BCPP-IAS : Blockchain-Based Cross-Domain Identity Authentication Scheme for IoT with Privacy Protection

Qianqian Li[1], Yubing Han[1,2(✉)], Guijuan Wang[2,3], Li Zhang[1,2], and Jiguo Yu[2]

[1] School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, Shandong , People's Republic of China
lqyxlq@163.com

[2] Big Data Institute, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, Shandong, People's Republic of China
jiguoyu@sina.com

[3] Shandong Provincial Key Laboratory of Computer Networks, Shandong Fundamental Research Center for Computer Science, Jinan, China
guijuan_wang@126.com

Abstract. In large-scale multi-domain Internet of Things (IoT) environments, secure communication and efficient identity authentication between IoT devices are of increasing importance. Traditional centralized identity authentication schemes commonly suffer from problems such as privacy leakage and single point of failure. What's worse, these conventional identity authentication schemes often generate digital certificates for users by certificate authority (CA), which will bring key escrowing and certificate management pressure to IoT devices with limited resources. This paper proposes a blockchain-based identity authentication scheme for IoT (BCPP-IAS) that is efficient, decentralized, and is able to tackle single point of failure. Additionally, a new certificateless aggregated signature algorithm is presented to address complex certificate management and key escrow problems. The edge server is capable of aggregating different signatures to achieve batch authentication, markedly improving authentication efficiency and reducing computational and storage overhead. Also, BCPP-IAS is designed to avoid costly bilinear pairing operations, providing less computational overhead for IoT devices with limited resources. To protect the privacy of IoT devices, BCPP-IAS uses pseudonyms instead of real identity. Finally, security and efficiency of BCPP-IAS are demonstrated through theoretical analysis and experiments.

Keywords: Cross-domain identity authentication · Blockchain · Certificateless aggregate signature · Internet of Things

1 Introduction

The IoT has broad relevance in our lives, being extensively utilized in various fields, including smart home [1], healthcare, intelligent transportation, and in-

dustrial manufacturing [2]. Gartner predicts a significant increase in IoT smart devices worldwide by 2025, estimating 41.6 billion in use [3].

The society has greatly benefited from the rapid advancement of IoT, which has brought immense convenience. However, concerns regarding security and privacy have also been raised as a result. Given this, the necessity for cross-domain authentication has become paramount. Within the IoT framework, devices belonging to distinct management domains are required to collaborate. To safeguard data integrity, access to sensitive information must be restricted solely to authorized domains, necessitating the anonymity of device identities [4] [5]. Through the implementation of cross-domain authentication schemes, trust and validity can be established among devices, ultimately facilitating the secure data [6].

Traditional cross-domain identity authentication solutions rely on public key infrastructure(PKI), which is a centralized facility that supports cross-domain authentication and allows relying parties to obtain a CA [7]. As the IoT system scales up, the workload of the certification service center increases dramatically, resulting in significant certificate management overhead [8]. However, this centralized nature also is susceptible to data leakage and single points of failure, which ultimately reduces the security of cross-domain identity authentication.

In recent years, blockchain has been widely utilized in various fields such as digital finance, the IoT, supply chain management, and intellectual property protection [9] [10]. It is characterized by decentralization, security, reliability, immutability, and traceability, thereby eliminating the reliance on third-party intermediaries. This technology has been particularly instrumental in providing innovative solutions for ensuring IoT identity security authentication. Blockchain have played a crucial role in enabling cross-domain identity authentication by facilitating the exchange of authorization information between devices in distinct trusted domain [11] [12]. By storing their public identities on the blockchain, devices from different trust domains can mitigate vulnerabilities associated with centralized facilities that are susceptible to single point of failure. Despite addressing the single point of failure issue, the adoption of blockchain does not eliminate the challenges associated with the costly and cumbersome certificate management and authentication efficiency found in traditional schemes.

To address the security and efficiency issues posed by traditional IoT identity authentication schemes [13], we have introduced BCPP-IAS, a novel blockchain-based cross-domain IoT identity authentication scheme. The key contributions of BCPP-IAS focus on resolving issues related to low authentication efficiency and high communication [14] overhead that have been identified in previous schemes.

1. We propose BCPP-IAS, a blockchain-based IoT identity authentication scheme which utilizing blockchain to record data sharing requests and signatures, avoiding single point of failure attacks, and uses pseudonyms to enable identity privacy protection for IoT devices.
2. We design a novel certificateless aggregation signature algorithm, which avoids expensive bilinear pairing operations, reduces the computational over-

head of IoT devices with limited resources, making it truth to cope with complex certificate management and key escrowning problems [3].
3. The security and high efficiency of BCPP-IAS are proved by theoretical analysis and experiments.

This article will delve into the relevant literature in Section 2 and describe the scheme framework and security requirements of BCPP-IAS in Section 3. Section 4 is the detailed design of the BCPP-IAS. Security analysis of BCPP-IAS in Section 5. In Section 6, performance analysis is carried out, and a comprehensive summary in Section 7.

2 Releted work

In this section, we discuss recent research focusing on secure cross-domain identity authentication with privacy preservation for IoT.

2.1 Cross-domain identity authentication scheme based on the combination of centralized technology and blockchain

In the realm of cross-domain identity authentication, the conventional methods frequently depend on a centralized facility for overseeing security services. PKI exemplifies a renowned approach in this context. Despite its widespread adoption across various domains for identity authentication purposes, a noteworthy limitation of PKI lies in the vulnerability to a single point of failure. In [15], all domains utilize PKI for identity authentication. The integration of a blockchain network for cross-domain identity authentication necessitates the involvement of each domain's CA. However, there appears to be insufficient consideration given to the compatibility of this identity authentication process IoT devices with limited resources. In [16], Zhou et al. created a domain for issuing certificates by utilizing identity encryption and secret sharing techniques. This innovative solution effectively tackles the identity authentication difficulties encountered in mobile ad hoc networks and the advancement of smart cities. In [17], a hierarchical multi-domain vehicle identity authentication scheme based on blockchain was proposed by Yang et al. The perception layer employs pseudonym-based authentication to ensure anonymity and traceability, while the management layer incorporates a PKI system. In [18], Fatemeh et al. proposed a innovative identity authentication framework that connects blockchain-based infrastructure with centralized networks in non-blockchain industries. This framework enables seamless cross-authentication among diverse sub-networks utilizing distinct protocols or authentication approaches, while preserving the transparency characteristics inherent to blockchain. In [19], Wu et al. proposed a framework called PPTMA which incorporates differential privacy techniques protecting sensitive user trust information from unauthorized access. Nevertheless, it should be noted that PPTMA's dependence on PKI for generating unique public and private keys per device introduces a potential vulnerability as it relies on a centralized system.While these PKI-based cross-domain identity authentication schemes have

made significant progress, they still pose identity authentication burdens and single point of failure for IoT devices with limited resources. Furthermore, the privacy protection and security aspects necessitate further enhancement.

2.2 Decentralized cross-domain identity authentication solution based entirely on blockchain

In [20], Feng et al. employed blockchain and threshold sharing multi-signature for identity trust in collaborative domains and cross-domain identity authentication. Nonetheless, the scheme's high number of communication rounds results in substantial communication overhead. In [8], Li et al. proposed a blockchain-based identity authentication and key agreement system that employs smart contracts for cross-domain purposes. The system utilizes smart contracts to handle the public key of each node and verifies system parameters through contract queries, thereby achieving authentication and key agreement. In [21], Mao et al. proposed a solution for enhancing trust among managers in cross-domain IoT systems by blockchain and Trusted Execution Environment (TEE) to establish a secure authentication scheme. In [22], Nakkar et al. proposed a lightweight group authentication protocol with a session key agreement, supporting multiple asynchronous identity authentications. In [23], Gong et al. proposed a lightweight identity authentication method called LCDMA, specifically designed for the mobile IoT setting. This technique employs symmetric polynomials as an alternative to the intricate bilinear pairing used in conventional approaches. These cross-domain identity authentication schemes are inefficient and inflexible.

3 System architecture

In this section, we present the system model and workflow of BCPP-IAS.

3.1 System Model

System model of BCPP-IAS mainly includes the following entities. The specific system model is shown in Fig. 1.

- KGC: Key Generation Center, the main role is to generate keys for users in the system to ensure the effectiveness of cryptographic algorithms.
- TO: Trusted organization, the main role is to help users to complete the registration.
- Blockchain: The blockchain is a distributed ledger, and the information recorded on it cannot be tampered with, bringing trust to the entire distributed system.
- IoT devices: IoT devices need identity authentication to restrict access to the network and secure communication between devices to only authorized ones.
- Edge Servers: Edge servers as administrators of each domain and their main role is to aggregate signatures, broadcast authentication requests, etc. Blockchain nodes are deployed on edge servers.

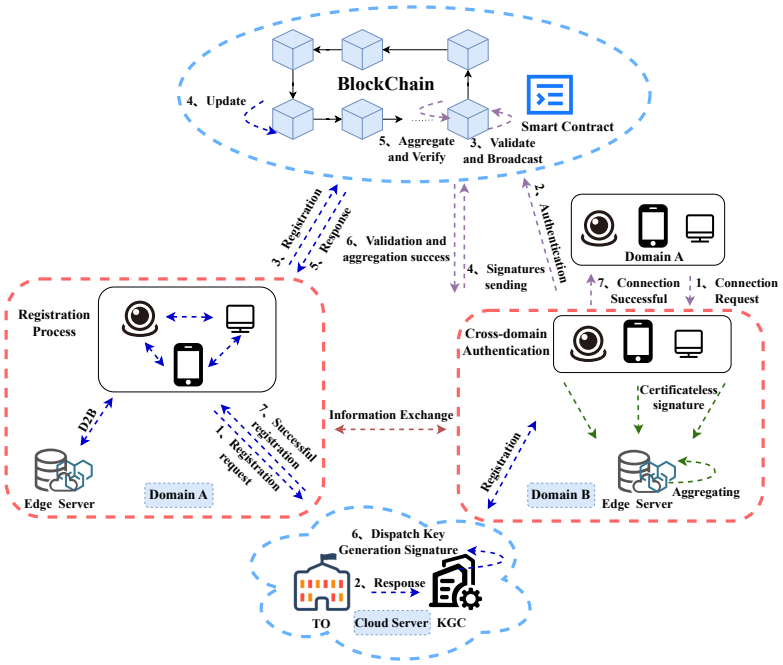


Fig. 1. System model of BCPP-IAS

3.2 BCPP-IAS Workflow

The workflow of BCPP-IAS is described in detail as follows.

1. Initialize the system and generate system parameters.
2. The IoT device first registers with the TO, which interacts with the KGC to help the IoT device complete the registration. TO solely assists users in registration and does not engage in the identity authentication process.
3. The IoT device accesses the edge server in its domain to record the authentication request on the blockchain.
4. The edge server of the requested domain broadcasts the authentication request to the entire domain.
5. The requested node generates a signature using a certificateless signature algorithm to send to the edge server in its own domain.
6. The edge server verifies the validity of individual signatures and aggregates all the signatures after successful verification.
7. The edge server uploads the aggregated signatures to the blockchain.
8. The requester verifies the validity of the aggregated signatures and realizes cross-domain identity authentication.

To facilitate the illustration of BCPP-IAS, Table I shows the notation used in BCPP-IAS and a detailed explanation.

Table 1. Notations and descriptions

Notations	Descriptions	Notations	Descriptions
q	A large prime number	T_0	The public key of TO
$\mathbb{G}_1$	An additive cyclic group	$\mathbb{G}_2$	A multiplicative cyclic group
P	The generator of $\mathbb{G}_1$	FID_i	Forged identity
e	A bilinear pairing	pk_i	The public key of user i
H_1, H_2, H_3	Three hash functions	PSK_i	The partial private key of user i
sk_i	The private key of user i	k	The private key of TO
P_0	The public key of KGC	σ	A certificateless signature
s	The private key of KGC	$\mathcal{C}$	The challenger
Adv_I	The first type of adversary	Adv_{II}	The second type of adversary

4 Detailed design of BCPP-IAS

In this section, we detail BCPP-IAS according to the following steps: 1) System Initialization, 2) User Key Generation, 3) Authentication Request and Generate Signature, 4) Aggregate Signature, and 5) Aggregate Signature Verification.

4.1 System Initialization

KGC generates the system parameters, including the additive group $\mathbb{G}_1$, the multiplicative group $\mathbb{G}_2$, and the generator P of $\mathbb{G}_1$. Subsequently, KGC randomly selects $s \in \mathbb{Z}_q^*$ ($\mathbb{Z}_q^*$ is a finite field of integers) as its private key. Then KGC computes $P_0 = sP$ to derive its public key. Following this, KGC proceeds to select three hash functions $\mathrm{H}_1 : \{0,1\}^* \times \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{Z}_q^*, \mathrm{H}_2 : \{0,1\}^* \times \mathbb{G}_1 \to \mathbb{Z}_q^*$, $\mathrm{H}_3 : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G}_1 \to \mathbb{Z}_q^*$. TO randomly selects $k \in \mathbb{Z}_q^*$ as private key and computes $T_0 = kP$ as public key. TO and KGC record the system parameters $sysparas = \{\mathbb{G}_1, \mathbb{G}_2, e, P, P_0, T_0, \mathrm{H}_1, \mathrm{H}_2, \mathrm{H}_3\}$ on the blockchain.

4.2 User Key Generation

The user needs TO interact with TO to generate the key. The user registration algorithm is shown in Algorithm 1. The specific key generation method is as follows.

- User U_i has identity ID_i, it chooses $d_i \in \mathbb{Z}_q^*$, computes $FID_{i1} = d_iP$ as his forged identity and sends (ID_i, FID_{i1}) to TO.
- TO first checks if ID_i has been registered, if ID_i has been registered, it will discard (ID_i, FID_{i1}). Otherwise, TO computes $FID_{i2} = ID_i \oplus H(kFID_{i1})$, generating $FID_i = (FID_{i1}, FID_{i2})$ to send to KGC.
- KGC randomly selects $w_i \in \mathbb{Z}_q^*$, and calculates $W_i = w_iP$, $h_{1i} = H_1(FID_i, W_i, P_0)$, $PSK_i = w_i + sh_{1i}$. After completing the calculation, KGC sends (PSK_i, W_i, FID_i) to user U_i.
- User U_i randomly selects $x_i \in \mathbb{Z}_q^*$ and calculates $h_{2i} = H_2(FID_i, X_i)$, $X_i = x_iP$, $Y_i = W_i + h_{2i}X_i$. U_i sets his public key $pk_i = Y_i$, private key $sk_i = x_i$, and stores W_i on the blockchain.

Algorithm 1: User Registration

Input: User's ID collection .
System master key s, TO private key k.
Output: pk_i, sk_i, PSK_i.

for $i \leftarrow 1$ *to* n **do**
  randomly choose $d_i \in \mathbb{Z}_q^*$; $FID_{i1} \leftarrow d_i P$;
  $FID_{i2} \leftarrow ID_i \oplus H(kFID_{i1})$;
  $FID_i \leftarrow (FID_{i1}, FID_{i2})$;
end
for $i \leftarrow 1$ *to* n **do**
  if ID_i *is not registered* **then**
    randomly choose $w_i \in \mathbb{Z}_q^*$; $W_i = w_i P$;
    $h_{1i} = H_1(FID_i, W_i, P_0)$; $PSK_i = w_i + sh_{1i}$;
  end
  else
    continue;
  end
end
for $i \leftarrow 1$ *to* n **do**
  $X_i \leftarrow x_i P$; $h_{2i} \leftarrow H_2(FID_i, X_i)$;
  $Y_i \leftarrow W_i + h_{2i} X_i$;
end
$pk_i \leftarrow Y_i$; $sk_i \leftarrow x_i$;

4.3 Authentication Request and Generate Signature

We take the authentication of domain A requesting domain B to illustrate our program, other cases across the identity authentication of the same reason, the specific process is as follows.

- A device in domain A accessing an edge server in its own domain will make a request for authentication to the devices in domain B $req(A, B) = (FID_{B1}, FID_{B2}, ..., FID_{Bn})$, and $req(A, B)$ is recorded on the blockchain.
- The edge server in domain B monitors the blockchain in real time, and when an identity authentication request is retrieved from domain B, the edge server broadcasts $req(A, B) = (FID_{B1}, FID_{B2}, ..., FID_{Bn})$ to the local territory.
- The device U_{Bi} randomly selected $v_i \in \mathbb{Z}_q^*$ and calculates $V_i = v_i P$, $h_{2i} = H_2(FID_{Bi}, X_i)$, $h_{3i} = H_3(FID_{Bi}, pk_{Bi}, V_i)$, $D_i = v_i + h_{3i}(PSK_i + h_{2i}x_i)$.
- U_{Bi} generate signature $\sigma = (V_i, D_i)$ and send σ to the edge server.

4.4 Aggregate Signature

- After receiving the signature of the device being requested for authentication, the edge server in domain B first unpacks the signature into $\sigma = (V_i, D_i)$, and then computes $h_1^{'} = H_1(FID_{Bi}, W_i, P_0), h_3^{'} = H_3(FID_{Bi}, pk_i, V_i)$.
- The edge server verifies equation $D_i P = V_i + h_{3i}^{'}(Y_i + h_{1i}^{'} P_0)$.

- If the verification is successful, the signature is added to the aggregated list L_{agg}, otherwise, a verification failure message is broadcast requesting the device that failed to verify to resubmit the signature.
- Edge server aggregates list of L_{agg}, calculates $D = \sum_{i=1}^{n} D_i$. The aggregated signature is $\sigma_{agg} = (\{V_1, ..., V_n\}, D)$, and the edge server stores the aggregated signature to the the blockchain.

Algorithm 2: Signature Generation

Input: Public key pk_i.
Private key sk_i.
Partial private key PSK_i.
Output: $\sigma = (V_i, D_i)$.

randomly choose $v_i \in \mathbb{Z}_q^*$;
$V_i = v_i P$;
$h_{2i} = H_2(FID_i, X_i)$;
$h_{3i} = H_3(FID_i, pk_i, V_i)$;
$D_i = v_i + h_{3i}(PSK_i + h_{2i}x_i)$;
$\sigma \leftarrow (V_i, D_i)$;

4.5 Aggregate Signature Verification

After obtaining σ_{agg} from the blockchain, the requester verifies that the following equation holds.

$$DP = \sum_{i=1}^{n} V_i + \sum_{i=1}^{n} h_{3i}(Y_i + h_{1i}P_0)$$

If the equation is valid then the identity authentication is successful, if the equation is not valid then the authentication needs to be requested again.

Algorithm 3: Identity Authentication

Input: Aggregated signature
$\sigma_{agg} = (\{V_1, ..., V_n\}, D)$.
Output: *True* or *False*.

for $i \leftarrow 1$ *to* n **do**
  $h_{1i}^{'} \leftarrow H_1(FID_i, W_i, P_0)$;
  $h_{3i}^{'} \leftarrow H_3(FID_i, pk_i, V_i)$;
  $D \leftarrow D + D_i$;
end
if $DP = \sum_{i=1}^{n} V_i + \sum_{i=1}^{n} h_{3i}(Y_i + h_{1i}P_0)$ **then**
  Return *True;*
end
else
  Return *False;*
end

5 Security Analysis

In this section, we analyze the individual signature correctness, aggregate signature correctness, and unforgeability of BCPP-IAS.

5.1 Proof of Exactness

The individual signature is corrected. In fact,

$$\begin{aligned} D_iP &= (v_i + h_{3i}(PSK_i + h_{2i}x_i))P = v_iP + h_{3i}P(PSK_i + h_{2i}x_i) \\ &= V_i + h_{3i}(W_i + h_{1i}P_0 + h_{2i}X_i) = V_i + h_{3i}(Y_i + h_{1i}P_0) \end{aligned} \tag{1}$$

In addition, the aggregation of signature is corrected. Since,

$$\begin{aligned} DP &= \sum_{i=1}^{n} D_iP = \sum_{i=1}^{n}(v_i + h_{3i}(PSK_i + h_{2i}x_i))P \\ &= \sum_{i=1}^{n}(V_i + h_{3i}(Y_i + h_{1i}P_0)) = \sum_{i=1}^{n} V_i + \sum_{i=1}^{n} h_{3i}(Y_i + h_{1i}P_0) \end{aligned} \tag{2}$$

5.2 Theoretical security analysis

Theorem 1 (Unforgeability of Adv_I): If the ECDLP[1] problem is hard, BCPP-IAS is existentially unforgeable for adaptive choice of message attacks by Adv_I.

Proof. The adversary Adv_I can make various queries with the challenger $\mathcal{C}$ to help himself get more information to forge signatures, and we assume that the challenger $\mathcal{C}$ is given $P_0 = sP$ as a random instance of ECDLP. FID^* as the target identity of the forged signature of Adv_I.

System initialization: Challenger $\mathcal{C}$ runs the SetUp algorithm to input system security parameters λ generate system parameters $sysparas = \{\mathbb{G}_1, \mathbb{G}_2, e, P, P_0, T_0, \mathrm{H}_1, \mathrm{H}_2, \mathrm{H}_3\}$ keeps the master key s secretly and sends the system parameters $sysparas$ to the adversary Adv_I.

H_1 *queries:* The challenger maintains an initially empty table T_1, the table contains the following tuple $(FID_i, W_i, P_0, h_{1i})$, and when the adversary Adv_I submits (FID_i, W_i, P_0) for the H_1 query , if the query has already been recorded in T_1 then $\mathcal{C}$ returns the result directly, otherwise, the challenger $\mathcal{C}$ randomly chooses $h_{1i} \in \mathbb{Z}_q^*$ and sets $h_{1i} = H_1(FID_i, W_i, P_0)$ to add $(FID_i, W_i, P_0, h_{1i})$ to the table T_1, and then $\mathcal{C}$ returns the results of h_{1i} back to Adv_I.

H_2 *queries:* The challenger maintains an initially empty table T_2, the table contains the following tuple (FID_i, X_i, h_{2i}), and when the adversary Adv_I submits (FID_i, X_i) for the query H_2, if the query has already been recorded in T_2

[1] Elliptic Curve Discrete Logarithm Problem (ECDLP):Given $Q, P \in \mathbb{G}_1$, the ECDLP is to find $x \in \mathbb{Z}_q^*$, makes the equation $Q = xP$ computationally infeasible.

then $\mathcal{C}$ returns the result directly, otherwise, the challenger $\mathcal{C}$ randomly chooses $h_{2i} \in \mathbb{Z}_q^*$ and sets $h_{2i} = H_2(FID_i, X_i)$ to add (FID_i, X_i, h_{2i}) to the table T_2, and then $\mathcal{C}$ returns the results of h_{2i} back to Adv_I.

H_3 queries: The challenger maintains an initially empty table T_3, the table contains the following tuple (FID_i, pk_i, h_{3i}), and when the adversary Adv_I submits (FID_i, pk_i) for the query H_3, if the query has already been recorded in T_3 then $\mathcal{C}$ returns the result directly, otherwise, the challenger $\mathcal{C}$ randomly chooses $h_{3i} \in \mathbb{Z}_q^*$ and sets $h_{3i} = H_3(FID_i, X_i)$ to add (FID_i, X_i, h_{3i}) to the table T_3, and then $\mathcal{C}$ returns the results of h_{3i} back to Adv_I.

User public key queries: The challenger $\mathcal{C}$ maintains the table T_{pk} which contains tuples of the form $(FID_i, Y_i, W_i, a_i, b_i)$, when the adversary Adv_I performs a public key query, if the queried tuple has been recorded in the table T_{pk}, then $\mathcal{C}$ returns (Y_i, W_i) to Adv_I and records in T_{pk}. If $FID_i = FID^*$, challenger $\mathcal{C}$ chooses $a_i, b_i, c_i \in \mathbb{Z}_q^*$ such that $W_i = a_iP$, $X_i = b_iP$, $c_i = H_2(FID_i, X_i)$, $Y_i = a_iP + c_i(b_iP)$, add (Y_i, W_i, a_i, b_i) to T_{pk} and send (Y_i, W_i) to the adversary Adv_I. If $FID_i \neq FID^*$, challenger $\mathcal{C}$ sets $W_i = a_iP$, $X_i = b_iP$, $c_i = H_2(FID_i, X_i)$, and $Y_i = W_i + c_iX_i$ and outputs (Y_i, W_i). Add $(FID_i, Y_i, W_i, a_i, b_i)$ to table T_{pk} and send (Y_i, W_i) to Adv_I.

User partial secret key queries: $\mathcal{C}$ maintains an initially empty table T_k, T_k contains tuples of the form (FID_i, W_i, PSK_i). When the adversary submits FID_i for a partial secret key query, the Game I terminates if $FID_i = FID_i^*$. Otherwise, $\mathcal{C}$ randomly chooses $a_i, b_i \in \mathbb{Z}_q^*$, sets $H_1(FID_i, W_i, P_0) = b_i$, $W_i = a_iP$, $PSK_i = a_i + sb_i$. $\mathcal{C}$ adds (FID_i, W_i, P_0, b_i) to the table T_1 and (FID_i, W_i, d_i) to the table T_k, returning PSK_i to Adv_I.

User secret key queries: Adv_I submits FID_i for the query, and if $FID_i = FID_i^*$, $\mathcal{C}$ terminates the game. If $FID_i \neq FID_i^*$ and the query tuple already exists in T_{pk}, $\mathcal{C}$ returns directly to the adversary x_i. If the query does not exist in T_{pk}, $\mathcal{C}$ passes the T_{pk} generates x_i, Y_i and adds the values to T_{pk}, then returns to the adversary x_i.

Replace public key: Adv_I chooses a new public key $(Y_i^{'}, W_i^{'})$ on FID_i and sends it to $\mathcal{C}$, who replaces the original public key of IFD_i using $(Y_i^{'}, W_i^{'})$ with a tuple of replaced tuples of the form $(FID_i, Y_i^{'}, W_i^{'}, \perp, \perp)$.

Signature queries: Adv_I submits (FID_i, m) to $\mathcal{C}$ for a signature query. $\mathcal{C}$ follow Algorithm 2 to generate a signature for Adv_I.

Forgery Phase: Adv_I eventually forges the signature $\sigma_{agg}^* = (V_i^*, D_i^*)_{i=1\ to\ n}$, which after aggregation is $\sigma_{agg}^* = (V_1^*, V_2^*, ..., V_n^*, D^*)$. If $FID_i \neq FID_i^*$, $\mathcal{C}$ stops the game and the Game I fails. Otherwise, by forking lemma [24], Adv_I is able to forge the remaining $2n$ valid signatures $\sigma_{agg}^{*(j)} = (V_i^{*(j)}, D^{*(j)})_{i=1\ to\ n}, j = 1, 2, ..., 2n+1$. All $2n+1$ signatures satisfy the verification equation $D^{*(j)}P = \sum_{i=1}^{n} V_i^* + \sum_{i=1}^{n} {h_3}_i^{*(j)}(Y_i^* + {h_1}_i^* P_0)$. We set $V_i^* = v_i, Y_i^* = y_i^*$, and simplify to get the following equation.

$$D^{*(j)} = \sum_{i=1}^{n} v_i^* + \sum_{i=1}^{n} {h_3}_i^{*(j)}(y_i^* + {h_1}_i^* s)$$

$\mathcal{C}$ can solve v_i, y_i and s from this system of $2n+1$ linearly unrelated equations, outputting s as a solution to the ECDLP, but the ECDLP is difficult and contradicts the assumptions, so BCPP-IAS is unforgeable for Adv_I.

Theorem 2 (Unforgeability of Adv_{II}): If ECDLP is hard, BCPP-IAS is unforgeable under an adaptive choice message attack by Adv_{II}.

Proof. The adversary Adv_{II} can make various queries with the challenger $\mathcal{C}$ to help himself get more information to forge signatures, and we assume that the challenger $\mathcal{C}$ is given $Q = \alpha P$ as a random instance of ECDLP. FID^* as the target identity of the forged signature of Adv_{II}. H_1 *queries*, H_2 *queries*, H_3 *queries* are similar to Game I.

User public key queries: The challenger $\mathcal{C}$ maintains the table T_{pk} which contains tuples of the form $(FID_i, Y_i, W_i, a_i, b_i)$, when the adversary Adv_{II} performs a public key query, if the queried tuple has been recorded in the table T_{pk}, then $\mathcal{C}$ returns (Y_i, W_i) to Adv_I and records in T_{pk}. If $FID_i = FID^*$, challenger $\mathcal{C}$ chooses $a_i, b_i, c_i \in \mathbb{Z}_q^*$, sets $W_i = a_i P$, $X_i = Q$, $c_i = H_2(FID_i, X_i)$, $Y_i = W_i + h_{2i} X_i = a_i P + c_i(b_i P)$, add (Y_i, W_i, a_i, b_i) to T_{pk} and send (Y_i, W_i) to the adversary Adv_{II}. If $FID_i \neq FID^*$, $\mathcal{C}$ sets $W_i = a_i P$, $X_i = b_i P$, $c_i = H_2(FID_i, X_i)$, and $Y_i = W_i + c_i X_i$ and outputs (Y_i, W_i). Add $(FID_i, Y_i, W_i, a_i, b_i)$ to table T_{pk} and send (Y_i, W_i) to Adv_{II}.

User secret key queries: Adv_{II} submits FID_i for the query, and if $FID_i = FID_i^*$, $\mathcal{C}$ terminates the game. If $FID_i \neq FID_i^*$ and the query tuple already exists in T_{pk}, $\mathcal{C}$ returns to the adversary x_i. If the query does not exist in T_{pk}, $\mathcal{C}$ passes the T_{pk} generates x_i, Y_i and adds the values to T_{pk}, then returns to the adversary x_i.

Signature queries: Adv_{II} submits (FID_i, m) to $\mathcal{C}$ for a signature query. $\mathcal{C}$ follow Algorithm 2 to generate a signature for Adv_I.

Forgery Phase: Adv_{II} eventually forges the signature $\sigma_{agg}^* = (V_i^*, D_i^*)_{i=1\ to\ n}$, which after aggregation is $\sigma_{agg}^* = (V_1^*, V_2^*, ..., V_n^*, D^*)$. If $FID_i \neq FID_i^*$, $\mathcal{C}$ stops the game and the Game I fails. Otherwise, by forking lemma [24], Adv_{II} is able to forge the remaining n valid signatures $\sigma_{agg}^{*(j)} = (V_i^{*(j)}, D^{*(j)})_{i=1\ to\ n}, j = 1, 2, ..., n+1$. All $n+1$ signatures satisfy the verification equation. Then, we get $D^{*(j)} P = \sum_{i=1}^{n} V_i^* + \sum_{i=1}^{n} h_{3_i}^{*(j)} (PSK_i^* P + h_{2i}^* X_i)$.

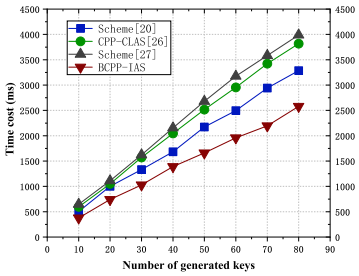
We set $V_i^* = v_i, X_i = \alpha P$ for $i = 1$ to n , and simplify to get the following equation.

$$D^{*(j)} = \sum_{i=1}^{n} v_i^* + \sum_{i=1}^{n} h_{3_i}^{*(j)} (PSK_i^* + h_{2_i}^* \alpha)$$

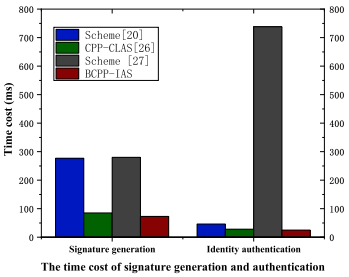
$\mathcal{C}$ can solve v_i and s from this system of $n+1$ linearly unrelated equations, outputting s as a solution to the ECDLP, but the ECDLP is hard and contradicts the assumptions, so BCPP-IAS is unforgeable for Adv_{II}.

6 Performance Evaluation

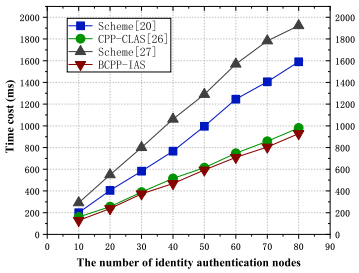
This section assesses the performance of BCPP-IAS in relation to computation cost, communication cost, and gas spending on the blockchain. BCPP-IAS was simulated using Java Pairing Based Cryptography (JPBC) [25] on a machine equipped with an AMD R7-6800H CPU (8 core 4.7 Ghz) and 32GB RAM. In addition, we simulated [20] , [26] , and [27] under the same experimental permutations.



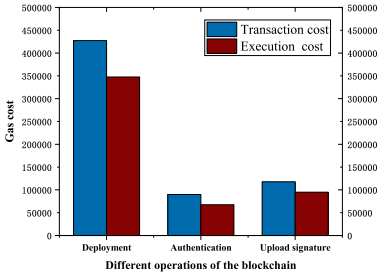
(a) Key generation time cost



(b) Signature and identity authentication time cost



(c) The time cost of multiple identity authentications



(d) The cost of gas on Ethereum for different operations of BCPP-IAS

Fig. 2. Results of BCPP-IAS experiment

6.1 Computational Cost

The results of the key generation time cost of BCPP-IAS compared to [20] , [26], and [27] are shown in Fig. 2(a), where the horizontal axis represents the number of keys generated and the vertical axis represents the time spent. It can be seen that BCPP-IAS spends less time in generating the same number

Table 2. Comparison of communication cost

Schemes	Length of signature	Length of public key	Length of partial private key	Length of private key
Scheme [20]	$\|\mathbb{G}_1\|$	$\|\mathbb{G}_1\|$	0	$\|\mathbb{Z}_q^*\|$
CPP-CLAS [26]	$\|\mathbb{G}_1\| + \|\mathbb{Z}_q^*\|$	$2\|\mathbb{G}_1\|$	$\|\mathbb{Z}_q^*\|$	$\|\mathbb{Z}_q^*\|$
Scheme [27]	$2\|\mathbb{G}_1\|$	$\|\mathbb{G}_1\|$	$\|\mathbb{G}_1\|$	$\|\mathbb{Z}_q^*\|$
BCPP-IAS	$\|\mathbb{G}_1\| + \|\mathbb{Z}_q^*\|$	$\|\mathbb{G}_1\|$	$\|\mathbb{Z}_q^*\|$	$\|\mathbb{Z}_q^*\|$

of keys compared to [20], [26], and [27], and the computational cost in the key generation phase is lower. Fig. 2(b) illustrates the overhead of signature time and verification time for [20], [26], [27], and BCPP-IAS. From the Fig. 2(b), we can see that the signature time cost and verification time cost of our [27] and [20] is higher than that of BCPP-IAS, and the signature time cost and verification time cost of [26] are slightly higher than that of BCPP-IAS. This is because BCPP-IAS does not use computationally complex bilinear pairing operations. Fig. 2(c) compares the verification time cost after aggregating signatures. The horizontal axis represents the signatures being aggregated, while the vertical axis represents the time spent on verifying the aggregated signatures. Obviously, [20] and [27] have a much higher aggregated signature verification time cost than BCPP-IAS, while CPP-CLAS has a nearly identical aggregated signature verification time to BCPP-IAS. In summary, BCPP-IAS has good performance in the key generation phase and good efficiency in the signature time cost and verification time cost.

6.2 Communication Cost

We use the length of the public key and the length of the ring signature to measure the communication cost [28]. Table 2 summarizes and shows the signature length comparison, public key length comparison, partial private key length comparison and private key length comparison of BCPP-IAS with [20] , [26], [27], where $|\mathbb{G}_1| = 128$ Bytes (B), $\left|\mathbb{Z}_q^*\right| = 20$ B. BCPP-IAS has the signature length of $|\mathbb{G}_1| + \left|\mathbb{Z}_q^*\right|$ which is the same as that of CPP-CLAS and 108 B shorter than that of scheme [27]. In addition, the public key length of BCPP-IAS is $|\mathbb{G}_1|$, which is 128 B shorter than the public key length of CPP-CLAS and the same as that of scheme [27], and the partial private key length is $\left|\mathbb{Z}_q^*\right|$, which is the same as that of CPP-CLAS with partial private key The length is the same as that of the partial private key of CPP-CLAS, which is 108 B shorter than that of the partial private key of scheme [27]. The length of the private key of [26],[27] is the same as that of the private key of BCPP-IAS, which is $\left|\mathbb{Z}_q^*\right|$. Although the signature length and public key of the scheme [20] are shorter than ours, their scheme does not consider the function of privacy protection. To summarize, BCPP-IAS has a lower signature length compared to having existing schemes, and the length of public key, part of private key, and private is also shorter, so BCPP-IAS has a lower communication cost.

6.3 Gas Spending on Blockchain

We use Ethereum [29], an open-source public blockchain platform with smart contract capabilities, as the experimental environment for BCPP-IAS. We first

start a Ganache chain, then use MetaMask to access Ganache, and then develop and compile and deploy smart contracts on Remix, which accesses MetaMask through Injected Web3, and then interacts with the underlying blockchain through MetaMask. We use contract execution cost and transaction cost to measure the performance of smart contracts[30]. The gas consumption for different operations on the blockchain is detailed in Fig. 2(d).

The smart contract deployment transaction cost of BCPP-IAS is 427,378 gas, and the execution cost of the contract deployment is 347,586 gas, the deployment is a one-time job, and no gas is consumed subsequently. the IoT device uploads an authentication request with a transaction cost of 89,918 gas, and an execution cost of 67,706 gas, and the edge server uploads an aggregated signature with a transaction cost of 118,054 gas, and an execution cost of 94,914 gas. The transaction cost for an IoT device to upload an authentication request is 89,918 gas, and the execution cost is 67,706 gas. The transaction cost for an edge server to upload an aggregated signature is 118,054 gas, and the execution cost is 94,914 gas. After aggregation, the signature is uploaded to the blockchain, reducing storage pressure on the blockchain. The immutable nature of the blockchain ensures that the data recorded is trustworthy. Therefore, subsequent requesters can directly utilize the aggregated signature for identity authentication.

7 Conclusion

In this article, we proposed BCPP-IAS to solve low efficiency of cross-domain identity authentication for IoT and the single point of failure. BCPP-IAS utilizing blockchain to record authentication requests, bringing trust to distributed systems. In addition, we designed a new certificateless aggregate signature algorithm to implement identity authentication between IoT devices, which is more efficient because it does not use bilinear pairing operation. Multiple certificateless signatures can be aggregated into one signature to achieve batch authentication, which greatly reduces the computing and storage costs of edge servers. Finally, we prove the efficiency and practicability of BCPP-IAS through theoretical analysis and experiments.

References

1. Qingru Ma, Haowen Tan, and Tianqi Zhou. Mutual authentication scheme for smart devices in iot-enabled smart home systems. *Computer Standards & Interfaces*, 86:103743, 2023.
2. Fengqun Wang, Jie Cui, Qingyang Zhang, Debiao He, Chengjie Gu, and Hong Zhong. Blockchain-based lightweight message authentication for edge-assisted cross-domain industrial internet of things. *IEEE Transactions on Dependable and Secure Computing*, 2023.
3. Suhui Liu, Jiguo Yu, Liquan Chen, and Baobao Chai. Blockchain-assisted comprehensive key management in cp-abe for cloud-stored data. *IEEE Transactions on Network and Service Management*, 2022.

4. Jiguo Yu, Suhui Liu, Minghui Xu, Hechuan Guo, Fangtian Zhong, and Wei Cheng. An efficient revocable and searchable ma-abe scheme with blockchain assistance for c-iot. *IEEE Internet of Things Journal*, 10(3):2754–2766, 2022.
5. Suhui Liu, Jiguo Yu, Yinhao Xiao, Zhiguo Wan, Shengling Wang, and Biwei Yan. Bc-sabe: Blockchain-aided searchable attribute-based encryption for cloud-iot. *IEEE Internet of Things Journal*, 7(9):7851–7867, 2020.
6. Jiguo Yu, Biwei Yan, Huayi Qi, Shengling Wang, and Wei Cheng. An efficient and secure data sharing scheme for edge-enabled iot. *IEEE Transactions on Computers*, 2023.
7. Osama A Khashan and Nour M Khafajah. Efficient hybrid centralized and blockchain-based authentication architecture for heterogeneous iot systems. *Journal of King Saud University-Computer and Information Sciences*, 35(2):726–739, 2023.
8. Guangsong Li, Yang Wang, Bin Zhang, and Siqi Lu. Smart contract-based cross-domain authentication and key agreement system for heterogeneous wireless networks. *Mobile Information Systems*, 2020:1–16, 2020.
9. Yimin Guo, Zhenfeng Zhang, Yajun Guo, and Ping Xiong. Bsra: Blockchain-based secure remote authentication scheme for the fog-enabled internet of things. *IEEE Internet of Things Journal*, 2023.
10. Minghui Xu, Shuo Liu, Dongxiao Yu, Xiuzhen Cheng, Shaoyong Guo, and Jiguo Yu. Cloudchain: a cloud blockchain using shared memory consensus and rdma. *ieee transactions on computers*, 71(12):3242–3253, 2022.
11. Guanjie Cheng, Yan Chen, Shuiguang Deng, Honghao Gao, and Jianwei Yin. A blockchain-based mutual authentication scheme for collaborative edge computing. *IEEE Transactions on Computational Social Systems*, 9(1):146–158, 2021.
12. Mohammad Dabbagh, Kim-Kwang Raymond Choo, Amin Beheshti, Mohammad Tahir, and Nader Sohrabi Safa. A survey of empirical performance evaluation of permissioned blockchain platforms: Challenges and opportunities. *computers & security*, 100:102078, 2021.
13. Yujian Zhang, Yuhao Luo, Xing Chen, Fei Tong, Yuwei Xu, Jun Tao, and Guang Cheng. A lightweight authentication scheme based on consortium blockchain for cross-domain iot. *Security and Communication Networks*, 2022:1–15, 2022.
14. Jing Liu, Yixin Liu, Yingxu Lai, Rongchen Li, Siyu Wu, and Sami Mian. Cross-heterogeneous domain authentication scheme based on blockchain. *Journal of Artificial Intelligence and Technology*, 1(2):92–100, 2021.
15. Wentong Wang, Ning Hu, and Xin Liu. Blockcam: a blockchain-based cross-domain authentication model. In *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, pages 896–901. IEEE, 2018.
16. Xingcan Zhou, Fuyou Miao, and Yan Xiong. A certificate authority domain-based cross-domain authentication scheme for virtual enterprise using identity based encryption. In *2021 7th International Conference on Big Data Computing and Communications (BigCom)*, pages 144–149. IEEE, 2021.
17. Yuhan Yang, Jing Wu, Chengnian Long, Qingquan Zou, and Ji Gao. A blockchain-based cross-domain authentication for conditional privacy preserving in vehicular ad-hoc network. In *2021 The 3rd International Conference on Blockchain Technology*, pages 183–188, 2021.
18. Fatemeh Stodt and Christoph Reich. Bridge of trust: Cross domain authentication for industrial internet of things (iiot) blockchain over transport layer security (tls). *Electronics*, 12(11):2401, 2023.

19. Xu Wu, Yang Liu, Jie Tian, and Yuanpeng Li. Privacy-preserving trust management method based on blockchain for cross-domain industrial iot. *Knowledge-Based Systems*, 283:111166, 2024.
20. Chaosheng Feng, Bin Liu, Zhen Guo, Keping Yu, Zhiguang Qin, and Kim-Kwang Raymond Choo. Blockchain-based cross-domain authentication for intelligent 5g-enabled internet of drones. *IEEE Internet of Things Journal*, 9(8):6224–6238, 2021.
21. Wenze Mao, Peng Jiang, and Liehuang Zhu. Btaa: Blockchain and tee assisted authentication for iot systems. *IEEE Internet of Things Journal*, 2023.
22. Mouna Nakkar, Riham AlTawy, and Amr Youssef. Gase: A lightweight group authentication scheme with key agreement for edge computing applications. *IEEE Internet of Things Journal*, 10(1):840–854, 2022.
23. Bei Gong, Guiping Zheng, Muhammad Waqas, Shanshan Tu, and Sheng Chen. Lcdma: Lightweight cross-domain mutual identity authentication scheme for internet of things. *IEEE Internet of Things Journal*, 2023.
24. Zirui Qiao, Yanwei Zhou, Bo Yang, Mingwu Zhang, Tao Wang, and Zhe Xia. Secure and efficient certificate-based proxy signature schemes for industrial internet of things. *IEEE Systems Journal*, 16(3):4719–4730, 2022.
25. Angelo De Caro and Vincenzo Iovino. jpbc: Java pairing based cryptography. In *2011 IEEE symposium on computers and communications (ISCC)*, pages 850–855. IEEE, 2011.
26. Yulei Chen and Jianhua Chen. Cpp-clas: Efficient and conditional privacy-preserving certificateless aggregate signature scheme for vanets. *IEEE Internet of Things Journal*, 9(12):10354–10365, 2022.
27. Yangfan Liang and Yining Liu. Analysis and improvement of an efficient certificateless aggregate signature with conditional privacy preservation in vanets. *IEEE Systems Journal*, 17(1):664–672, 2023.
28. Mohamed Amine Ferrag and Lei Shu. The performance evaluation of blockchain-based security and privacy systems for the internet of things: A tutorial. *IEEE Internet of Things Journal*, 8(24):17236–17260, 2021.
29. Huashan Chen, Marcus Pendleton, Laurent Njilla, and Shouhuai Xu. A survey on ethereum systems security: Vulnerabilities, attacks, and defenses. *ACM Computing Surveys (CSUR)*, 53(3):1–43, 2020.
30. Tong Wu, Weijie Wang, Chuan Zhang, Weiting Zhang, Liehuang Zhu, Keke Gai, and Haotian Wang. Blockchain-based anonymous data sharing with accountability for internet of things. *IEEE Internet of Things Journal*, 10(6):5461–5475, 2023.