electronics

MDPI

Article

IRBA: An Identity-Based Cross-Domain Authentication Scheme for the Internet of Things

Xudong Jia, Ning Hu *, Shen Su, Shi Yin, Yan Zhao, Xinda Cheng and Chi Zhang

Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China; jiaxudong@e.gzhu.edu.cn (X.J.); johnsuhit@gmail.com (S.S.); 2111806065@e.gzhu.edu.cn (S.Y.); 2111906107@e.gzhu.edu.cn (Y.Z.); 2111906003@e.gzhu.edu.cn (X.C.); 2111906100@e.gzhu.edu.cn (C.Z.)

* Correspondence: huning@gzhu.edu.cn; Tel.: +86-138-7311-2138

Received: 4 March 2020; Accepted: 10 April 2020; Published: 11 April 2020

Abstract: The incredible development of Internet of things technology promotes the integration of application systems, which enable people to enjoy the convenience of multiple application services through a single intelligent device or terminal. In order to implement value exchange and information sharing between different applications, cross-domain access is inevitable. In order to prevent illegal access, identity authentication is necessary before the terminal accesses the service. Because of the need to introduce a trusted third party, the traditional centralized authentication model not only destroys the autonomy and flexibility of the application system, but also causes issues such as single point of failure and hidden dangers of unilateral control. This paper proposes an identity-based cross-domain authentication scheme for the Internet of Things. This scheme uses the Blockchain as a decentralized trust anchor instead of the traditional certificate of authority, and uses the identity-based self-authentication algorithm to replace the traditional PKI authentication algorithm. The scheme proposed in this paper implements a decentralized authentication model, which can guarantee the autonomy and initiative of the security domain.

Keywords: Internet of Things; decentralized authentication; blockchain; network security

1. Introduction

Internet of Things, also referred to as IoT, has become one of the most eye-catching technologies in recent years. In the immediate future, in 2025, more than 24.9 billion IoT connections are forecasted [1]. IoT brings huge innovation space and business value to many application fields, such as environmental monitoring, smart homes, smart cities, and so on. [2–6]. As the Internet of things technology is used more and more widely, security issues have also started to attract widespread attention. According to the report from Gartner, 20% of companies or organizations have experienced at least one IoT-based attack in the past three years [7].

Most IoT devices are vulnerable to malicious attacks, and the attack surfaces mainly include hardware, software, and protocols. The first is attack on the hardware. Because of the simple structure and low cost of the IoT device, it is relatively easy to be imitated and replaced. In addition, some devices are placed in unmanned areas, such as sensor nodes, which are vulnerable to physical damage. The second is attack on the software. An attacker can gain control of the terminal by hacking the operating system of the device or injecting malicious code, and illegally read user data stored on the device [8–10]. Once control of the IoT device is obtained, the device can be used as a springboard to further invade the back-end service system [11]. The last attack surface is attacking the protocol. Due to the limited capabilities of computing, storage, and communication, most IoT devices adopt lightweight designed communication protocols and identity authentication protocols. Because the key and encryption algorithm are too simple, the data is vulnerable to eavesdropping and tampering

during transmission, and the confidentiality and integrity of the data are difficult to guarantee. Because of the huge number of IoT devices and the wide attack surface, once IoT devices are maliciously controlled, a large-scale denial of service attack could be constituted, such as the Dyn event in 2016 [12].

Through the above analysis, it can be found that the IoT device is the key to achieving IoT security. Usually, to prevent unauthorized users from accessing the system, most IoT terminals use password-based authentication. Facts have shown that this approach has huge security risks. Taking the city camera as an example, due to the large number of cameras, it is difficult for the staff to set the passwords one by one, and most of them use the default initial password. Once the password of a single camera is cracked, it will cause other cameras to lose control [13]. Therefore, identity authentication for IoT terminal is considered a key issue to ensure the security of IoT systems [14].

With the continuous enrichment of IoT application, it is inevitable that there will be requirement for cross-domain access, value exchange and collaborative control among different application systems. Since different application systems have their own user space and autonomy, these requirements of cross-domain access caused cross-domain authentication issue. For example, the smart bracelet of the smart medical system needs to be connected to the smart home system to obtain the environmental parameters of the patient's living in order to provide reference data for the doctor's diagnosis. Since the bracelet needs to access smart home system and smart medical system at the same time, Cross-domain authentication is needed. The issue of cross-domain authentication is described as follows. Assume that there are multiple security domains in a network, and each domain has its own users and certificate authority. The goal of cross-domain authentication is to integrate these security domains so that the same user identity can log in to different domains and access resources and services in it.

The traditional identity authentication protocol based on the public key infrastructure, also known as PKI, is considered to be the most secure identity authentication method, but this type of scheme is not suitable for IoT terminals [15]. Firstly, most IoT terminals have limited resources in terms of energy, memory and processing capacity, the calculation, storage, and communication costs generated by this type of identity authentication protocol are too high to resource constrained IoT terminals [16,17]. Secondly, the traditional PKI based authentication model is a centralized authentication model, which has a single point of failure risk and is difficult to resist DOS / DDoS attacks [18,19]. Additionally, in the mMtc application scenario of 5G, the authentication peak caused by massive terminal accesses may also cause the authentication system to be paralyzed. Finally, due to political, economic and many other reasons, it is difficult to require all application systems to believe in a unified third-party organization. Compared with the centralized authentication model, the decentralized authentication model has no single point of failure, and has good scalability, which is considered as the development trend of the Internet of things authentication technology. Blockchain technology is regarded as an effective means to realize decentralized authentication [20].

In this paper, we propose an identity-based cross-domain authentication scheme for the Internet of Things. For convenience, our algorithm is called IRBA, which is consisted by the first letter of the four words Identity, Recognize, Blockchain, and Algorithm. IRBA explored the cross-domain access control problems caused by multiple IoT applications and proposed a novel solution with low cost, fast response, and anti-attack function, while eliminating the security risks brought by trusted third parties.

Compared with the previous research works, the innovations and contributions of IRBA are as follows:

1. Proposed a cross-domain access control oriented authentication method based on IBC (Identity-Based Cryptograph) algorithm.

   IRBA decomposes the cross-domain access control into two stages: identity authentication and access authorization. In the identity authentication stage, IRBA uses the identity of the IoT terminal to replace digital certificates issued by third parties and implements a decentralized identity authentication. Since every application domain can verify the authenticity of its identity through the identity of terminal, it does not need to rely on a third-party authentication server

during the authentication stage. Our method avoids the problem that IoT terminals need to maintain multiple digital certificates for different application domains.

2. Proposed a cross-domain joint authorization method based on threshold cryptographic algorithm.

   Using threshold cryptography algorithm, IRBA has designed a joint authorization method for cross-domain access. With this method, authentication servers in different application domains can jointly calculate the authorization signature and can independently verify the authorization signature. Therefore, the authorization process does not rely on trusted third parties. IRBA implements the authorization process through smart contracts to ensure the credibility of the authorization process. At the same time, the Blockchain is used to save the authorization results to ensure the authenticity of the authorization results. Through the above mechanism, IRBA achieves decentralized storage of access authorization results.

3. Proposed an implementation scheme and evaluated the effectiveness through the prototype system.

   We implemented a prototype system of IRBA based on two open source projects, which are Hyperledger Fabric and YH-RADIUS. Furthermore, we performed a performance evaluation of the core mechanism of IRBA. The experimental results show that IRBA has good processing performance and low computing overhead, which is very suitable for solving the cross-domain authentication problem of IoT.

The remainder of this article is organized as follows. Section 2 introduces the existing research and related work. Section 3 presents the motivation and our objectives. Section 4 gives a detail description of IRBA. Section 5 introduces the technical implementation of IRBA. Section 6 gives the results of performance evaluation. Section 7 summarizes of this article.

2. Related Work

2.1. Centralized Cross-Domain Authentication Scheme

Single sign-on is a typical representative of cross-domain authentication problems, and its purpose is to enable users to access data and services of different application systems through a one-time log in [21]. Single sign-on implements a type of federated identity management. By saving the user's identity in a trusted third party, when the user accesses the application system, the application system forwards the user's login request to the trusted third party. After the third party completes the authentication, the authorization code is returned to the application system and the user, and then the user uses the authorization code to access the application system. OAuth is the main solution for single sign-on [22]. OAuth implements cross-domain identity authentication in a proxy authentication manner instead of proposing an actual authentication algorithm.

Millán et al. construct a Bridge CA (BCA) model for cross-certification in the inter-domain networking [23]. The Bridge CA builds trust with several unrelated CAs. Each CA shares one or two cross certificates with BCA, thus establishing a trust relationship between CAs through BCA, a trusted independent node. Zhang et al. proposed a non-trusted center elliptic curve threshold signature to propose a virtual bridge CA-based virtual enterprise cross-domain authentication scheme, but due to the relatively large overhead cost, the scalability was not strong [24]. Yao et al. implemented the mutual authentication between PKI and Kerberos heterogeneous domains by using the bridge CA authentication model, but the certificate maintenance is complex, and the bridge CA management is difficult [25].

Due to the problem of certificate management in CA authentication system, researchers have conducted extensive research on identity-based cryptography (IBC). The authentication protocol proposed by Jiang et al uses the identity based cryptosystem, which takes the user's identity as the public key, does not need to store certificates, and simplifies the network configuration [26]. Li et al. proposed a certificate-based wireless mesh network cross-domain authentication key protocol to

implement mutual authentication and key exchange protocols between users [27]. Yuan et al. proposed a key agreement scheme for cross-heterogeneous domain authentication between the PKI domain and the IBC domain, but still bring about a large amount of calculation and communication [28].

2.2. Decentralized Cross-Domain Authentication Scheme

Certcoin, first proposed by Fromknecht et al., maintains the public ledger of the domain name and its related public keys [29]. The certificate issuance process is open and accessible to every user, which solves maintenance issues on single-point obstacles and certificate management in the traditional CA system. Ma et al. proposed a privacy-based Blockchain-based distributed key management scheme to achieve hierarchical access control [30]. Zhang et al. proposed the Town Crier (TC) system, which solves the problem of security authentication of its data source by providing the authenticated information for smart contracts [31]. Al-Bassam et al. proposed a PKI system based on decentralization and transparency to make malicious certificates easier to detect when they are issued [32]. The design of the trust network model enables the entities in the system to fine-granularity attributes of the identity of another entity verification becomes possible and realizes the trust transfer relationship between entity identity and entity attributes.

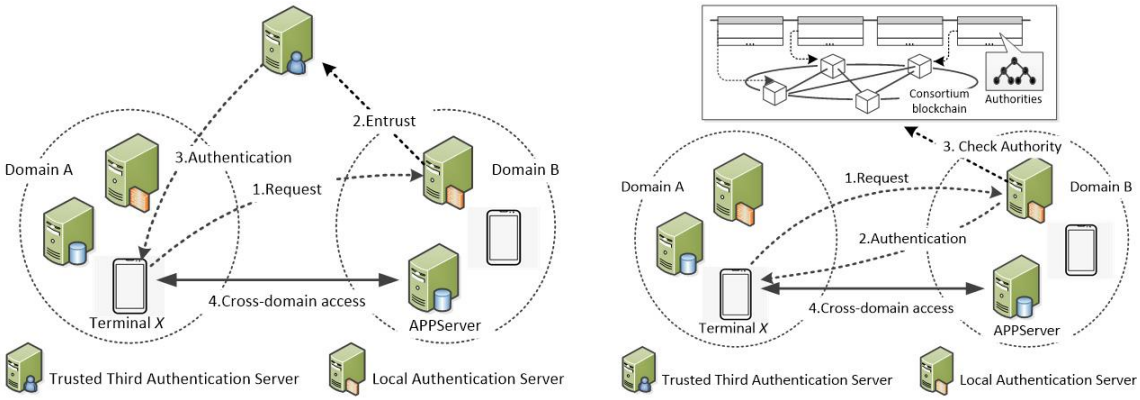
Zhou et al. proposed an efficient cross-domain authentication scheme based on Blockchain technology, and designed the trust model and system architecture of the Blockchain Certificate Authority (BCCA) [33]. In the BCCA trust model, the root CA joining the consortium chain is trusted, and the hash value of the certificate is recorded in the Blockchain to achieve safe and efficient cross-domain authentication. Chen et al. proposed a trust transfer scheme based on Blockchain to enhance trust and transfer [34]. This solution explores how to resolve the PKI unified trust service problem at the national level through consensus, transfers some CA management functions to the Blockchain, and builds the root CA of all security domains into a trust consortium. Trustroam is a Blockchain-based distributed authentication scheme for cross-domain roaming [35]. Different from the previous two schemes, the Trustroam verification process triggers a smart contract, and each verification is a transaction, instead of using the Blockchain as a database to query the Blockchain for data during the verification process. Ma et al. proposed a cross-heterogeneous domain authentication scheme based on Blockchain technology, in which the consortium chain model is composed of a Blockchain domain proxy server in the IBC domain and PKI domain Blockchain certificate server [36]. It enables secure and efficient communication between IBC and PKI heterogeneous domains.

3. Motivation and Objectives

Considering the vulnerability of IoT terminals in terms of security, most IoT application systems use identity authentication mechanisms to prevent malicious access by illegal terminals. When an IoT terminal needs to access the background services, it must first authenticate its identity through an authentication server. After confirming the authenticity of the terminal, the authentication server also needs to check the network access authorization result of the terminal. Only authorized users can be allowed to access. This process is often referred to as network access control. Access control not only occurs in a single application system, but also access control problems between different application systems. The problem of access control not only occurs in case of single application system, but also exists between different application systems. For example, with the popularity of wearable devices, people began to use these devices to replace wallets and ID cards. When users use wearable devices as identity cards or wallets to consume or enjoy services in different chain stores or supermarkets, cross-domain authentication problems exist. Because different application systems have their own user space and security policies, they are independent security autonomous domains. When customers in store *A* need to access the system of store *B*, cross-authentication requirement arise.

In order to solve the problem of cross-domain authentication, the traditional solution uses a centralized authentication model based on PKI. The authentication process is shown in Figure 1a. For convenience, we define an application system with a unified Certificate Authority (CA) and

secure access control policies as a security domain. In Figure 1a, terminal *X* represents a user of security domain *A*. When *X* needs to access the service of security domain *B*, it has to pass the authentication and authorization checks from the authentication server of domain *B*. Since there is no direct trust relationship between domain *A* and domain *B*, Domain *B* entrusts *C*, which is a trusted third party, to authenticate terminal *X* and perform an authorization check before allowing *X* to access. The centralized authentication model destroys the autonomy and independence of the application system, and there are hidden dangers of single points and unilateral control problems.



(**a**) Centralized Cross-domain model (**b**) Decentralized Cross-domain model

Figure 1. The model of cross-domain access.

For eliminating the disadvantages of the centralized authentication model, we hope to implement a decentralized authentication model, which is shown in Figure 1b. In Figure 1b, domain A and domain B establish a federal relationship and generate a user authorization list that allows cross-domain access, then jointly calculate the digital signature for the authorization list. After that, the authorization result is submitted to Blockchain to prevent repudiation and tampering. When terminal *X* accesses domain *B*, it uses a self-authentication identity authentication method instead of RSA authentication, so that *B* can complete the authentication of *X* identity without introducing a trusted third party. Next, the authentication server of domain *B* checks whether the authorization result stored on the Blockchain contains the terminal *X*. If it does, authentication succeeds otherwise fails. Compared with the centralized authentication model, decentralized authentication can guarantee the autonomy and initiative of the security domain. It does not need to rely on a third party to dynamically adjust the mutual trust relationship.

4. IRBA System

4.1. Basic Idea

Usually, the process of identity authentication mainly includes two stages: the first one is authentication stage, and the second one is authorization stage. In the authentication stage, the authentication server (*AS*) verifies the authenticity of the device (*UE*) identity. When it passes the authenticity verification, it enters the authorization stage. In the authorization stage, it mainly checks the network access rights of the device.

Figure 2 shows two typical authentication scenarios: intra-domain access and inter-domain access. In case of intra-domain access, both of the identity information and authorization information of the device are stored in the same server. Therefore, when the device wants to access intra domain services, the local authentication server can complete the whole authentication process. In case of inter-domain access, the situation is different. The identity information is stored locally, while the authorization information is stored in the remote domain. At this time, neither the local domain nor the remote domain can independently complete the authentication of the device. To adress this problem, the

traditional authentication model introduces a trusted third party. The basic idea is to integrate the identity and authorization of the device in the third party, and the third party completes the identity check and authorization check of the device. Obviously, this solution destroys the autonomy of the security domain and violates the privacy protection requirements of the security domain.

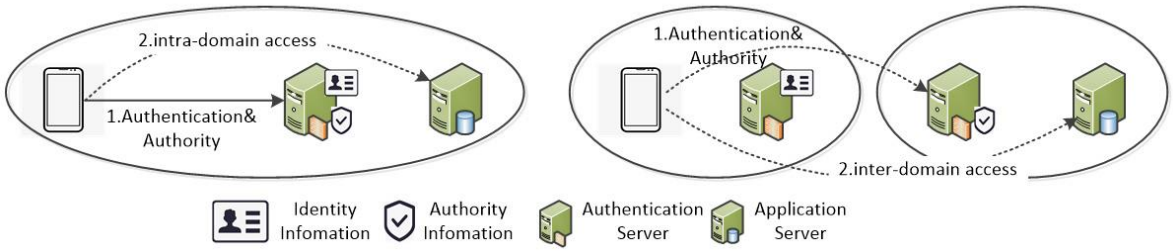


Figure 2. The model of intra-domain access and inter-domain access.

In this paper, we propose a decentralized and self-organized control model for cross domain access and the basic idea includes two aspects. The first aspect is to replace the public key-based authentication algorithm with the identity-based authentication algorithm. During the process of authentication, the authentication algorithm based on identity can determine the authenticity of user without a trusted third party. We will discuss the identity-based authentication algorithm in Section 4.2. The second basic idea is to take advantage of consensus mechanism and smart contract of Blockchain to replace the trusted third-party authorization process.

When a user's device wants to access to the service in other domain, the authentication servers of the local domain and remote domain will launch a transaction, and the transaction will trigger the execution of the Blockchain smart contract. In the smart contract, the authentication servers of both sides jointly calculate the signature for the authorization result and store it on the Blockchain. In this paper, we use the threshold cryptography algorithm to complete the signature calculation. We will also discuss threshold cryptography in the following section. Since the authorization results are jointly made by the security domains of both parties and stored on the Blockchain, it cannot be denied. By designing such a mechanism, we implement a cross-domain access authorization model without a trusted third party. In the following chapters, we will give a complete process description.

4.2. Signature Algorithm for Authentication

4.2.1. Identifier of Object

Before introducing the signature algorithm based on identity, we present the definition of ID. Object identifier (OID) is a coding scheme for identity proposed by ITU-T x.660. **OID** has many advantages, such as sufficient coding space, being independent of network technology, and not being affected by the underlying equipment. In this paper, we adopt **OID** as the identity of the device. The recommended **OID** structure is <Domain_ID.Category_ ID.Entity_ID>. Table 1 presents the field description of **OID.** The length of each field can be any value from 1 to 1,600,000 [37].

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, servers, etc.
Entity ID	M	Unique number assigned to the entity

4.2.2. Identity-Based Signature Algorithm

• Identity-based cryptosystem (IBC)

The traditional certificate-based security system involves a large number of key management operations, including certificate issuance, query and revocation. Identity-based cryptosystem (IBC) is a cryptosystem that allow entities uses some public information to be public keys, such as name, address and email [38]. Compared with the traditional cryptosystem, the greatest advantage of IBC is that it has no certificate, is easy to use and manage, and can easily achieve data encryption, identity authentication and other security services. Therefore, IBC has unique advantage in ensuring the data security of the Internet of Things, which can save overall cost substantially and effectively support the needs of identity authentication, data security, transmission security, access control, etc., in the application process of the Internet of things [39].

However, IBC also has an inherent disadvantage: the problem of private key escrow, a "trusted" private key generator (PKG) knows all users' private keys, so it can pretend to be any user to sign documents or decrypt encrypted messages. To overcome the private key escrow problem of PKG, Chen et al. [40] and Li et al [41] proposed the identity-based signature scheme without trusted PKG. Inspired by their work, we design an identity-based authentication scheme. In this scheme, each device has a key pair generated by ID as the public key. The authentication process can directly use the ID of the device to verify the signature without the participation of a third party. In the scheme, Authentication Server (*AS*) participates in the generation of keys as a PKG. We assume that the authentication server of each security domain uses the same public parameter and only the master key and the public key are different.

• Theory description of IBC

The Identity-based Signature Scheme mainly includes four stages: Setup, Extract, Signing and Verification. Algorithm 1 shows the IBC based authentication process.

Algorithm 1. Authentication Algorithm based on IBC

/*
    $\boldsymbol{G_1}$ is an additive group and $\boldsymbol{G_2}$ is a multiplicative group. The $\boldsymbol{P}$ and the $\boldsymbol{q}$ are
    the generator and order of the group, respectively.
    $H_1 : \{0,1\}^* \times G_1 \rightarrow G_1$, $H_2 : \{0,1\}^* \rightarrow G_1$, $H_3 : G_2^4 \rightarrow Z_q$
    $e : G_1 \times G_2 \rightarrow G_2$ is a bilinear mapping.
    $\forall P, Q \in G_1$ and $\forall a, b \in Z_q$, $e(aP, bQ) = e(P, Q)^{ab}$.
*/
Operation of ***AS*** (Authentication Server)
[Setup]:
    $Gen(P) \rightarrow (s, P_{Pub})$;
    $\{G_1, G_2, e, q, P, P_{Pub}, H_1, H_2, H_3\} \rightarrow params$;
[Extract]:
    $GenSK(x, params, ID_{UE}) \rightarrow Sk_{UE}$;
    [Verification]:
    $Ver(\sigma, params, ID_{UE}) \rightarrow valid/invalid$;

Operation of *UE* (User Device)
[Signing]:
    $SigID(m, params, Sk_{UE_1}) \rightarrow \sigma$;

The Setup and Extract are executed in the registration phase, and Signing and Verification are executed in the authentication phase. The registration phase is performed interactively by the

authentication server (*AS*) and the device (*UE*). We assume that the communication channel between *AS* and *UE* is private and secure at this phase.

1. **Setup**

   Step 1. The *AS* first picks group $\boldsymbol{G_1}$ and group $\boldsymbol{G_2}$. Among them, $\boldsymbol{G_1}$ is an additive group and $\boldsymbol{G_2}$ is a multiplicative group. The $\boldsymbol{P}$ and the $\boldsymbol{q}$ are the generator and order of the group, respectively. $e : G_1 \times G_2 \to G_2$ is a bilinear mapping, and $\forall P, Q \in G_1$ and $\forall x, y \in Z_q$, $e(xP, yQ) = e(P, Q)^{xy}$. Three hash functions $H_1 : \{0,1\}^* \times G_1 \to G_1$, $H_2 : \{0,1\}^* \to G_1$, and $H_3 : G_2^4 \to Z_q$. **Step 2.** At first, the *AS* randomly chooses the master private key $s \in Z_q^*$, and then computes the master public key through formula $P_{pub} = sP$. **Step 3.** The *AS* secret storage s.

2. **Extract**

   Step 1. The *UE* chooses an integer $r \in Z_q^*$, and computes $R = rp$. Then *UE* submits its identity information ID_{UE}, R, and using period t to the *AS*. **Step 2.** After receiving the message, *AS* computes $Q_{ID} = H_1(ID_{UE} \parallel t, R), S_{ID} = sQ_{ID}$. **Step 3.** The *AS* sends S_{ID} to the *UE*. (S_{ID},r) are the private key pair of *UE*, whose corresponding public key is ID_{UE}.

3. **Signing**

   Step 1. If *UE* wants to access the server, the *UE* first sends a authentication request to *AS*. **Step 2.** After the authentication request message is received, the *AS* generates a random number N. The *AS* then sends a response message to the *UE*. **Step 3.** The *UE* responds to the received message. *UE* computes $\theta = rH_2(m)$, $\sigma = e(H_2(N), S_{ID}), \omega = e(H_2(N), Q_{ID})$. $\varepsilon = Q + zS_{ID}, z = H_3(x, y, \omega, \sigma)$. **Step 4.**$(\theta, \sigma, R, \varepsilon)$ are the signatures of the message N. *UE* sends the signatures to *AS*.

4. **Verification**

   Step 1. The *AS* computes $Q_{ID} = H_1(ID_{UE} \parallel t, R), \omega = e(H_2(N), Q_{ID}), \mu = e\left(P_{pub}, Q_{ID}\right), z = H_1(x, y, \omega, \sigma)$. **Step 2.** If the equation $e(\theta, P) = e(H_2(N), R), e(P, \varepsilon) = x\mu^z, e(H_2(N), \varepsilon) = yv^z$ hold, the signature verification succeed.

• Security Analysis

We consider the following two situations:

1. As is untrustworthy

According to the working principle of PKG introduced earlier, the PKG server will computes S_{ID} and sends it back to *UE* as a partial private key. In case of *AS* is untrustworthy, the adversary can obtain the target *UE*'s public key and S_{ID} from the PKG server.

If the adversary wants to compute the corresponding V for a special message m, it has to compute the spoof without knowing the randomly element selected by *UE*. Obviously, under the assumption of CDHP in G_1 is intractable, the probability of an adversary successfully forging a valid signature is negligible [40–42].

2. *As* is trustworthy

In case of *AS* is trustworthy, the adversary cannot get the *UE*'*s* S_{ID} and from the PKG server. Since the randomly integer chosen by *UE* only exists t time. Under the assumption of CDHP, there is no algorithm that can generate fake signatures with a non-negligible probability. Nor the CDHP can be addressed.

4.2.3. Threshold Signature Algorithm

• Threshold signature

The idea of threshold signature was first proposed by Shamir [43] in 1979. It is used to obtain consensus or approval of a group of participants. The threshold signature scheme allows sharing

signing rights among n participants, each of whom has a private signing key. The signature requires a threshold of t ($t \leq n$) or more participants, and any group whose population less than t cannot generate a signature. The digital signature can be verified though public key [44].

We propose a threshold cryptography-based signature algorithm for cross-domain access authorization. The authorization for cross-domain access can only be generated when the threshold is reached. The verification process of threshold signature is the same as that of traditional signature, which will not affect the verification efficiency [40]. The algorithm mainly includes five parts, which are Setup, Extract. Private Key Distribution, Signing and Verification. Algorithm 2 shows the process of using threshold signature authorization.

Algorithm 2. Authentication Algorithm based on Threshold Signature

Authority Issuance
/* The process of [Setup] and [Extract] is the same as Authentication Algorithm */
[Private Key Distribution]:
    $GenTh()$: generates a set of n secret key shares $\{Sk_{ID}^1, Sk_{ID}^2, \ldots, Sk_{ID}^n\}$;
[Signing]:
    $SigTh\left(m, params, Sk_{ID}^i\right) \rightarrow \sigma_i$;
    $SigCom\left(\sigma_1, \sigma_2, \ldots, \sigma_k\right) \rightarrow \sigma, k \geq T.$;

Authority validation
[Verification]:
    $VerTh(\sigma, params, ID) \rightarrow valid/invalid$;

The signature scheme is described in detail as follows:

1. **Private Key Distribution:**

Step 1. The Method of public key setting is similar to the identity-based signature, except that the public and private key pair are generated using the Blockchain address of authority contract as the ID_{AC} after the authority contract is created.

Step 2. The authority contract distributes the private key to the n ASes to sign the authority.

1. The authority contract chooses $m_i \in_R Z_q, n_i \in_R G_1$,which $1 \leq i \leq t-1$.

2. Computes the polynomials:

$$h(x) = r + m_1 x + m_2 x^2 + \ldots + m_{t-1} x^{t-1}$$

$$H(x) = S_{ID} + n_1 x + n_2 x^2 + \ldots + n_{t-1} x^{t-1}$$

3. Computes the distribution key for each AS $h(i) = r_i, H(i) = \varepsilon_i$. The verification key corresponding to each key are $\lambda_i = r_i P, \mu_i = e(P, \varepsilon_i)$.

Step 3. Use the public key of AS_i encrypted and sent to AS_i $(1 < i < n)$. $h(x) = \sum_{j \in \Phi} z_{xj}^{\Phi} r_j, H(x) = \sum_{j \in \Phi} z_{xj}^{\Phi} \varepsilon_j$, where $z_{xj}^{\Phi} = \prod_{l \in \Phi, l \neq j} \frac{x-l}{j-l}, \Phi \subset \{1, 2, \ldots, n\}, |\Phi| \geq t$.

2. **Signing**

After receiving the signature request, AS_j $(1 < j < n, j \neq i)$ perform the following steps to sign the authorization information, assuming that the authorization information is M

Step 1. AS_j computes and broadcasts $\theta_j = r_j H_2(m), \sigma_j = e\left(H_2(m), \varepsilon_j\right), x_j = e\left(P, Q_j\right), y_j = e\left(H_2(m), Q_j\right)$, $Q_j \in G_1$ which is chosen randomly.

Step 2. AS_j Computes $x = \prod_{j\in\Phi} x_j^{z_{0j}^{\Phi}}, y = \prod_{j\in\Phi} y_j^{z_{0j}^{\Phi}}, \sigma = \prod_{j\in\Phi} \sigma_j^{z_{0j}^{\Phi}}$.

Step 3. AS_j computes and broadcasts $V_j = Q_j + z\varepsilon_j, \omega = e(H_2(M), Q_{ID})$ where $z = H_3(x, y, \omega, \sigma)$.

Step 4. AS_i verify the validity of partial signature. AS_i checks if $e\left(\theta_j, P\right) = e\left(H_2(m), l_j\right), e\left(P, V_j\right) = x_j\mu_j^z, e\left(H_2(M), V_j\right) = y_j\sigma_j^z, j \neq i$ hold. If some of the equations do not hold, the broadcast restarts the calculation.

Step 5. After the completion of partial signature verification, the authority contract collects partial signatures of AS, computes $\theta = \sum_{j\in\Phi} z_{0j}^{\Phi}\theta_j, \varepsilon = \sum_{j\in\Phi} z_{0j}^{\Phi} V_j$.

Step 6. $(\theta, \sigma, R, \varepsilon)$ are the signatures of the authorization information M.

3. **Verification**

Suppose the authentication server AS_k verifies the authority signature.

Step 1. The AS_k first computes

$$Q_{ID} = H_1(ID_{AC} \parallel t, R), \omega = e(H_2(M), Q_{ID}), \mu = e\left(P_{pub}, Q_{ID}\right), z = H_1(x, y, \omega, \sigma).$$

Step 2. If the equation $e(\theta, P) = e(H_2(M), R), e(P, \varepsilon) = x\mu^z, e(H_2(M), \varepsilon) = y\sigma^z$ hold, the signature verification succeeds.

- **Correctness Analysis**

Step 1. Note that

$$\theta = \sum\nolimits_{j\in\Phi} z_{0j}^{\Phi}\theta_j = \sum\nolimits_{j\in\Phi} z_{0j}^{\Phi} r_j H_2 M = rH_2(M)$$

Therefore, we have $\varepsilon = \sum_{j\in\Phi} z_{0j}^{\Phi} W_j = \sum_{j\in\Phi} c_{0j}^{\Phi}\left(Q_j + z\varepsilon_j\right)$

Step 2.

$$e(\theta, P) = e(H_2(M), R)$$

$$e(P, \varepsilon) = e\left(P, \sum\nolimits_{j\in\Phi} z_{0j}^{\Phi}\left(Q_j + z\varepsilon_j\right)\right) = e\left(P, \sum\nolimits_{j\in\Phi} z_{0j}^{\Phi} Q_j + \sum\nolimits_{j\in\Phi} z_{0j}^{\Phi}\varepsilon_j\right) = e\left(P, \sum\nolimits_{j\in\Phi} z_{0j}^{\Phi} Q_j\right) e(P, S_{ID})^z = x\mu^z$$

Step 3.

$$e(H_2(M), \varepsilon) = e\left(H_2(M), \sum\nolimits_{j\in\Phi} z_{0j}^{\Phi}\left(Q_j + z\varepsilon_j\right)\right) = e\left(H_2(M), \sum\nolimits_{j\in\Phi} z_{0j}^{\Phi} Q_j + \sum\nolimits_{j\in\Phi} z_{0j}^{\Phi}\varepsilon_j\right) = e(H_2(M), \sum\nolimits_{j\in\Phi} z_{0j}^{\Phi} Q_j e(H_2(M), S_{ID})^z = y\sigma^z$$

- **Security of Analysis**

The "Simulatability" of identity-based threshold signatures was defined by Baek [45]. Meanwhile he proved that the security of identity-based threshold signature depends on the identity-based signature sel.

Definition 1. *If an identity-based threshold signature satisfies the following conditions, the signature scheme can be simulated.*

1. "Private key distribution" can be simulated: The adversary's view can be simulate by the simulator when key distribution is executed under the condition that the public parameters, identity ID are known.

2."Signature" can be simulated: The adversary's view can be simulate by the simulator when the signature is executed by knowing the public parameters of the identity based threshold signature, the authorization information M and the corresponding signature τ, as well as the $t-1$ key shares and the corresponding public information for verification.

Theorem 1. *The threshold signature scheme based on identity is un-forgeable, while the signature scheme is un-forgeable and the identity-based threshold signature scheme can be simulated.*

In Reference [45], Theorem 1 has been proved. Therefore, the security of the identity-based threshold signature scheme only needs to prove that it can be simulated.

Lemma 1. *The threshold signature method based on identity in our scheme can be simulated.*

Proof.

Step1. let's suppose that the adversary corrupted the authentication server AS_i, where $1 \leq i \leq t-1$.

Step2. We firstly prove the "Private Key Distribution" can be simulated.

1. The adversary computes $\mu = e\left(P_{pub}, Q_{ID}\right)$ by system parameters *params* and the identity ID_{AC}

2. Since $\mu = \prod_{j=1}^{t} \mu_i^{z_{0j}^{\Phi}}$, the correct simulated value $\mu(t)$ can be computed by the adversary. It means that $\mu(t)$ is same as the result of "Private Key Distribution". The adversary also can compute the simulated value $r_t P$ correctly.

Step3. Then, we prove that the "signature" can be simulated.

1. The adversary computes $\theta_i = r_i H_2(M)$. Given the identity ID_{AC}, system parameters, authorization information M, $t-1$ private key shares (r_i, ε_i), corresponding signature $\tau = (\theta, \sigma, R, x, y, \varepsilon)$ and corresponding verification keys $(R, e(P, \varepsilon_i))$.

2. Suppose that $H(x)$ be a polynomial function of degree $t-1$, and $H(0) = \theta$, $(i) = \theta_i$ $1 \leqq i \leqq t-1$. The adversary can compute $\theta(i) = H(i)$, $t \leq i \leq n$. The adversary also can compute σ_i, x_i, V_i, $t \leq i \leq n$.

According to the security analysis of identity-based signature, Theorem 1 and Lemma 1, we can get the conclusion that the identity-based threshold signature scheme is un-forgeability. □

4.3. Authority Mechanism

4.3.1. Smart Contracts in IRBA

The smart contract is a kind of computer program which aims to spread, verify or execute the contract by way of code. Different to real contracts, smart contracts can carry out traceable, irreversible and secure transactions without the participation of third parties. All information related to the transaction is included in the smart contract, which can only be executed when the conditions are met.

In order to achieve the request and issue of cross-domain authority, we use the following three types of smart contracts:

- **The main contract** accepts authority requests and maintains the application list. There is only one main contract on the Blockchain, and all entities know its Blockchain address. To obtain a cross-domain authority, the authentication server should set up a new authority contract by using the master contract.
- **The authority contract** is created by the main contract and receives a signature. It enables authentication server to create threshold signatures by providing an exchange of publicly available random numbers.
- **The storage contract** is used as the recipient of a transaction that contains authority data and signatures.

4.3.2. Authority for Cross-Domain Access

When a device, UE_1, which belongs to domain D_1, wants to access the services of another domain, the procedure of authority is presented in Figure 3.

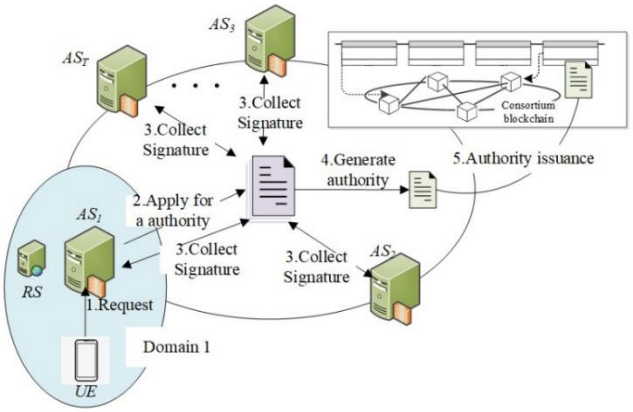


Figure 3. The procedure of authority for cross-domain access.

For the convenience of description, we define the symbols as shown in Table 2.

Table 2. Description of symbols.

Parameter	Description
$UE_{i:}$	UE in domain i
AS_i:	AS in domain i
$Sig_{Sk}()$:	Signing used Sk
ID_{UE_1}:	Public key of UE_i

Step 1. $UE_1 \rightarrow AS_1 : \left\{Request, ID_{UE_1}, Sig_{Sk_{UE_1}}(Requset)\right\}$

UE_1 requests the cross domain authority from AS_1 and uses the private key Sk_{UE_1} to generate signature $Sig_{Sk_{UE_1}}(Requset)$;

Step 2. After verifying the UE_1 request, AS_1 use the main contract to create an authority contract, and specify the AS address of the security domain to be accessed to collect signatures.

Step 3. The authority contract generates a key for signing and encrypts it with the public key of the ***AS*** in the specified domain, and distributes the encrypted signature key to the corresponding AS. The AS generates the partial signature and sends it to the authority contract.

Step 4. The authority contract collects signatures, combines the collected signatures into a complete authority signature.

Step 5. The storage contract packages the authority transaction into a block and stores it on the Blockchain.

4.4. Cross-Domian Authentication Process

It is assumed that the device UE_1 in the security domain D_1 initiates an authentication request to the authentication server AS_2 in security domain D_2. The authentication process is shown in Figure 4. The symbol definition in the authentication process is shown in Table 3. The procedure of protocol is described as follows:

1. $UE_1 \rightarrow AS_2 : \left\{Access\ request, ID_{UE_1}\right\}$ UE_1 initiate an authentication request to authentication server AS_2 in D_2.
2. $AS_2 \rightarrow UE_1 : \{N_1\}$ After receiving the request from UE_1, D_2 authentication server AS_2 responds to the request and sends a random number N_1 to D_1 device UE_1.
3. $UE_1 \rightarrow AS_2 : \left\{Sig_{Sk_{UE_1}}(N_1), N_1\right\}$

   (1) Device UE_1 receive the response from authentication server AS_2 using its private key Sk_{UE_1} to generate a signature $Sig_{Sk_{UE_1}}(N_1)$ of random number N_1;

(2) UE_1 responds to the request of AS_2 and send the signature $Sig_{Sk_{UE_1}}(N_1)$ and random number N_1 are sent to AS_2.

4. $AS_2 \rightarrow BC : \left\{ID_{UE_1}\right\}$

(1) AS_2 uses ID_{UE_1} to verify signature $Sig_{Sk_{UE_1}}(N_1)$.

(2) AS_2 queries the result of authority from Blockchain.

5. $BC \rightarrow AS_2 : \{Authority_1\}$

(1) If there is no authority of UE_1 in the query result, the authentication fails

(2) If the authority exists, check whether the validity period and the list of trusted domains allowed to access are valid. Verify the signature of "$Authority_1$". If the authentication succeeds, the authentication passes; otherwise, the authentication fails.

(3) If the authorization for UE_1 is not found, the authentication fails.

(4) If the authorization exists, check whether the validity period and the list of trusted domains that are allowed to be accessed are valid.

6. $AS_2 \rightarrow UE_1 : \{Auth - result\}$

AS_2 returns the authentication result of UE_1.

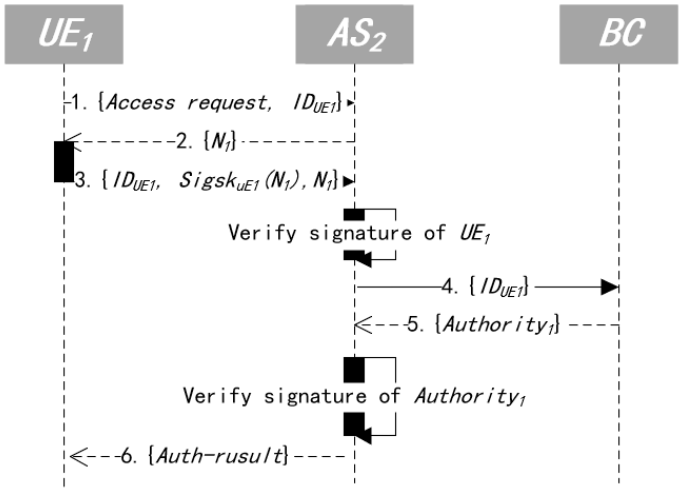


Figure 4. Cross-domain authentication process.

Table 3. Description of symbols.

Parameter	Description
$ID_{UE_1 i}$	Public key of UE_i
UE_i:	*UE in domain i*
AS_i:	*AS in domain i*
BC:	Blockchain
$Sig_{Sk}()$:	Signing used *Sk*
$Authority_i$:	Authority of UE_i

5. System Implementation

We develop a prototype system of IRBA with Go language. For convenience, we still use IRBA to represent the prototype system in this article. The IRBA consists of three parts. The first is the Blockchain platform. We adopt Hyperledger Fabric v1.0 [46] in IRBA. Since IRBA does not need to change the underlying mechanism of the Blockchain, not only Hyperledger Fabric, but other Blockchain platforms that support smart contracts can also be selected. The second part is smart contract, which is

called chain-code in Hyperledger Fabric. IRBA implements threshold cryptographic algorithms based on smart contracts. When the security domain wishes to establish a cross-domain trust relationship with other domains, it issues smart contracts through transactions. The third part is the authentication protocol. As we all know, Remote Authentication Dial In User Service (RADIUS) [47] is often used to build AAA servers, also known as Authenticate, Authority and Audit server. Therefore, we choose the open source project-YH-RADIUS [48]. This project implements an extensible development framework of RADIUS. The composition of the entire prototype system is shown in Figure 5.

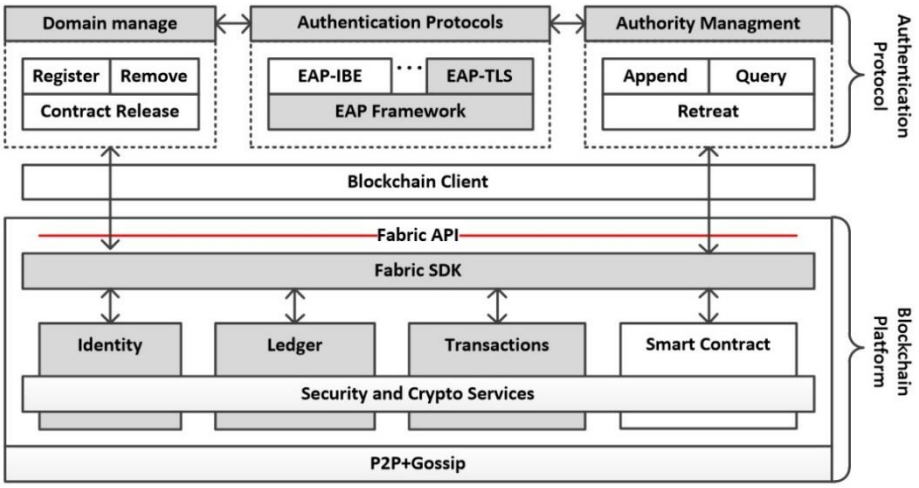


Figure 5. System architecture of Prototype system of IRBA.

Given that near-field communication protocols such as Bluetooth cannot support remote access, in our prototype system, only the case of TCP / IP-based networks is considered for the time being.

In the Figure 5, Extensible Authentication Protocol (EAP) is a universal authentication protocol standard framework running on TCP / IP networks [49]. This framework implements different authentication processes by encapsulating different protocol plug-ins. By using an identity-based signature algorithm, we implement the EAP-IBE protocol in IRBA. The module of domain management in Figure 5 is responsible for member registration, member removing, cross-domain request processing and smart contract release. When different security domains need to access each other, they release smart contracts through the domain management module, which is used to calculate joint authorization signatures of permission for cross-domain access. The model of authority management provides functions of appending, querying and retreating for cross-domain access authority. Because the Blockchain does not allow deletion of records that have been stored, both appending and retreating function will generate a new record with timestamp. When querying authorization results, the record with latest timestamp must prevail.

IRBA is installed on all authentication servers, providing full identity authentication and cross-domain access authentication capabilities. All IRBA-certified servers form a Fabric-based permission Blockchain [50]. Every authentication server can be configured as endorsement node, confirmation node and CA node. Each authentication server locally saves the identity information of members in the domain, and also maintained the distributed ledger and chain code. In order to ensure the reliability of the Blockchain system, the deployment of the sequencing service adopts the Kafka model of Fabric, which can prevent single points of failure of the sequencing nodes. Figure 6 presents a conceptual deployment scenario.

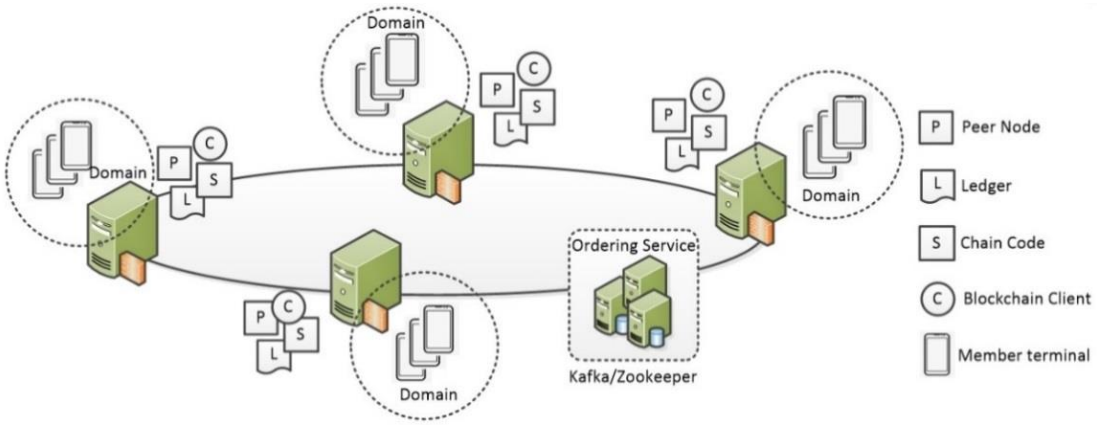


Figure 6. Conceptual deployment scenario of IRBA.

IRBA is composed of two subsystems which are authentication and access authorization. In the traditional AAA system implementation, the identity authentication subsystem interacts with the authorization subsystem through the RADIUS protocol. However, in IRBA, due to the special nature of smart contracts, the identity authentication subsystem calls the smart contract through the API of fabric SDK to complete the interaction.

Table 4 shows the schematic codes.

Table 4. Interaction with smart contract.

Schematic codes. Calling smart contract in IRBA
var Fabric_Client = require('fabric-client');
var channel = fabric_client.newChannel('newChannel');
var peer = fabric_client.newPeer('grpc://localhost:7051');
channel.addPeer(peer);
var order = fabric_client.newOrderer('grpc://localhost:7050')
channel.addOrderer(order);
var request = {
chaincodeId: 'AuthorityChaincode',
fcn: 'appendFunc',
args: ['targetDomianID', ' srcDomianID ', 'UserID'],
chainId: ' newChannel ',
txId: tx_id };
channel.sendTransactionProposal(request);

6. Performance Evaluation

6.1. Experiment Setup

To evaluating the performance of IRBA, we built an experimental environment as shown in Figure 7. In this experimental environment, there are ten PC servers with IRBA system and Hyper Ledger Fabric installed on them. Each of these servers represents an authentication server of a security domain. The authentication server is responsible for cross-domain access authorization and device cross-domain access authentication. These ten authentication servers form a consortium Blockchain. We also deploy an IoT device in every security domain with authentication client software installed on it. The configuration of our experiment is listed in Table 5.

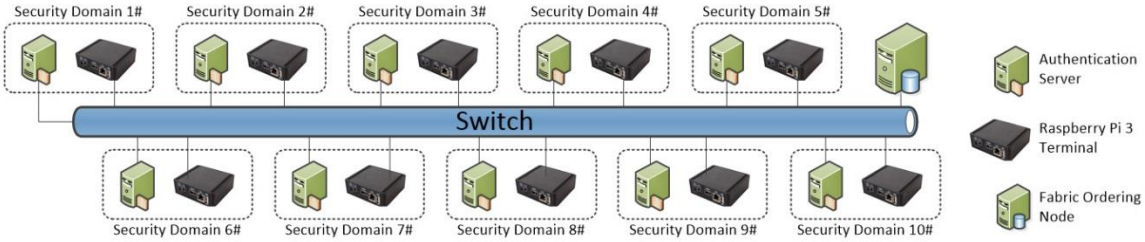


Figure 7. IRBA experimental topology.

Table 5. Configuration for the experiment.

Parameters	Values
Authentication servers	Intel-Core i5 6300 HQ CPU (2.30 GHz),16GB, Ubuntu 16.04
IoT device	ARM Cortex A53 (1.2 GHz),1 GB, Raspbian GNU/Linux 8
Switch	1Gbps*24
Block Size	30 transactions per block
Block Timeout	2 second

6.2. Processing Performance of Authority for Cross-Domain Access

In this experiment, we establish cross-domain access relationships from 2 to 10 security domains in turn, and observe the performance change of authorized signature calculations as the number of domains increases. For example, the first time, we have two security domains access each other and then we have three security domains access each other and so on, and finally expand to 10 security domains to access each other.

The results of this experiment are shown in Figure 8. From the chart, we found that the authority processing time does not exceed 5 seconds, and in most cases, the duration does not exceed 3 seconds, accounting for 60%–70%; meanwhile, the time required to issue an authority varies slightly with the threshold $\boldsymbol{T}$.

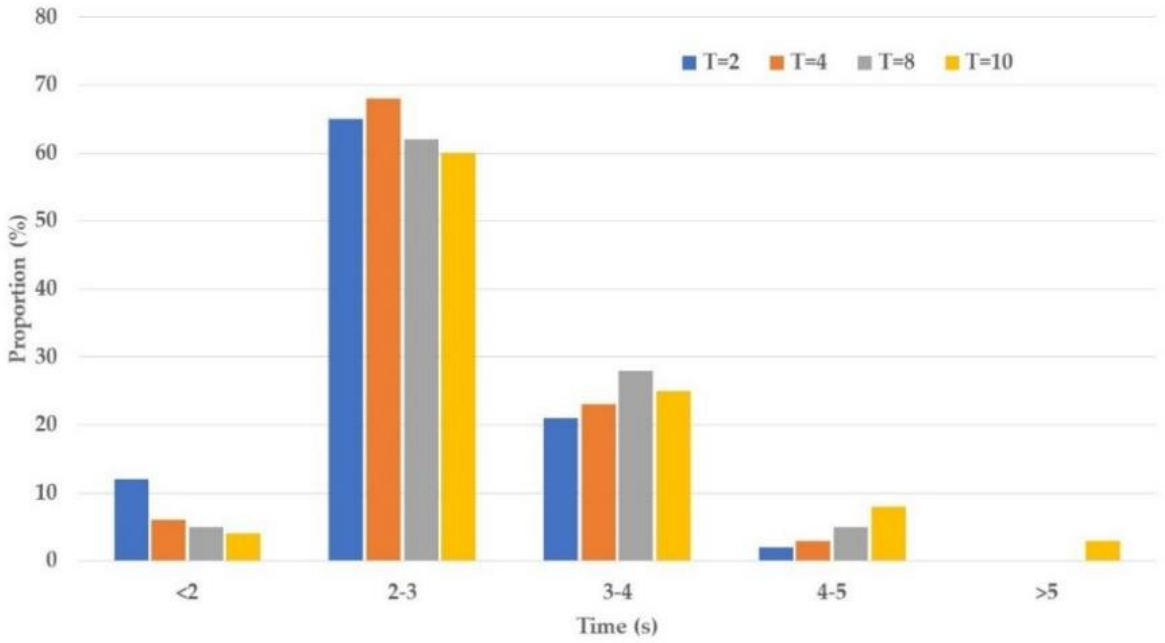


Figure 8. Distribution of authority time for different thresholds.

6.3. Processing Performance of Authorization Verify

In this experiment, we evaluate the processing performance of authorization verify. Similarly, we use the threshold $\boldsymbol{T}$ as a variable. The test cast is re-executed for 100 times and the average of the 100 samples is taken. The results obtained are shown in the Table 6.

Table 6. Authority verification time (TIDTV).

Count of Authentication Server(T)	Time (ms)
2	5.556
4	5.672
8	5.724
10	5.542

From the experiment results we found that the verification time does not change much with the change of threshold $\boldsymbol{T}$, which means IRBA has good scalability.

6.4. Processing Performance of the Authentication

In this section, we make a performance comparison between IRBA and three typical authentication methods, which include which include Enterprise Instant Messaging Authenticated Key Agreement Protocol (EIMAKP) [28], Blockchain Certificate Authority (BCCA) [33], and Cross heterogeneous domain authentication scheme (CHDA) [36]. Among these three methods, EIMAKP adopts centralized authentication scheme, BCCA uses Blockchain storage certificate for cross domain authentication and CHDA uses Blockchain to build a heterogeneous trust alliance between PKI and IBC domains. We will compare the performance of some cryptographic operations in the related protocols given by Ma et al [24]. The experiment results are listed in Table 7. The overall performance of the system is little affected by the performance evaluation in the registration phase and the running time of some lightweight operations, so we ignore them. Table 8 summarizes the calculation and communication costs of different schemes.

Table 7. Computation time consuming.

Description	Time (ms)
Identity-based signature (T_{IDS})	23.866
Identity-based signature verification (T_{IDV})	5.872
Asymmetric signature (T_{AS})	3.85
Asymmetric signature verification (T_{AV})	0.1925
Public-key-based encryption (T_{PE})	3.85
Public key-based decryption (T_{PD})	3.85
symmetrical encryption (T_{SE})	0.0046
Symmetric decryption (T_{SD})	0.0046
Scalar multiplication (T_M)	20226
Bilinear pairing (T_P)	5.811
$H_n : \{0,1\}^* \rightarrow Z_n$	0.0023
$H_P : \{0,1\}^* \rightarrow G_1$	12.418
$H_M : \{0,1\}^* \rightarrow G_2$	0.974
$H_S : \{0,1\}^* \rightarrow \{0,1\}^*$	0.0046

Table 8. Analysis for overhead of computation and communication.

Schemes		Computational Cost	Delay (ms)	Communication Cost (bit)
EIMAKP-I	User	$T_{IDV} + T_{AV} + 2T_{PE} + 2T_{PD} + T_{SE} + TSD + 3TM + TP + H_M$	116.25	1808
	Server	$T_{AV} + 2T_{PD} + T_{SE} + T_{SD} + T_{IDS} + 3T_{AS} + T_{AV} + 3T_{PE} + T_{PD} + T_{SE} + T_{SD} + 2_{TP} + 2H_M + H_P$		
EIMAKP-II	User	$T_{IDV} + T_{AS} + 2T_{PE} + T_{PD} + T_{SE} + T_{SD} + H_P$	111.05	1408
	Server	$T_{IDV} + 2T_{PD} + T_{SE} + T_{SD} + 2T_{IDS} + T_{AS} + 2T_{AV} + 2T_{PE} + T_{PD} + T_{SE} + T_{SD}$		
BCCA	User	$2T_{AS}$	32.92	2720
	Server	$2T_{AV} + H_S$		
CHDA-I	User	T_{PE}	96.98	1040
	Server	$T_{IDS} + T_{AV} + T_{PE} + T_{PD} + T_{IDV} + 3T_{AS} + 2T_{AV} + 4T_{PE} + 5T_{PD} + 4T_{M2Hn} + H_S$		
CHDA-II	User	T_{PE}	80.27	1040
	Server	$T_{IDV} + T_{AS} + T_{PD} + T_{IDS} + 2T_{AS} + 3T_{AV} + 4T_{PE} + 4T_{PD} + 2H_S$		
IRBA	User	T_{IDS}	40.502	592
	Server	$T_{IDV} + T_{IDTV}$		

In Figure 9, we can see that IRBA takes much less time than EIMAKP and CHDA. This is because EIMAKP and CHDA need several rounds of message exchange during the authentication process, which increases the calculation cost of the system. However, compared with BCCA, IRBA takes more time. This is because the IBC algorithm used by IRBA is more complex than the hash algorithm used by BCCA.

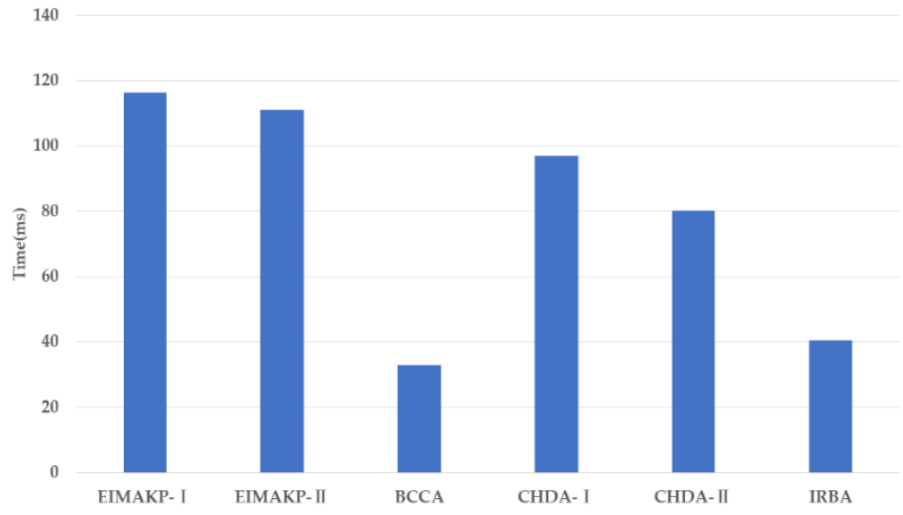


Figure 9. The total time consuming (ms)

In Figure 10, we found the cost of IRBA is lower than for other solutions. For EIMAKP and CHDA, because of multiple round of cipher text and cert exchange, the communication overhead is the highest of all. For IRBA, since the authority results are stored on the Blockchain, and the authentication server has the copy of ledge, less information needs to be exchanged. Therefore, IRBA is more suitable for resource-constrained IoT devices.

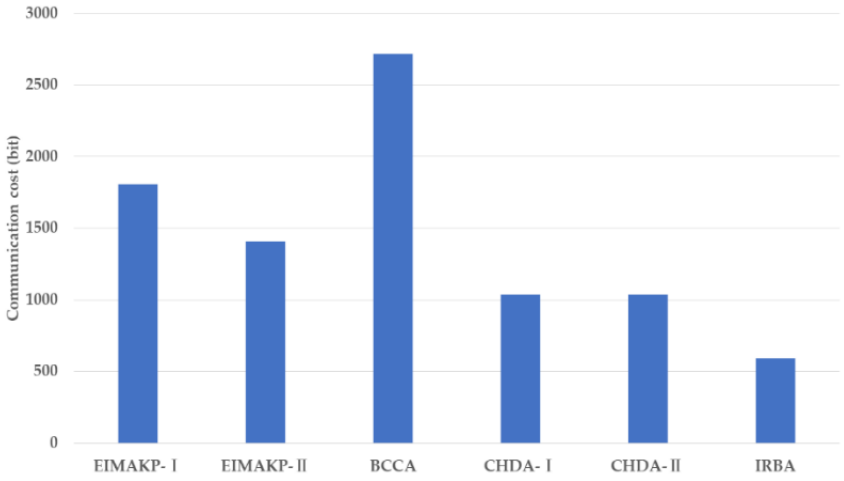


Figure 10. Total communication cost (bit).

7. Conclusions

In this paper, we propose an identity-based cross-domain authentication scheme for the Internet of Things. We have made innovations in the traditional authentication scheme, which include cross-domain authentication method based on IBC and a multi-domain joint authorization mechanism based on threshold cryptography and smart contracts. Through the combination of these methods, we implement a decentralized cross-domain authentication model. We have also developed a prototype system for the evaluation of information energy. Experimental results show that IRBA has good processing performance and flexibility, and it is very suitable for many scenarios of the IoT.

Author Contributions: Methodology: X.J. and N.H.; Project administration: N.H. and S.S.; Conceptualization: S.Y. and Y.Z.; Validation: X.C. and C.Z.; Funding acquisition: N.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (Grant No. 61976064, 61902083), Project of National Defense Science and Technology Innovation Zone (Grant No.18-H863-01-ZT-005-027-02), the Guangdong Province Key Research and Development Plan (Grant No. 2019B010137004); the National Key Research and Development Plan (Grant No. 2018YFB0803504); and Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2019).

Acknowledgments: The author would like to thank the equipment support of Guangzhou University and the support of National Natural Science Fund of China.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ericsson, A.B. Ericsson Mobility Report: On the Pulse of the Networked Society. *Ericsson*. Available online: https://www.ericsson.com/mobility-report (accessed on 10 August 2019).
2. Badshah, A.; Ghani, A.; Qureshi, M.A.; Shamshirband, S. Smart security framework for educational institutions using internet of things (IoT). *Comput. Mater. Contin.* **2019**, *61*, 81–101. [CrossRef]
3. Tian, Z.; Gao, X.; Su, S.; Qiu, J. Vcash: A Novel Reputation Framework for Identifying Denial of Traffic Service in Internet of Connected Vehicles. *IEEE Internet Things J.* **2019**, 1. [CrossRef]
4. Tian, Z.; Shi, W.; Wang, Y.; Zhu, C.; Du, X.; Su, S.; Sun, Y.; Guizani, N. Real-Time Lateral Movement Detection Based on Evidence Reasoning Network for Edge Computing Environment. *IEEE Trans. Ind. Inf.* **2019**, *15*, 4285–4294. [CrossRef]
5. Wang, B.; Kong, W.; Guan, H.; Xiong, N.N. Air Quality Forecasting Based on Gated Recurrent Long Short Term Memory Model in Internet of Things. *IEEE Access* **2019**, *7*, 69524–69534. [CrossRef]
6. Wang, B.; Kong, W.; Li, W.; Xiong, N.N. A dual-chaining watermark scheme for data integrity protection in internet of things. *Comput. Mater. Contin.* **2019**, *58*, 679–695. [CrossRef]

7. Gartner Gartner Says Worldwide IoT Security Spending Will Reach $1.5 Billion in 2018. Available online: https://www.gartner.com/en/newsroom/press-releases/2018-03-21-gartner-says-worldwide-iot-security-spending-will-reach-1-point-5-billion-in-2018 (accessed on 8 July 2019).
8. Thai, M.T.; Wu, W.; Xiong, H. *Big Data in Complex and Social Networks*; CRC Press: Boca Raton, FL, USA, 2016.
9. Tsiropoulou, E.E.; Baras, J.S.; Papavassiliou, S.; Qu, G. On the Mitigation of Interference Imposed by Intruders in Passive RFID Networks. In Proceedings of the International Conference on Decision and Game Theory for Security, New York, NY, USA, 2–4 November 2016; Volume 1, pp. 62–80. Available online: http://link.springer.com/10.1007/978-3-319-47413-7_4 (accessed on 2 February 2020).
10. Li, M.; Sun, Y.; Lu, H.; Maharjan, S.; Tian, Z. Deep Reinforcement Learning for Partially Observable Data Poisoning Attack in Crowdsensing Systems. *IEEE Internet Things J.* **2019**, *1*. [CrossRef]
11. Wikipedia Mirai (malware). Available online: https://en.wikipedia.org/wiki/Mirai_(malware) (accessed on 20 December 2019).
12. Wikipedia 2016 Dyn Cyberattack. Available online: https://en.wikipedia.org/wiki/2016_Dyn_cyberattack (accessed on 15 February 2020).
13. Alfred, N. Smart Cities around the World Were Exposed to Simple Hacks. Available online: https://www.cnet.com/news/smart-cities-around-the-world-were-exposed-to-simple-hacks/ (accessed on 5 April 2019).
14. Atzori, L.; Iera, A.; Morabito, G. The Internet of Things: A survey. *Comput. Netw.* **2010**, *54*, 2787–2805. [CrossRef]
15. Merino, A.S.; Matsunaga, Y.; Shah, M.; Suzuki, T.; Katz, R.H. Secure authentication system for public WLAN roaming. *Mob. Netw. Appl.* **2005**, *10*, 355–370. [CrossRef]
16. El-Hajj, M.; Fadlallah, A.; Chamoun, M.; Serhrouchni, A. A survey of internet of things (IoT) authentication schemes. *Sensors (Switz.)* **2019**, *19*, 1141. [CrossRef]
17. Kou, L.; Shi, Y.; Zhang, L.; Liu, D.; Yang, Q. A lightweight three-factor user authentication protocol for the information perception of IoT. *Comput. Mater. Contin.* **2019**, *58*, 545–565. [CrossRef]
18. Cao, B.; Li, Y.; Zhang, L.; Zhang, L.; Mumtaz, S.; Zhou, Z.; Peng, M. When Internet of Things Meets Blockchain: Challenges in Distributed Consensus. *IEEE Netw.* **2019**, *33*, 133–139. [CrossRef]
19. Hammi, M.T.; Hammi, B.; Bellot, P.; Serhrouchni, A. Bubbles of Trust: A decentralized blockchain-based authentication system for IoT. *Comput. Secur.* **2018**, *78*, 126–142. [CrossRef]
20. Salman, T.; Zolanvari, M.; Erbad, A.; Jain, R.; Samaka, M. Security services using blockchains: A state of the art survey. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 858–880. [CrossRef]
21. Radha, V.; Reddy, D.H. A Survey on Single Sign-On Techniques. *Procedia Technol.* **2012**, *4*, 134–139. [CrossRef]
22. Hardt, D. *The OAuth 2.0 Authorization Framework*. 2013. Available online: https://datatracker.ietf.org/doc/rfc6749/ (accessed on 9 February 2020).
23. López Millán, G.; Gil Pérez, M.; Martínez Pérez, G.; Gómez Skarmeta, A.F. PKI-based trust management in inter-domain scenarios. *Comput. Secur.* **2010**, *29*, 278–290. [CrossRef]
24. Zhang, W.; Wang, X.; Khan, M.K. A virtual bridge certificate authority-based cross-domain authentication mechanism for distributed collaborative manufacturing systems. *Secur. Commun. Netw.* **2015**, *8*, 937–951. [CrossRef]
25. Yao, Y.; Xingwei, W.; Xiaoguang, S. A Cross Heterogeneous Domain Authentication Model Based on PKI. In Proceedings of the 2011 Fourth International Symposium on Parallel Architectures, Algorithms and Programming, Tianjin, China, 9–11 December 2011; pp. 325–329.
26. Jiang, X.; Liu, M.; Yang, C.; Liu, Y.; Wang, R. A blockchain-based authentication protocol for WLAN mesh security access. *Comput. Mater. Contin.* **2019**, *58*, 45–59. [CrossRef]
27. Li, Y.; Chen, W.; Cai, Z.; Fang, Y. CAKA: A novel certificateless-based cross-domain authenticated key agreement protocol for wireless mesh networks. *Wirel. Netw.* **2016**, *22*, 2523–2535. [CrossRef]
28. Yuan, C.; Zhang, W.; Wang, X. EIMAKP: Heterogeneous Cross-Domain Authenticated Key Agreement Protocols in the EIM System. *Arabian J. Sci. Eng.* **2017**, *42*, 3275–3287. [CrossRef]
29. Fromknecht, C.; Velicanu, D.; Yakoubov, S. A Decentralized Public Key Infrastructure with Identity Retention. *IACR Cryptology ePrint Archive* **2014**, *803*, 1–16.
30. Ma, M.; Shi, G.; Li, F. Privacy-Oriented Blockchain-Based Distributed Key Management Architecture for Hierarchical Access Control in the IoT Scenario. *IEEE Access* **2019**, *7*, 34045–34059. [CrossRef]

31. Zhang, F.; Cecchetti, E.; Croman, K.; Juels, A.; Shi, E. Town crier: An authenticated data feed for smart contracts. In Proceedings of the ACM Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 270–282. [CrossRef]
32. Al-Bassam, M. SCPKI: A Smart Contract-based PKI and Identity System. In *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts-BCC' 17*; ACM Press: New York, NY, USA, 2017; pp. 35–40.
33. Zhou, Z.; Li, L.; Li, Z. Efficient cross-domain authentication scheme based on blockchain technology. *J. Comput. Appl.* **2018**, *38*, 316–320.
34. Chen, Y.; Dong, G.; Bai, J.; Hao, Y.; Li, F.; Peng, H. 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, Guilin, China, 17–19 October 2019; pp. 103–110. Available online: https://ieeexplore.ieee.org/document/8945998/ (accessed on 9 October 2019).
35. Li, C.; Wu, Q.; Li, H.; Liu, J. *Trustroam: A Novel Blockchain-Based Cross-Domain Authentication Scheme for Wi-Fi Access*; Springer International Publishing: Berlin/Heidelberg, Germany, 2019; Volume 2, pp. 149–161. ISBN 9783030235970.
36. Ma, X.T.; Ma, W.P.; Liu, X.Y. A Cross Domain Authentication Scheme Based on Blockchain Technology. *Acta Electron. Sin.* **2018**, *46*, 2571–2579.
37. ITU-T. *ITU-T X.660—Supplement on Guidelines for Using Object Identifiers for the Internet of Things*. Available online: https://www.itu.int/rec/T-REC-X.Sup31-201709-I (accessed on 30 October 2019).
38. Shamir, A. Identity-Based Cryptosystems and Signature Schemes. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; 196 LNCS. Springer: Berlin/Heidelberg, Germany, 1985; pp. 47–53. Available online: http://link.springer.com/10.1007/3-540-39568-7_5 (accessed on 9 April 2019).
39. Neto, A.L.M.; Souza, A.L.F.; Cunha, I.; Nogueira, M.; Nunes, I.O.; Cotta, L.; Gentille, N.; Loureiro, A.A.F.; Aranha, D.F.; Patil, H.K.; et al. AoT: Authentication and access control for the entire iot device life-cycle. In Proceedings of the 14th ACM Conference on Embedded Networked Sensor Systems, SenSys, Stanford, CA, USA, November 2016; pp. 1–15. [CrossRef]
40. Chen, X.; Zhang, F.; Konidala, D.M.; Kim, K. New ID-Based Threshold Signature Scheme from Bilinear Pairings. *Informatica*. 2004, 20, pp. 371–383. Available online: http://link.springer.com/10.1007/978-3-540-30556-9_29 (accessed on 1 February 2020).
41. Xu, L.; Wang, M. New ID-Based Signatures without Trusted PKG. In Proceedings of the First International Workshop on Knowledge Discovery and Data Mining (WKDD 2008), Adelaide, Australia, 23–24 January 2008; pp. 589–593.
42. Boneh, D.; Lynn, B.; Shacham, H. Short Signatures from the Weil Pairing. In *International Conference on the Theory and Application of Cryptology and Information Security*; Okamoto, T., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2001; pp. 514–532. ISBN 978-3-540-41404-9.
43. Shamir, A. How to Share a Secret. *Commun. ACM* **1979**, *22*, 612–613. [CrossRef]
44. Farley, N.; Fitzpatrick, R.; Jones, D. BADGER—Blockchain Auditable Distributed (RSA) key GEneRation. *IACR Cryptol. ePrint Arch.* **2019**, *1*, 1–16.
45. Baek, J.; Zheng, Y. Identity-based threshold signature scheme from the bilinear pairings (extended abstract). In Proceedings of the International Conference on Information Technology: Coding Computing, ITCC, Las Vegas, NV, USA, 5–7 April 2004; Volume 1, pp. 124–128.
46. Hyperledger Fabric. Available online: https://www.hyperledger.org/projects/fabric (accessed on 30 October 2018).
47. Remote Authentication Dial In User Service (RADIUS). Available online: https://datatracker.ietf.org/doc/rfc2865/ (accessed on 6 October 2019).
48. Yh-Radius. Available online: https://github.com/cometowell/yh-radius (accessed on 2 January 2020).
49. Extensible Authentication Protocol (EAP). Available online: https://tools.ietf.org/html/rfc3748 (accessed on 1 January 2020).
50. Guo, S.; Hu, X.; Guo, S.; Qiu, X.; Qi, F. Blockchain Meets Edge Computing: A Distributed and Trusted Authentication System. *IEEE Trans. Ind. Inf.* **2020**, *16*, 1972–1983. [CrossRef]