing to deploy authentication service nodes at the edge of the network, which we call edge authentication nodes (EAs). Since the authentication service is closer to the end device side, authentication requests from a large number of endpoints do

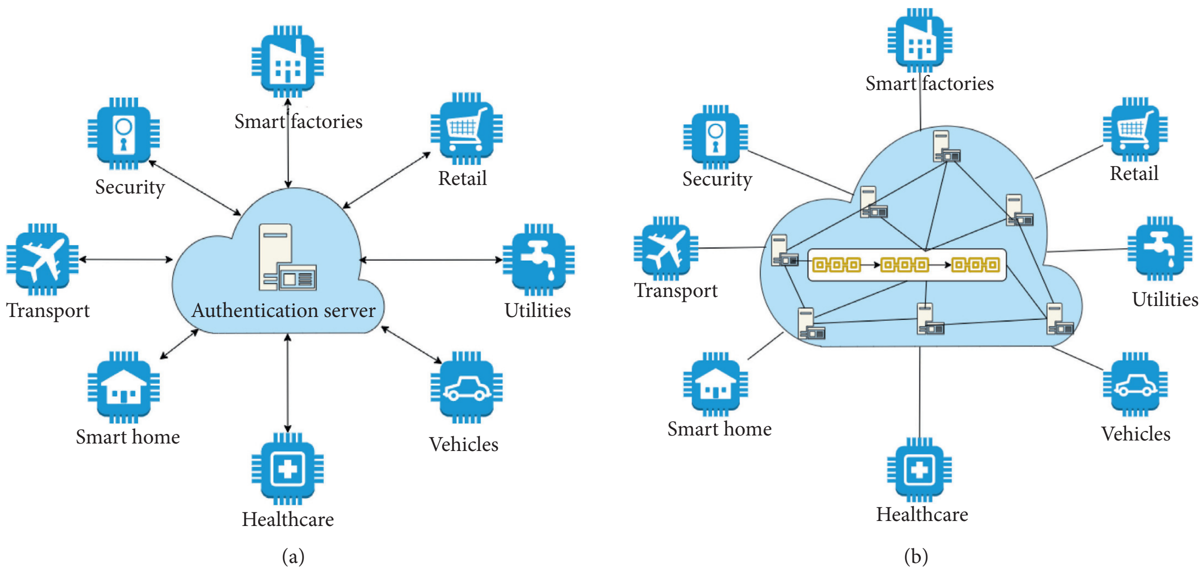

FIGURE 1: (a) Centralized and (b) decentralized authentication models.

not need to be sent to the core network, which can effectively reduce authentication latency and network burden [41, 42]. Second, we use blockchain and sidechain technology to build different application domain blockchains and alliance blockchains to share authentication information, instead of the trusted third-party authorization process. On the one hand, different application domains can ensure the independence of their own applications in different sidechains; on the other hand, sidechain technology provides a secure decentralized peer-to-peer data-sharing platform, and each application domain does not need to store unnecessary authentication information, which reduces the storage burden of blockchain nodes and improves the scalability of the authentication model [43, 44]. Finally, in terms of the signature algorithm, we propose to adopt an identity-based signature algorithm [45], which can determine the authenticity of the user without a trusted third party and reduce the overhead of storing certificates on the end device.

When a user's device wants to access the network of the application domain to which it belongs, it can initiate an authentication request to the nearest edge authentication server to quickly pass identity authentication. When the device wants to access the network or data of other application domains, it can use sidechain technology to prove the reliability of its authentication information through SPV to achieve cross-domain authentication. In the following chapters, we will describe our plan in detail.

3.2. Assumptions.

3.2. Assumptions. We propose an IoT authentication model scheme based on an application domain-alliance blockchain based on several reasonable assumptions that can be satisfied under certain conditions. The assumptions are as follows:

(1) Each IoT device has a unique object identifier [21], the object identifier (OID) structure is <Domain_ID. Category_ ID. Entity_ID>, and Table 1 describes the meaning of each field

(2) All domain management nodes and edge authentication nodes are legitimate and trusted

(3) The system initialization and key distribution process is secure

3.3. System Architecture. According to different node functions, IoT nodes can be divided into domain management nodes, edge authentication nodes, and terminal devices. In order to facilitate the management of IoT devices and achieve their secure authentication, the system architecture is designed as shown in Figure 2 according to the different terminal device functions or usage scenarios. The entire architecture is divided into multiple application domains, and each network includes domain management nodes, edge authentication nodes, and end devices:

(1) Domain manage node (DM): the main function is to manage the nodes in the application domain. As a node manager, it is trusted by the nodes in the network, and the terminal devices in the application domain need to register with the domain manage node before entering the network, and the domain manage node generates the private key for them according to the identity information provided and returns to the terminal devices.

(2) Edge authentication node (EA): edge authentication node is used to authenticate the identity of end devices and has strong computational and storage capabilities. Edge authentication nodes are deployed decentralized at the edge of the network, near the end device side, and have low latency, which reduces the load on authentication services and the network [12]. Through distributed edge authentication nodes, we have realized the decentralization of the authentication structure.

Table 1: OID description.

Field	Mandatory (M)/Option (O)	Interpretation
Domain ID	M	Registered domain ID
Category ID	O	Categories of entities in the security domain, such as devices and servers
Entity ID	M	The unique number assigned to the entity

(3) Terminal device (TD): This consists mainly of a large number of IoT terminal devices deployed in various application scenarios for sensing, serving, and communicating. These devices can detect or generate data for transmission to different IoT applications. Authentication is required to access the network or to access data.

3.3.1. Types of Authentication. The various nodes in the network collaborate with each other to accomplish various IoT application tasks. When a terminal device accesses the network or needs to access data, it needs to be authenticated, and in our proposed scenario, two types of authentication scenarios are involved:

(A) Intradomain authentication: due to business requirements, the terminal device needs access to its registered application domain data, as shown in Request 1 in Figure 2. In this case, the edge authentication node can easily retrieve the public parameters of the terminal signature through the application domain blockchain to authenticate the identity of the terminal device. Such authentication type we call intradomain authentication.

(B) Cross-domain authentication: with the rapid development of IoT, the number of application domains is increasing rapidly, and the interaction between different application domains is becoming more frequent. In some cases, devices from different application domains need to collaborate to complete a task, and terminal devices need to access application domain data outside their registered application domain, such as request 2 in Figure 2. Unlike intradomain authentication scenarios, application domains do not necessarily trust each other, as a domain is usually reluctant to let others access its sensitive data. In addition, the edge authentication node and the terminal device belong to different application domains, and the public parameters of the system signature are also different; in order to achieve the secure transmission of data from different application domains and terminal communication, cross-domain authentication of IoT terminals needs to be implemented, and the detailed process is described in Section 4.

3.3.2. A^2 Chain Model. In the past blockchain-based authentication schemes, IoT nodes join the same blockchain network, but a large number of IoT nodes frequently undergoing authentication operations will bring a lot of resources and time consumption, cannot meet the real-time requirements of IoT devices [29]; at the same time, different application domains join the same blockchain, application domain authentication devices not only need to store the authentication information in this domain, but also need to store the authentication information of other application domains, greatly increasing the storage pressure of the authentication device and information search space, will also affect the efficiency of the authentication service.

To this end, in this paper, we propose an A^2 Chain authentication model, which consists of two main parts: the application domain blockchain and the alliance blockchain.

(1) Alliance blockchain: all domain management nodes are connected to the alliance blockchain as nodes of the alliance blockchain. The alliance blockchain is connected to multiple application domain blockchains via domain management nodes for secure management and sharing of authentication information between different application domains.

(2) Application domain blockchain: application domain blockchain consists of domain management nodes and edge authentication nodes according to the application domain and location, which avoids the consumption of computation and storage irrelevant transactions and reduces the delay caused by transmission. The application domain blockchain is used to store the authentication information of end devices within the domain.

In addition, through the noncentralized nature of the blockchain, A^2 Chain does not require a trusted third-party entity and achieves a good decentralized authentication.

3.4. Signature Algorithm. In the authentication process, the proposed system uses an identity-based cryptosystem [45]. Since there is no need to use public key certificates in identity-based cryptographic systems and no trusted third-party entities to issue certificates, it satisfies the need for decentralized authentication. Moreover, the use of identity-based signature algorithms can reduce the complexity of deployment and management and has unique advantages in protecting IoT applications.

We use the identity-based cryptographic standard SM9 [46] issued by the State Cryptography Administration for authentication, and the strength of SM9's encryption is equivalent to the RSA encryption algorithm for 3072-bit keys.

The SM9 signature algorithm consists of five steps: system parameter generation, master key generation, device key generation, signature generation, and signature verification. The signer holds an identity and a corresponding private key, which is generated by the key generation server

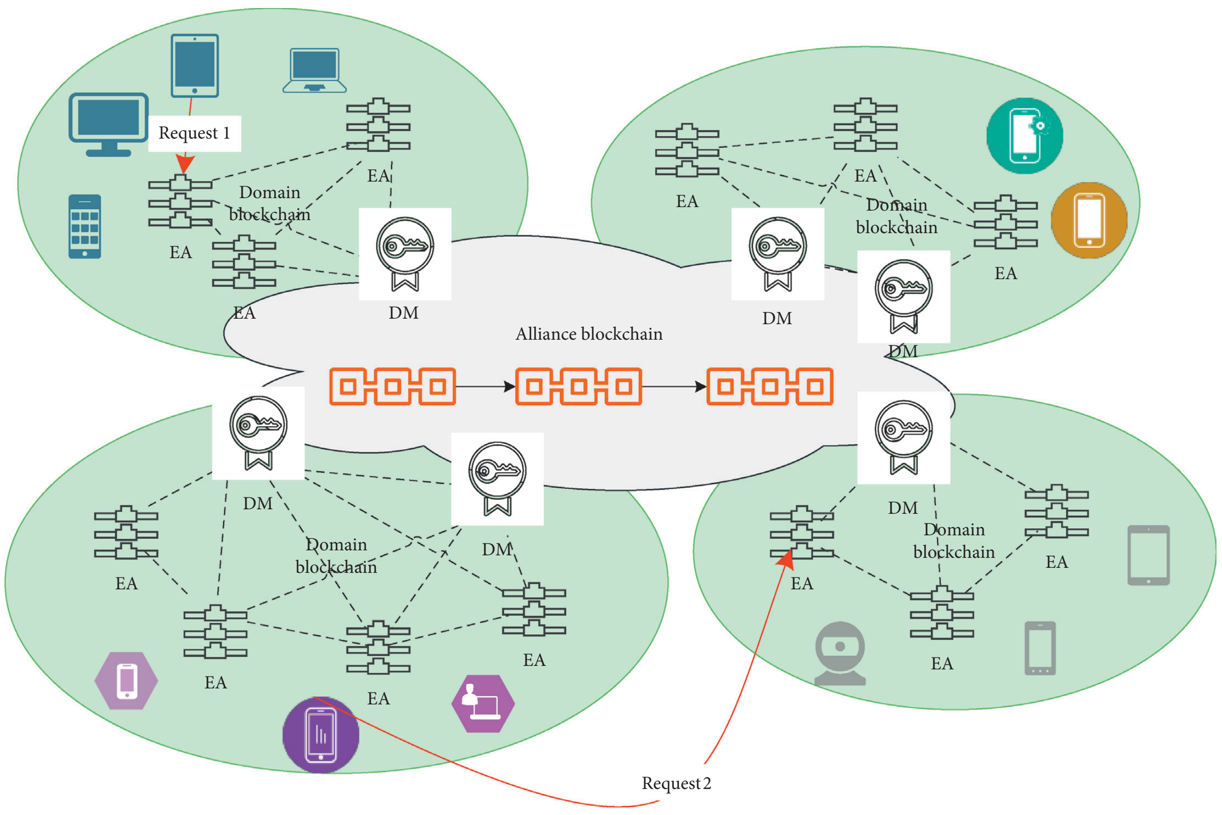

Figure 2: System architecture of A^2 Chain.

(KGS) through the combination of the master private key and the signer's identity. The signer uses its own private key to generate a digital signature on the data, and the verifier uses the signer's identity to generate its public key to verify the reliability of the signature, i.e., to verify the authenticity and integrity of the sent data and the identity of the data sender:

System parameter generation (SPG): it includes the curve identifier cid; the parameters q of the base field F_q of the elliptic curve; the parameters a and b of the elliptic curve equation; the prime factor N of the curve order and the residual factor cf relative to N; the embedding degree k of the curve $E(F_q)$ with respect to N; the generator P_1 of the N order cyclic subgroup G_1 of $E(F_{q^{d_1}})$, where d_1 divides k; the generator P_2 of the N-order cyclic subgroup G_2 of $E(F_{q^{d_2}})$, where d_2 divides k; identifier eid for bilinear pair e, bilinear pair e: $G_1 \times G_2 \longrightarrow G_T$, the order of G_T is N; optionally, the homomorphic mapping Ψ from G_2 to G_1.

System master key generation (MKGen): the KGS server generates a random number $s \in [1, N-1]$ as the system's master private key and computes the element $P_{\text{pub}} = [s]P_2$ in G_2 as the system's master public key pair (s, P_{pub}).

Device signature key generation (DKGen): KGS selects and exposes a byte to represent the private key generation function identifier hid. Assuming the device's identity is ID, to generate the device's private key d, KGS first computes $t_1 = H_1(\text{ID}\|\text{hid}, N) + s$ on the finite domain F_N. If $t_1 = 0$, then it is necessary to regenerate the master private key, compute and expose the master public key, and update the existing device's private key; otherwise compute $t_2 = s \cdot t_1^{-1}$, and then calculate $d = [t_2]P_1$. d is the user's signed private key.

Signature generation (SigGen): assuming the message to be signed is a bit string M, perform the algorithmic steps given in Algorithm 1 as a signature device in order to obtain the digital signature (h, S) of the message M.

Signature verification (SigVer): in order to verify the received message M and its digital signature (h, S), the verifier performs the following arithmetic steps:

4. Authentication Mechanism

In this section, we will present the working of the proposed A^2 Chain. The system consists of three main phases: the initialization phase, the registration phase, and the device authentication phase.

4.1. Initialization Phase. In the initialization phase, each domain management node uses the SPG algorithm to generate public parameter group *parameters* and calls MKGen and DKGen to generate master key pair (s, P_{pub}) and asymmetric key pairs DM_{SK}, DM_{ID} of the domain management node, in which DM_{SK} demonstrates the signed private key of the domain management node and DM_{ID} represents the identity of the domain management node and the public key corresponding to DM_{SK}.

```
**Input**: message $M$, $P_{pub}$, and *parameters*m private key $d$
**Output**: signature $(h, S)$
(1) Compute the element $g = e(P_1, P_{\text{pub}})$ in group $G_T$;
(2) Generate a random number $r \in [1, N-1]$;
(3) Compute the element$=\ g^r$ in the group $G_T$, converting the data type of $\omega$ to a bit string;
(4) Compute the integer $h = H_2(M\|\omega, N)$;
(5) Compute the integer $L = (r-h) \bmod N$; if $L = 0$, then return 2;
(6) Compute the element $S = [L]d$ in group $G_1$;
(7) Convert the data types $h$ and $S$ to byte strings and the signature of the message $M$ is $(h, S)$.
```

Algorithm 1: Signature generation (SigGen) algorithm.

```
**Input**: message $M$, $P_{\text{pub}}$, *parameters*, *ID*, and signature $(h, S)$
**Output**: verification result—succeed or fail.
(1) Convert the data type of $h$ to an integer, check whether $h \in [1, N-1]$ holds, and if it does not, the verification fails;
(2) Convert the data type of $S$ to a point on an elliptic curve, and check whether $S \in G_1$ holds, and if not, the verification fails;
(3) Compute the element $g = e(P_1, P_{\text{pub}})$ in the group $G_T$;
(4) Compute the element $t = g^h$ in the group $G_T$;
(5) Compute the integer $h_1 = H_1(\text{ID}\|\text{hid}, N)$;
(6) Compute the element $P = [h_1]P_2 + P_{\text{pub}}$ in the group $G_2$;
(7) Compute the element $u = e(S, P)$ in the group $G_T$;
(8) Compute the element $\omega = u \cdot t$ in the group $G_T$, converting the data type of $\omega$ to a bit string;
(9) Compute the integer $h_2 = H_2(M\|\omega, N)$, check whether $h_2 = h$ is valid, if so the verification passes; otherwise the verification fails.
```

Algorithm 2: Signature verification (SigVer) algorithm.

After generating public parameters and keys, the domain management node broadcasts them in the application domain; creates blocks with the identity ID DM_{ID}, the public parameter group *parameters*, and the master public key *P_pub*; writes them into the application domain blockchain; and stores their block numbers in the alliance blockchain. The relevant steps are as follows:

(1) First, the domain management node generates the public parameter group *parameters*, the master key pair (s, P_{pub}), and the asymmetric key pair DM_{SK}, DM_{ID}
(2) The domain management node creates a transaction $T_1 = (D_{ID}, DM_{ID}, parameters, P_{pub})$ in the application domain blockchain and checks whether there is a DM_{ID}in the blockchain to verify the transaction, where D_{ID} indicates the application domain ID
(3) If DM_{ID} already exists in the application domain blockchain, the transaction validation fails and an error notification is returned to the domain management node
(4) If the DM_{ID} does not exist in the application domain blockchain, the transaction will be allowed and a new block will be created for it
(5) When the domain management node initialization is successful, the domain management node DM_{ID}, domain ID, and its block number *Block_num* are uploaded to the alliance blockchain to form a reference record of the authentication information sharing process
(6) The domain management node calls the DKGen algorithm to generate the device's signature private key TD_{SK} according to the device's identification TD_{ID}, and the domain management node calls the SigGen algorithm to sign the TD_{SK} and generate the signature $Sig\ (TD_{SK})$ to send the device's signature private key TD_{SK} with the signature $Sig\ (TD_{SK})$ to the device TD_{ID}in a secure manner

4.2. Registration Phase. During this phase, the IoT device will register with a domain management node within its application domain, and upon successful registration, the device will be associated with that application domain and the associated information will be counted in the alliance blockchain.

The main steps in the registration phase are as follows:

(1) The device sends the registration request message $M_1 = TD_{ID}\|DM_{ID}\|\ \text{Request}_{\text{Reg}}\ \|\ LT\ \|\ TS\ \|SigGen\ (TD_{ID}\|DM_{ID}\|\text{Request}_{\text{Reg}}\|LT\|TS)_{TD_{SK}}$ to the domain management node; M_1 includes the identifier TD_{ID}, the domain management node DM_{ID}, the registration request $\text{Reqest}_{\text{Reg}}$, life time (LT), timestamp (TS), and the corresponding signature

(2) Upon receipt of the message M_1, the domain management node verifies that the received DM_{ID} is in this application domain and that the TD_{ID} has not been registered

(3) If the DM_{ID} does not belong to the application domain or a device with a TD_{ID} that has been registered, registration will not be allowed and the registration process will be terminated

(4) If the DM_{ID} belongs to the application domain and the TD_{ID} is not registered, registration is allowed

(5) Call the SigVer algorithm to verify the validity of the message by verifying the signature. If the validation is valid, the registration process continues; otherwise the registration process is aborted

(6) The domain management node creates a transaction $T_2 = TD_{ID}\|DM_{ID}\|LT\|SigGen(U_{ID}\|DM_{ID}\|LT)_{DM_{SK}}$ in the application domain blockchain, and the block structure is shown in Figure 3

(7) When the new device registration is successful, the device TD_{ID} and its block number *Block_num* will be uploaded to the alliance blockchain along with the corresponding domain management node DM_{ID} to form a reference record of the authentication information sharing process, as shown in Figure 4

Due to the business requirements of IoT applications, etc., the terminal device may need access to the network or data of other application domains. When a terminal device requests access to other application domains, the terminal device needs to be authenticated, i.e., cross-domain authentication. At this point, it can be combined with our previous study—IRBA scheme [21], where an end device with cross-domain access needs can make a cross-domain authorization request to the domain management node and the end device obtains the authorization from the management node of the application domain it needs to access through a threshold signature and credits the authorization to the alliance blockchain.

Similar to the terminal device registration process, an edge authentication node registers with the domain management node of the application domain to which it belongs in the same way to obtain its signature key pair. EA_{ID} and its block number *Block_num* and the corresponding domain management node DM_{ID} are also uploaded to the alliance blockchain to form a reference record of the authentication information sharing process.

4.3. Authentication Phase.

In the authentication phase, the authentication process is divided into intradomain and cross-domain authentication.

4.3.1. Intradomain Authentication.

The edge authentication node authenticates the identity of the registered device. The edge authentication node authenticates the following conditions to allow the device to communicate and access to other devices or to the system: (1) the DM_{ID}exists in the application domain blockchain; (2) the TD_{ID} is registered in the application domain blockchain; (3) the TD_{ID} is in the life cycle of the registration; (4) verify that the registration information is valid; and (5) verify that the TD_{ID} signature is valid.

The authentication process within the application domain is described as follows:

(1) The device TD sends authentication request message $M_2 = \text{Request}_{Auth\_local}\|D_{ID}\|DM_{ID}\| TD_{ID}\| TS\|Hash(Sig(TD_{ID}\|DM_{ID}\|LT)_{DM_{SK}}) \|Sig(\text{Request}_{Auth\_local}\| D_{ID}\|DM_{ID}\|TD_{ID}\|TS)_{TD_{SK}}$, $\text{Re}quest_{Auth\_local}$ on behalf of the application domain authentication request, D_{ID} on behalf of the application domain to which the terminal device belongs,DM_{ID} indicates the domain management node identity associated with the device, TD_{ID}indicates the identity of the device, TS for the timestamp, and Sig indicates the Signature for TD_{ID}.

(2) The edge authentication nodes check for the presence of the application domain blockchain for DM_{ID}.

(3) If $\mathrm{DM_{ID}}$ does not exist in the application domain blockchain, the authentication process ends with an error; otherwise, the edge authentication node obtains the master public key P_{pub} and the public parameter *parameters* of DM_{ID} and proceeds to the next authentication step.

(4) Checking for the presence of TD_{ID} in the application domain blockchain.

(5) If the given TD_{ID} does not exist in the application domain blockchain, the authentication process stops due to an error. Otherwise, the process will continue to the next step.

(6) Check that TD_{ID} is within the life cycle of the registration.

(7) If not in the life cycle, stop the authentication; otherwise continue to the next step.

(8) Check that the hash of the $\|DM_{ID}$ signature of the registration information is consistent.

(9) Verifying the validity of the signature $Sig(\text{Request}_{Auth\_local}\|D_{ID}\|DM_{ID}\|TD_{ID}\|TS)_{TD_{SK}}$.

(10) If the signature validation is successful, the authentication is successful; if the signature validation fails, the authentication fails.

(11) The edge authentication node returns the authentication results to the device and signs the results using the private key.

(12) The terminal device confirms the validity of the result by verifying the signature of the authentication result.

Because the device authentication process is implemented at the nearest edge authentication node rather than on a cloud-

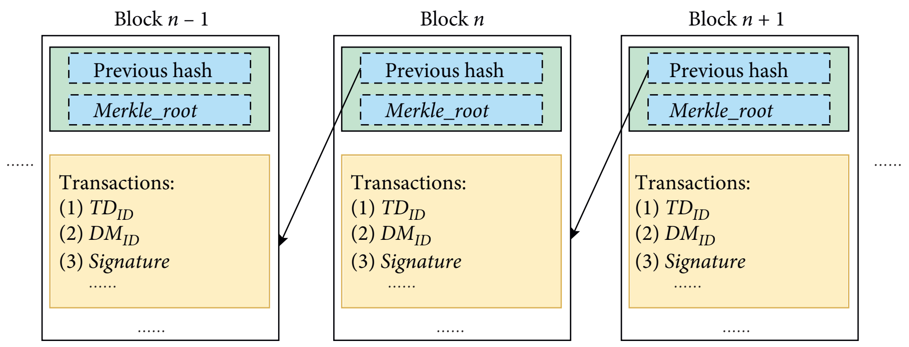

Figure 3: Structure and content of the application domain blockchain.

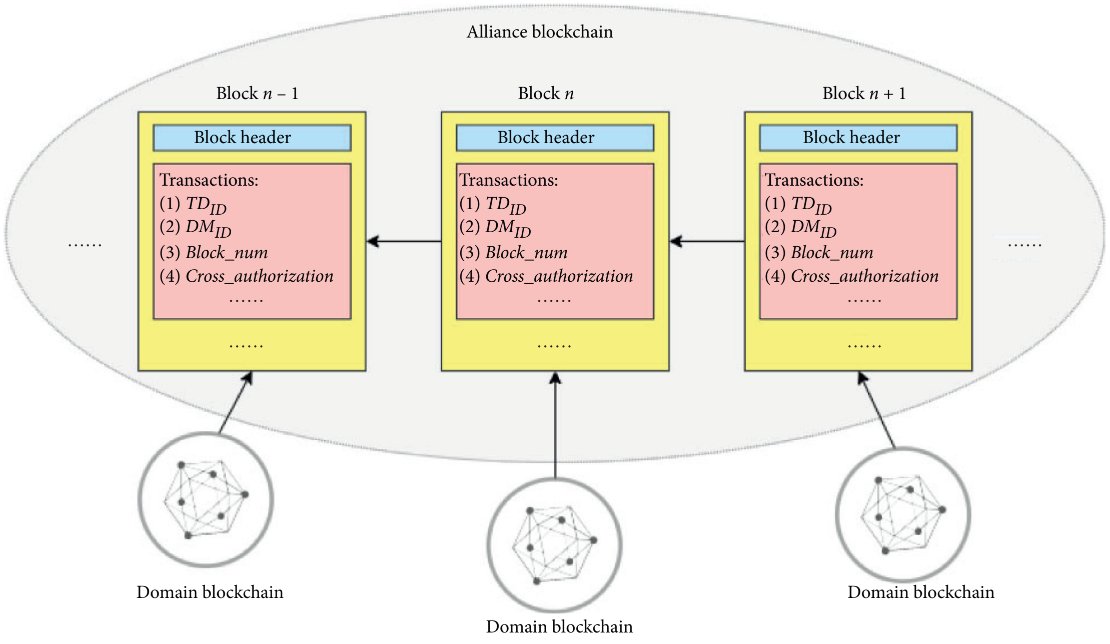

Figure 4: Store indexes in the alliance blockchain.

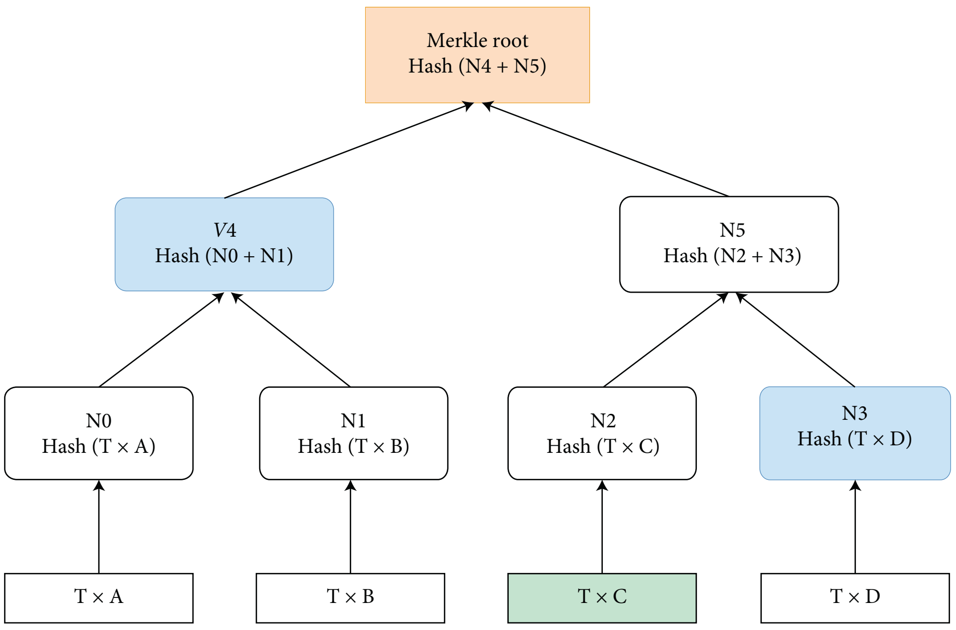

Figure 5: Merkle tree Path.

based server, the communication burden and latency are greatly reduced.

4.3.2. Cross-Domain Authentication. At present, a wide range of IoT applications are widely used and a large amount of IoT data is generated in different applications. Sharing data with other areas can be more useful as data sharing allows for a more rational allocation of resources and saves social costs. In order to achieve secure data sharing, future IoT networks need to implement a secure data sharing mechanism. In this case, if the terminal device needs to access data across application domains, the accessing application domain needs to be authenticated. However, if the previous authentication information block does not exist in the accessed application domain, the device will be required to re-register in the new application domain, which will take a lot of effort and time. Therefore, the authentication information should be able to be shared between application domains. In the cross-domain authentication process, we propose the use of sidechaining techniques to share authentication information from different application domains and use SPVs to prove the validity of the authentication information.

(1) Sidechain and Merkel Tree. Sidechain technology [47] was proposed to improve the scalability and extensibility of the blockchain, the basic idea being that digital assets can be transferred from one blockchain to another via sidechain protocols and to reduce the burden on the main chain, thereby increasing the throughput and speed of transactions [44, 48]. The flow of data between the main chain and the sidechains can be done using SPV (simple payment verification) proofs. SPV proofs consist of two parts: a list of block headers and a cryptographic proof, such as a Merkle proof, which indicates that a certain output occurred at a certain block in the list [49]. To prove that a certain transaction exists in a block, simply calculate the final Merkle root using the hash of this transaction against the hash of other related transactions and compare it to the root of the block header. If the result of the calculation agrees with the Merkle root of the block header, the transaction is proven to exist in this block. As shown in Figure 5, if we need to verify that a block contains a transaction Tx C and can get the hash value of the Merkle tree root, then we only need the Merkle path consisting of the hash values of N3 and N4 to prove it, as follows:

(1) First calculate the hash value of the transaction TX C, N2 = Hash (TX C).
(2) The hash value of the parent node is then obtained by summing the hash values of N2 and N3 and calculating the hash: N5 = Hash (N2 + N3).
(3) As above, calculate the hash value of the root node from the hash values of N4 and N5: root = hash (N4 + N5)
(4) Finally compare the hash value from the previous calculation with the root hash value of Merkle Tree in the block header; if it is the same then, the transaction TX C exists; otherwise it does not.

As we discussed in Section 3.1, existing blockchain-based authentication schemes suffer from authentication inefficiencies, scalability, and storage overloads. We propose to use a decentralized edge computing model to reduce authentication latency to improve authentication efficiency. To handle scalability and storage problems, we propose to build blockchains of different application domains using sidechain technology.

Sidechain, which is an extension of the blockchain, provides a decentralized peer-to-peer platform to maintain stored data while securely transferring authentication information between different application domains. The advantage of A^2 Chain's use of sidechain architecture is the independence of data and smart contracts, the alliance blockchain is primarily responsible for indexing, and the burden of the alliance blockchain does not increase with the number of application domains, avoiding the problem of rapid growth of data in the alliance blockchain. If the index between the alliance blockchain and the application domain blockchain is discarded, the application domain blockchain is an independently running blockchain that can run the domain authentication process independently. Based on the above, the sidechain technology not only improves the overall scalability of the system but also reduces the storage space of each application domain server and improves the search efficiency.

(2) Cross-Domain Authentication Process. The cross-domain authentication process is described below:

(1) The device TD sends an authentication request message $M_3 = \text{Request}_{Auth\_cross} \| D_{ID} \| DM_{ID} \| TD_{ID} \| TS \| parameters \| P_{pub} \| Sig(TD_{ID} \| DM_{ID} \| LT)_{DM_{SK}} \| Sig(\text{Request}_{Auth\_cross} \| D_{ID} \| DM_{ID} \| TD_{ID} \| TS)_{TD_{SK}}$ to the edge authentication node EA_B to access the application domain B. The D_{ID} terminal device belongs to the application domain, $\text{Request}_{Auth\_cross}$ is a cross-domain authentication request, DM_{ID} indicates the domain management node identity associated with the device, TD_{ID} indicates the identity of the device, TS is a timestamp, $parameters, P_{pub}$ indicates the public parameters required for its signature, and Sig indicates the signature of TD_{ID}.
(2) The edge authentication node EA_B receives the authentication request, and the authentication request is forwarded to the domain management node DM_B of the application domain in which it is located.

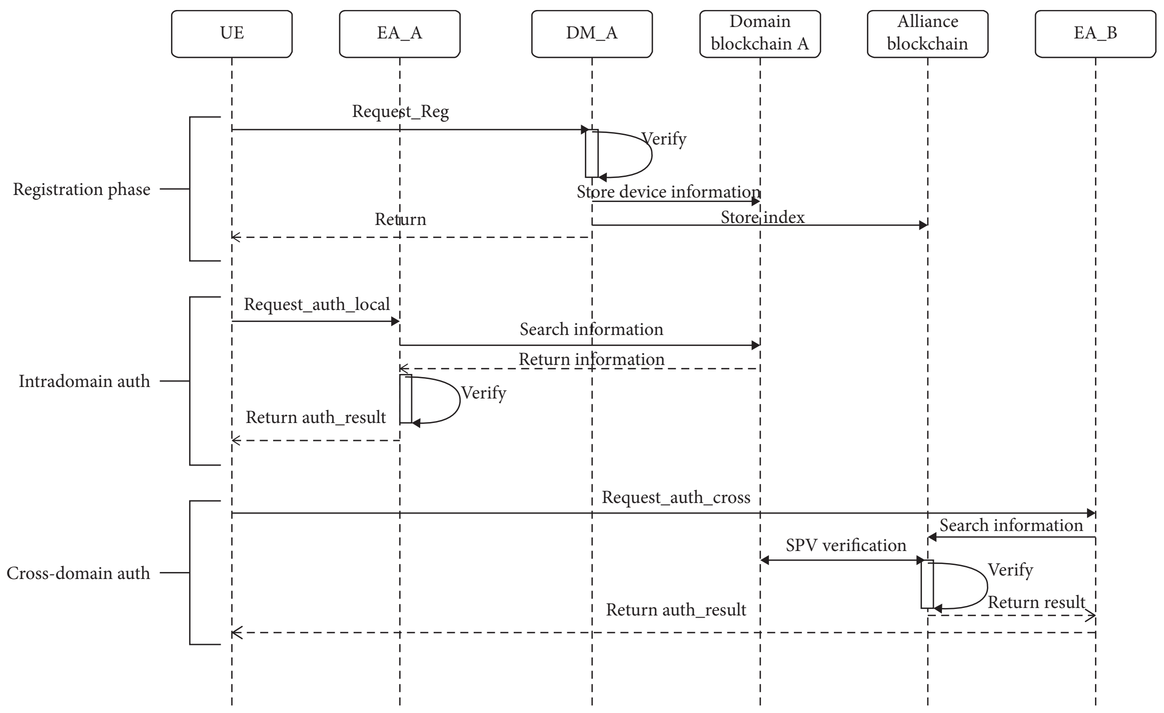

Figure 6: Overview of the authentication process. The device TD belongs to application domain A with intradomain authentication and cross-domain authentication, respectively.

(3) After the domain management node DM_B receives the request, it searches the alliance blockchain containing the corresponding information (includingDM_{ID},TD_{ID}, and the corresponding *Block_num*) through the authentication terminal device ID and its management domain management node ID to judge that the terminal device has been registered. If it is not registered, then stop the authentication; otherwise, continue the authentication process.

(4) Get the *Merkle_Path* of DM_{ID} andTD_{ID} in the blockchain of the application domain to which it belongs via Block_num.

(5) Verify the authentication information provided by the device. Compute the hash value of the authentication information provided by the end device and the hash value of each node on its *Merkle_Path*, and compute the hash *Merkle_hash* of the Merkle root of the block in which it is located.

(6) Check that the computed *Merkle_hash* matches the *Merkle_Root* of the block header of the block in which it is located.

(7) Verify that the signature $Sig(\text{Request}_{Auth\_cross}\|D_{ID}\|DM_{ID}\|TD_{ID}\|TS)_{TD_{SK}}$ is valid.

(8) Get terminal device authorization information to verify that the signature is valid.

(9) Return authentication results to the edge authentication server.

(10) The edge authentication server forwards the results to the terminal device and attaches a signature that uses the private key.

(11) Upon receipt of the authentication result that has been signed by the EA_B node, the terminal device will perform the same validation process to verify the signature to confirm the validity of the authentication result.

The authentication process and algorithm are shown in Figure 6 and Algorithm 3, respectively. At the end of the authentication process, the accessed application domain saves the end device's authentication information in the local blockchain so that the end device can later achieve fast authentication and simplify the cross-domain authentication process. (Algorithm 3).

5. Security Evaluation

In order to ensure the effective operation and service of the proposed authentication scheme, in this section, we analyse the proposed scheme for common security requirements and attacks in IoT applications:

(1) Integrity authentication: requests in the proposed system are signed by the requesting party using an identity-based signature algorithm before they are sent, and the final request message contains the data and the signature of the requestor. The receiving party can verify the message with the signature. In addition, authentication-related information is submitted to the alliance blockchain and the

```
 Initialization phase
 (1) if ($DM_{ID}.exist = true$) then
 (2)     return error()
 (3) else
 (4)     creat.block($D_{ID}$, $DM_{ID,\ parameters}$, $P_{pub}$, $Domian\_Chain$)
 (5)     creat.block($D_{ID}$, $DM_{ID}$, $Block\_num$, $Alliance\_Chain$)
 (6) end if
 Registration phase
 (1) if ($D_{ID}.exist = true$) then
 (2)    get($parameters$, $P_{pub}$)
 (3)    If ($TD_{ID}.exist = false$) then
 (4)        If ($Sig.TD_{ID} = valid$) then
 (5)            creat.block ($\mathrm{TD_{ID}}$, $\mathrm{DM_{ID}}$, LT, Sig $(\mathrm{TD}_I D \| \mathrm{DM_{ID}} \| \mathrm{LT})_{(\mathrm{DM_{SK}})}$, Domain_Chain)
 (6)            creat.block ($\mathrm{TD_{ID}}$, $\mathrm{DM_{ID}}$, Block_num, cross_authorization, Alliance_Chain)
 (7)            reg_sucess
 (8)        end if
 (9)    end if
(10) else
(11)     return error()
(12) end if
 Intradomain authentication
 (1) if ($DM_{ID}.exist = true$) then
 (2)    get($parameters$, $P_{pub}$)
 (3)    if ($TD_{ID}.exist \& LT = true$) then
 (4)        if ($Hash(Sig.DM_{ID}) \& Sig.TD_{ID} = valid$) then
 (5)            return auth_sucess
 (6)        end if
 (7)    end if
 (8) else
 (9)    return error()
(10) end if
 Cross-domain authentication
 (1) if ($DM_{ID}.exist \& TD_{ID}.exit = true$) then
 (2)    get($parameters$, $P_{pub}$)
 (3)    $DM_{ID}.Merkle\_Path$ = getDomainChainPath($DM_{ID}.Block\_num$)
 (4)    $DM_{ID}.Merkle\_Root$ = getDomainChainPath($DM_{ID}.Block\_num$)
 (5)    $DM_{ID}.Merkle\_hash$ = computeMerkleTree($DM_{ID}\_info$, $Merkle\_Path$)
 (6)    if ($DM_{ID}.Merkle\_hash = DM_{ID}.Merkle\_Root$) then
 (7)        $TD_{ID}.Merkle\_Path$ = getDomainChainPath($TD_{ID}.Block\_num$)
 (8)        $TD_{ID}.Merkle\_Root$ = getDomainChainRoot($TD_{ID}.Block\_num$)
 (9)        $TD_{ID}.Merkle\_hash$ = computeMerkleTree($TD_{ID}\_info$, $Merkle\_Path$)
(10)        if ($TD_{ID}.Merkle\_hash = TD_{ID}.Merkle\_Root$) then
(11)            if ($Sig.TD_{ID}$)&$cross\_authorization = valid$) then
(12)                return auth_sucess
(13)            end if
(14)        end if
(15)    end if
(16) else
(17)    return error()
(18) end if
```

Algorithm 3: Authentication mechanism.

application domain blockchain. Due to the features of the blockchain, the data cannot be tampered with once submitted, also ensuring the integrity of the message.

(2) Scalability: due to a large number of IoT applications and terminal devices, scalability is one of the important security requirements for IoT applications. In our proposed solution, terminal devices that can effectively authenticate access an identity-based signature scheme, and terminals do not need to store CA certificates, which is more flexible. The combination of application domain blockchain and alliance blockchain makes cross-domain authentication more convenient and business expansion of IoT applications more convenient and secure.

(3) Non-repudiation authentication: requests require a signature from the sender, and the private key used for the signature is generated by the sender's identifier and is kept by the sender. Therefore, the sender cannot repudiate the authentication request it has made.

(4) Authentication: for the terminal device that is going to access the network, it will first be registered in the system. The registration information will remain in the blockchain, and during the authentication process, the smart contract will check its legitimacy to allow the device to access the network.

(5) Mutual authentication: first, in our scheme, we assume that all domain management nodes and edge authentication nodes are trustworthy; if there are malicious nodes disguised as authentication nodes to perform phishing attacks on terminal devices, in our scheme, we require the authentication nodes to sign the authentication results, and the terminal devices can identify whether the authentication nodes are trustworthy and the validity of the authentication results by verifying the signatures.

(6) Sybil attack: in our proposed scheme, each endpoint device has a unique TD_{ID} in the network and is associated with its registered application domain D_{ID} and domain management node DM_{ID} during the registration process, and each communication is preceded by endpoint authentication. Authentication takes place on the application domain blockchain and the federation blockchain. It is not possible for an attacker to forge legitimate nodes in the network to communicate with other nodes.

(7) Spoofing attack: because each communication must be authenticated and its signature must be verified each time to prove its unique identity, an attacker cannot fake the identity of another node for an attack.

(8) Message replay attack: in our scheme, authentication requests need to be signed with a timestamped token attached to them. A request with invalid signature validation will be rejected by the system.

(9) Denial of service attack: authentication servers are scattered around the edge of the network, and attackers cannot expend significant resources on denial of service attacks against all authentication nodes. Even if one or some of the nodes fail, the remaining nodes can still work without affecting the normal operation of the system.

Through the above analysis, we compare it with existing blockchain-based IoT authentication solutions and get the results as shown in Table 2. Our proposed solution is more comprehensive in terms of security.

6. Performance Evaluation

6.1. Experimental Setup. We simulate two application domain blockchains and one alliance blockchain in our experiments, each containing the necessary entities, including domain management nodes and edge authentication nodes. The edge authentication node for each application domain runs on a separate host configured with an Intel (R) Core (TM) i7-6600U CPU with 8 GB of RAM. The domain management nodes run in a virtual machine that uses VMware Workstation 15 Pro hosted on an Intel (R) Core (TM) i7-6700 CPU 3.40 GHZ and 16 GB RAM. Four edge authentication nodes are set up per application domain. All machines are interconnected in a local network. The network connection of the virtual machines is configured to connect directly to the same LAN as the host in bridge mode. Terminal device operations are performed on a laptop with an Intel(R) Core(TM) i5-6300HQ and 4 GB RAM.

Table 2: Security comparison of different schemes.

	[37]	[17]	[18]	[28]	[29]	[50]	A^2 Chain
Sybil	✓	✓	✓	✓	✓		✓
Message replay	✓	✓	✓	✓	✓		✓
DOS		✓	✓	✓	✓	✓	✓
Scalability	✓	✓		✓	✓	✓	✓
Cross-domain authentication	✓					✓	✓
Decentralization	✓	✓	✓	✓	✓	✓	✓

The blockchain platform we have chosen for the proposed system is Hyperledger Fabric [51]. Hyperledger Fabric provides a scalable and extensible architecture that provides the basis for developing blockchain applications with a modular architecture. Unlike public blockchains, the Fabric platform is license-based, meaning that the participants in the blockchain network are not completely trustless with each other, which ensures the trustworthiness of the nodes. Smart contracts in Fabric become chain codes, and the writing of chain codes in Fabric can be done using the Written in a common programming language (e.g., Go, Node.js, and Java) rather than being restricted to domain-specific languages (domain-specific language, DSL), and Fabric does not require any transaction fees to perform operations such as chain coding or querying blockchain information. For authentication services, we use remote authentication dial-in user service (RADIUS) [52] to build the authentication servicer, which is often used to provide AAA (authenticate, authority, and audit) services. We chose the open source project, YH-RADIUS [53]. This project implements an extensible development framework for RADIUS.

6.2. Computing Consumption. Each entity is involved in different cryptographic operations during the system operation. We summarize the cryptographic operations involved in the operation of the system, as shown in Table 3 (in statistics, the authorization issuance and verification of the cross-domain authentication process is not available yet). Also, Table 4 shows the computational burden of the different components of the system during its operation. It should be noted that operations such as hashing, integer addition, and multiplication are not taken into account, as they take very little time in the tests.

Table 3: Notation description of cryptographic operations.

Notation	Description
T_{PA1}	A point addition in G_1
T_{PA2}	A point addition in G_2
T_{SM1}	A scale multiplication in G_1
T_{SM2}	A scale multiplication in G_2
T_{SMT}	A scale multiplication in G_T
T_{ET}	A exponentiation in G_T
T_{BP}	A bilinear pairing

Table 5 shows the computational overhead of the different components of the test in different processes. From the computational overhead, it can be seen that the main overhead of our proposed system lies in the initialization of the domain management nodes and the registration process of the end devices, which do not need to be performed in the authentication. The results show that common smart IoT devices can bear the computational burden of the system. In addition, the bilinear pair computation $g = e(P_1, P_{\mathrm{pub}})$ in the used SM9 signature algorithm can be stored as a constant in advance in the end devices during the signature process to further reduce the computational burden.

To further demonstrate the advantage of our proposed system in terms of computational overhead, we compare it with existing authentication schemes ES^{3}A [25], CPAL [27], LCCH [54], and E-AUA [55]. We first compared the overhead on the user side, as shown in Figure 7(a). It is clear that our proposed scheme takes less time to implement on the user side than the other schemes, as shown in Figure 7(b). We compared the computational overhead on the service side, and our proposed scheme also outperforms the other schemes.

In order to assess the time cost of the relevant operations on the blockchain, we used the blockchain testing tool Hyperledger Caliper [56] to test each type of operation 10,000 times, with run times as shown in Table 6. The time cost of both the registration and the transaction process is about 200 ms, which may seem high, but it is acceptable. Therefore, registration and transactions do not happen frequently, but only when new devices are registered and device authentication information is changed. It takes about 10 milliseconds to look up the authentication information on the blockchain. In the authentication process, even taking into account the time spent querying the blockchain, the time spent is far less compared to other schemes.

We further counted the number of computational operations included in each scheme, as shown in Table 7. It can be seen from the table that ES^{3}A [25], CPAL [27], and LCCH [54] are the three schemes with more complex cryptographic operations. E-AUA [55] and our proposed scheme are simpler in terms of cryptographic operations compared to the other three schemes, thus achieving a better performance.

6.3. Communication Consumption. In this section, we analyse the communication overhead of the proposed scheme. The end device sends 192 bytes signed authentication request to the edge authentication node. In the intra-application domain authentication process, the edge authentication node obtains the relevant authentication information directly from the local blockchain to verify the signature and authenticate the end device. In the cross-domain authentication process, the edge authentication node forwards the request to the domain management node after receiving the authentication request. The domain management node obtains 196 bytes of authentication information in a two-way Peg protocol. At the end of the authentication, the 32-byte authentication result is returned to the interrupting device. Therefore, the communication overhead of our scheme during intradomain and cross-domain authentication is 228 bytes and 616 bytes, respectively.

The number of interactions of our proposed scheme is significantly less than other schemes in the authentication process, and there is no certificate exchange process. Therefore, compared with ES3A, LCCH, CPAL, and E-AUA whose communication cost is 1336 bytes, 2016 bytes, 1232 bytes, and 652 bytes, respectively, the communication cost of our scheme is much smaller and more efficient. In Figure 8, we list the cost comparison between the above schemes and our scheme, which shows more visually the advantages of our scheme in communication overhead performance.

6.4. Storage Consumption. Different from existing blockchain-based authentication schemes, in our scheme the alliance blockchain stores only the index information of the IoT devices, so only simplified blocks of information need to be stored additionally in the domain management nodes. In contrast, in the existing scheme, the entire blockchain is updated by all nodes each time a new device is registered.

In our scheme, it is assumed that there are 10 application domains, each containing 10,000 IoT devices. As described in the scheme, the authentication information of the blockchain storage device in the application domain is about 8 bytes in the federation blockchain storage device ID and block number. The block header is 80 bytes, and the authentication information is about 192 bytes. For the traditional blockchain-based case and our proposed scheme, the storage overhead is about 26.32 MB and 5.95 MB, respectively. Our proposed scheme is only 22.6% of the traditional blockchain-based scheme, which greatly reduces the available storage space.

7. Discussion

A^2 Chain builds a decentralized IoT authentication scheme by introducing blockchain-sidechain technology with edge computing technology. In this scheme, we utilize the edge computing model to reduce authentication latency. However, in IoT applications, there are some scenarios where the terminals are dense. In this case, the authentication service needs to implement load balancing and congestion control for authentication requests. This is one of our future research directions.

In addition, in our scheme, we apply an identity-based signature algorithm SM9, and the keys of the terminal

Table 4: Stats on time-consuming cryptographic operations.

Phase	Computation cost		
	UE	EA	DM
Setup and register	—	—	$2T_{\mathrm{SM1}} + T_{\mathrm{SM2}} + 2T_{\mathrm{SMT}} + T_{\mathrm{PA2}} + 2T_{\mathrm{ET}} + 2T_{\mathrm{BP}}$
Intradomain authentication	$T_{\mathrm{SM1}} + T_{\mathrm{SM2}} + 2T_{\mathrm{SMT}} + T_{\mathrm{PA2}} + 2T_{\mathrm{ET}} + 2T_{\mathrm{BP}}$	$T_{\mathrm{SM1}} + T_{\mathrm{SM2}} + 2T_{\mathrm{SMT}} + T_{\mathrm{PA2}} + 2T_{\mathrm{ET}} + 2T_{\mathrm{BP}}$	\
Cross-domain authentication	$T_{\mathrm{SM1}} + T_{\mathrm{SM2}} + 2T_{\mathrm{SMT}} + T_{\mathrm{PA2}} + 2T_{\mathrm{ET}} + 2T_{\mathrm{BP}}$	—	$T_{\mathrm{SM1}} + T_{\mathrm{SM2}} + 2T_{\mathrm{SMT}} + T_{\mathrm{PA2}} + 2T_{\mathrm{ET}} + 2T_{\mathrm{BP}}$

Table 5: Computation cost on each entity.

	Computation cost (ms)		
	Setup and register	Intradomain authentication	Cross-domain authentication
UE		23.866	25.754
EA	—	15.872	—
DM	85.067	—	34.538

Table 6: Time costs (in s) of the blockchain.

Operations	Registration	Query	Transfer
Max time (s)	2.19	0.06	2.23
Min time (s)	0.03	0.01	0.03
Avg time (s)	0.18	0.01	0.16

Table 7: Comparison of time cost cryptographic operations in authentication.

Scheme	User	Server
A^2 Chain	$T_{SM1} + T_{SM2} + 2T_{SMT} + T_{PA2} + 2\,T_{ET} + 2T_{BP}$	$T_{SM1} + T_{SM2} + 2T_{SMT} + T_{PA2} + 2T_{ET} + 2T_{BP}$
ES^3A [25]	$6T_{SM1} + 3T_{SM2} + 3T_{BP} + T_{ET}$	$3T_{SM1} + 8T_{BP} + 4T_{ET}$
CPAL [27]	$16T_{ET} + 7T_{BP}$	$10T_{ET} + 7T_{BP}$
LCCH [54]	$30T_{E1}$[1] $+ 8T_{ET} + 8T_{BP}$	$23T_{E1} + 6T_{ET} + 5T_{BP}$
E-AUA [55]	$2T_{SM1} + 3T_{PA1} + T_{E1}$	$2T_{BP} + 2T_{E1} + T_H$

[1]T_{E1} indicates the exponentiation in G_1, and the meanings of other symbols are similar to the above definition.

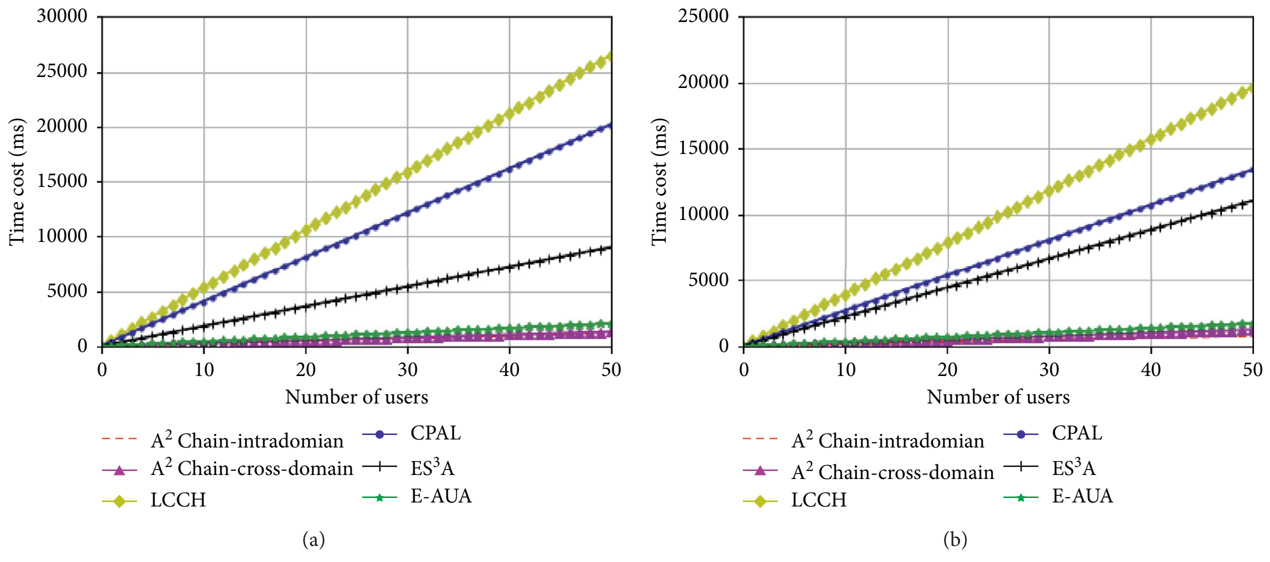

Figure 7: Comparison of computational overhead. (a) Cost on users for authentication. (b) Cost on servers for authentication.

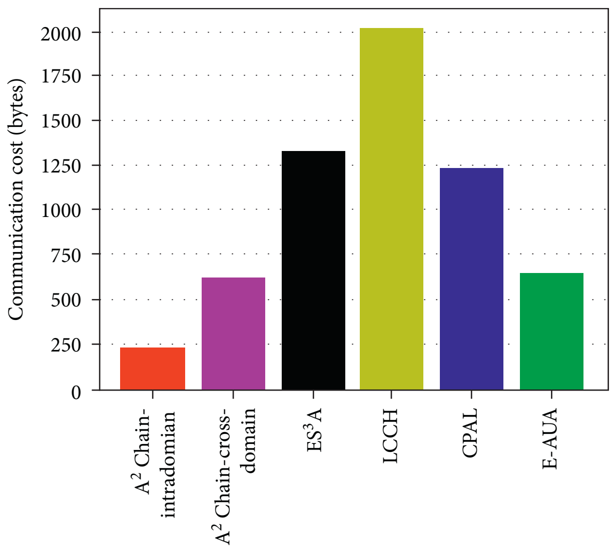

FIGURE 8: Communication cost comparisons of different schemes.

devices are generated by the domain management nodes. In our scheme, we assume that the domain management node is honest and trustworthy. In practice, there may be a malicious domain management node or a domain management node that is controlled by an adversary. In this case, it may lead to the leakage of the device's private key and jeopardize the security of the IoT application. In future work, we need to investigate the signature algorithm in case the key generation center is not fully trustworthy.

8. Conclusions

In this paper, we propose an application domain blockchain-alliance blockchain combined decentralized IoT authentication scheme called A^2 Chain, which enables a secure authentication information sharing process. Simulation results show that the scheme can significantly shorten the authentication time and reduce the communication cost and storage space compared to existing IoT authentication methods. In addition, we deploy the authentication server at the edge of the network through edge computing technology, which greatly reduces the authentication time and network latency.

Data Availability

All relevant data are included within the article.

Conflicts of Interest

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

Acknowledgments

The authors would like to thank the Guangzhou University for the equipment support and the National Natural Science Foundation of China for the support. This research was supported by the National Natural Science Foundation of China (Grant No. 61976064), Project of National Defence Science, and Technology Innovation Zone (Grant No. 18-H863-01-ZT-005-027-02), and Equipment Pre-research Key Laboratory Fund Project (61421030203).

References

[1] ITU-RM.2083-0, *IMT Vision Framework and Overall Objectives of the Future Development of IMT for 2020 and Beyond*, International Telecommunication Union, Geneva, Switzerland, 2015.

[2] J. Cao, P. Yu, M. Ma, and W. Gao, "Fast authentication and data transfer scheme for massive NB-IoT devices in 3GPP 5G network," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1561–1575, 2019.

[3] J. Cao, P. Yu, X. Xiang, M. Ma, and H. Li, "Anti-quantum fast authentication and data transmission scheme for massive devices in 5G NB-IoT system," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9794–9805, 2019.

[4] J. Li, M. Wen, and T. Zhang, "Group-based authentication and key agreement with dynamic policy updating for MTC in LTE-A networks," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 408–417, 2016.

[5] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Blockchain for 5G and beyond networks: a state of the art survey," *Journal of Network and Computer Applications*, vol. 166, Article ID 102693, 2020.

[6] J. Xing zhong, X. Qingshui, M. Haifeng, C. Jiageng, and Z. Haozhi, "The research on identity authentication scheme of internet of things equipment in 5G network environment," in *Proceedings of the International Conference on Communication Technology, ICCT*, pp. 312–316, Xi'an, China, October 2019.

[7] I. Psaras, "Decentralised edge-computing and IoT through distributed trust," in *Proceedings of the MobiSys 2018—16th ACM International Conference on Mobile Systems, Applications, and Services*, pp. 505–507, Munich, Germany, June 2018.

[8] S. Behrad, E. Bertin, S. Tuffin, and N. Crespi, "A new scalable authentication and access control mechanism for 5G-based IoT," *Future Generation Computer Systems*, vol. 108, pp. 46–61, 2020.

[9] N. K. Pratas, S. Pattathil, C. Stefanovic, and P. Popovski, "Massive machine-type communication (mMTC) access with integrated authentication," in *Proceedings of the 2017 IEEE International Conference on Communications (ICC)*, pp. 1–6, Paris, France, May 2017.

[10] M. Nasimi, M. A. Habibi, B. Han, and H. D. Schotten, "Edge-assisted congestion control mechanism for 5G network using software-defined networking," in *Proceedings of the 2018 15th International Symposium on Wireless Communication Systems (ISWCS)*, pp. 1–5, Lisbon, Portugal, August 2018.

[11] S. Hong, "P2P networking based internet of things (IoT) sensor node authentication by Blockchain," *Peer-to-Peer Networking and Applications*, vol. 13, no. 2, pp. 579–589, 2020.

[12] Z. Nezami, K. Zamanifar, K. Djemame, and E. Pournaras, "Decentralized edge-to-cloud load-balancing: service placement for the Internet of Things," 2020, http://arxiv.org/abs/2005.00270.

[13] M. El-Hajj, A. Fadlallah, M. Chamoun, and A. Serhrouchni, "A survey of internet of things (IoT) authentication schemes," *Sensors*, vol. 19, no. 5, pp. 1141–1143, 2019.

[14] L. Kou, Y. Shi, L. Zhang, D. Liu, and Q. Yang, "A lightweight three-factor user authentication protocol for the information perception of IoT," *Computers, Materials & Continua*, vol. 58, no. 2, pp. 545–565, 2019.

[15] C. Ellison and B. Schneier, "Ten risks of PKI: what you are not being told about public key infrastructure," *Public Key Infrastructure: Building Trusted Applications and Web Services*, vol. 14, no. 1, pp. 299–306, 2004.

[16] Z. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, "When mobile blockchain meets edge computing," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 33–39, 2018.

[17] S. Guo, X. Hu, S. Guo, X. Qiu, and F. Qi, "Blockchain meets edge computing: a distributed and trusted authentication system," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1972–1983, 2020.

[18] R. Almadhoun, M. Kadadha, M. Alhemeiri, M. Alshehhi, and K. Salah, "A user authentication scheme of IoT devices using blockchain-enabled fog nodes," in *Proceedings of the 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA)*, pp. 1–8, Aqaba, Jordan, October-November 2018.

[19] T. Salman, M. Zolanvari, A. Erbad, R. Jain, and M. Samaka, "Security services using blockchains: a state of the art survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 858–880, 2019.

[20] T. Hewa, A. Braeken, M. Ylianttila, and M. Liyanage, "Blockchain based Automated Certificate Revocation for 5G IoT," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Dublin, Ireland, June 2020.

[21] X. Jia, N. Hu, S. Su et al., "IRBA: an identity-based cross-domain authentication scheme for the internet of things," *Electronics*, vol. 9, no. 4, p. 634, 2020.

[22] K. Kaur, S. Garg, G. Kaddoum, F. Gagnon, and S. H. Ahmed, "Blockchain-based lightweight Authentication mechanism for vehicular fog infrastructure," in *Proceedings of the 2019 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, Shanghai, China, May 2019.

[23] R. Mahmoud, T. Yousuf, F. Aloul, and I. Zualkernan, "Internet of things (IoT) security: Current status, challenges and prospective measures," in *Proceedings of the 2015 10th International Conference for Internet Technology and Secured Transactions, ICITST 2015*, pp. 336–341, London, UK, December 2015.

[24] A. Esfahani, G. Mantas, R. Matischek et al., "A lightweight Authentication mechanism for M2M communications in industrial IoT environment," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 288–296, 2019.

[25] J. Ni, X. Lin, and X. S. Shen, "Efficient and secure service-oriented authentication supporting network slicing for 5G-enabled IoT," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 644–657, 2018.

[26] H. Gross, M. Hölbl, D. Slamanig, and R. Spreitzer, "Privacy-Aware Authentication in the Internet of Things," in *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, M. Reiter and D. Naccache, Eds., vol. 9476, pp. 32–39, Springer International Publishing, Cham, Switzerland, 2015.

[27] C. Lai, H. Li, X. Liang et al., "CPAL: a conditional privacy-preserving authentication with access linkability for roaming service," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 46–57, 2014.

[28] M. T. Hammi, B. Hammi, P. Bellot, and A. Serhrouchni, "Bubbles of Trust: a decentralized blockchain-based authentication system for IoT," *Computers & Security*, vol. 78, pp. 126–142, 2018.

[29] Z. Bao, W. Shi, D. He, and K.-K. R. Chood, "IoTChain: a three-tier blockchain-based IoT security architecture," 2018, http://arxiv.org/abs/1806.02008.

[30] U. Khalid, M. Asim, T. Baker, P. C. K. Hung, M. A. Tariq, and L. Rafferty, "A decentralized lightweight blockchain-based authentication mechanism for IoT systems," *Cluster Computing*, vol. 23, no. 3, p. 2067, 2020.

[31] H. Liu, P. Zhang, G. Pu, T. Yang, S. Maharjan, and Y. Zhang, "Blockchain empowered cooperative authentication with data traceability in vehicular edge computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4221–4232, 2020.

[32] Z. Cui, F. Xue, S. Zhang et al., "A hybrid BlockChain-based identity authentication scheme for multi-WSN," *IEEE Transactions on Services Computing*, vol. 13, no. 2, p. 1, 2020.

[33] N. Shahin, R. Ali, S. Y. Nam, and Y.-T. Kim, "Performance evaluation of centralized and distributed control methods for efficient registration of massive IoT devices," in *Proceedings of the 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*, vol. 2018, pp. 314–319, Prague, Czech Republic, July 2018.

[34] O. Alphand, M. Amoretti, T. Claeys et al., "IoTChain: a blockchain security architecture for the Internet of Things," in *Proceedings of the IEEE Wireless Communications and Networking Conference, WCNC*, pp. 1–6, Barcelona, Spain, April 2018.

[35] C. Lin, D. He, N. Kumar, X. Huang, P. Vijayakumar, and K.-K. R. Choo, "HomeChain: a blockchain-based secure mutual authentication system for smart homes," *IEEE Internet of Things Journal*, vol. 7, no. 2, pp. 818–829, 2020.

[36] M. A. Jan, W. Zhang, M. Usman, Z. Tan, F. Khan, and E. Luo, "SmartEdge: an end-to-end encryption framework for an edge-enabled smart city application," *Journal of Network and Computer Applications*, vol. 137, pp. 1–10, 2019.

[37] M. Shen, H. Liu, L. Zhu et al., "Blockchain-assisted secure device authentication for cross-domain industrial IoT," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 942–954, 2020.

[38] M. Ma, G. Shi, and F. Li, "Privacy-oriented blockchain-based distributed key management architecture for hierarchical access control in the IoT scenario," *IEEE Access*, vol. 7, pp. 34045–34059, 2019.

[39] M. A. Xiaoting, M. A. Wenping, and L. I. U. Xiaoxue, "A cross domain authentication scheme based on blockchain technology," *Acta Electronica Sinica*, vol. 46, no. 11, pp. 2571–2579, 2018.

[40] Y. Chen, G. Dong, J. Bai, Y. Hao, F. Li, and H. Peng, "Trust Enhancement Scheme for Cross Domain Authentication of PKI System," in *Proceedings of the 2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 103–110, Guilin, China, October 2019.

[41] J. Cui, L. Wei, J. Zhang, Y. Xu, and H. Zhong, "An efficient message-authentication scheme based on edge computing for vehicular ad hoc networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1621–1632, 2019.

[42] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C.-H. Hsu, "Edge server placement in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 160–168, 2019.

[43] Y. Jiang, C. Wang, Y. Wang, and L. Gao, "A cross-chain solution to integrating multiple blockchains for IoT data management," *Sensors*, vol. 19, no. 9, pp. 2042–2060, 2019.

[44] G.-H. Hwang, P.-H. Chen, C.-H. Lu et al., “A multi-chain architecture with distributed auditing of sidechains for public blockchains,” in *Proceedings of the Blockchain-ICBC*, vol. 10974, pp. 47–60, Springer International Publishing, Honolulu, HI, USA, September 2018.

[45] A. Shamir, “Identity-based cryptosystems and signature schemes,” in *Advances in Cryptology*, vol. 196 LNCS, pp. 47–53, Springer Berlin Heidelberg, Berlin, Heidelberg, 1985.

[46] F. Yuan and Z. Cheng, “Overview on SM9 identity-based cryptographic algorithm,” *Journal of Information Security Research*, vol. 2, no. 11, pp. 1008–1027, 2016.

[47] A. Back, M. Corallo, and L. Dashjr, “Enabling blockchain innovations with pegged sidechains,” pp. 1–25, 2014, http://newspaper23.com/ripped/2014/11/, http://www.blockstream.com.sidechains.pdf.

[48] A. Garoffolo, D. Kaidalov, and R. Oliynykov, “Zendoo: a zk-SNARK verifiable cross-chain transfer protocol enabling decoupled and decentralized sidechains,” 2020, http://arxiv.org/abs/2002.01847.

[49] S. Nakamoto, “Bitcoin: a peer-to-peer electronic cash system,” 2009, https://bitcoin.org/en/bitcoin-paper.

[50] M. Li, H. Tang, A. R. Hussein, and X. Wang, “A sidechain-based decentralized authentication scheme via optimized two-way Peg protocol for smart community,” *IEEE Open Journal of the Communications Society*, vol. 1, pp. 282–292, 2020.

[51] “Hyperledger fabric,” https://www.hyperledger.org/projects/fabric.

[52] Remote authentication dial in user service (RADIUS).

[53] “yh-radius,” https://github.com/cometowell/yh-radius.

[54] J. K. Liu, C.-K. Chu, S. S. M. Chow, X. Huang, M. H. Au, and J. Zhou, “Time-bound anonymous authentication for roaming networks,” *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 178–189, 2015.

[55] X. Zeng, G. Xu, X. Zheng, Y. Xiang, and W. Zhou, “E-AUA: an efficient anonymous user authentication protocol for mobile IoT,” *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1506–1519, 2019.

[56] “Hyperledger caliper,” https://github.com/hyperledger/caliper.