Blockchain-Based Lightweight Message Authentication for Edge-Assisted Cross-Domain Industrial Internet of Things

Fengqun Wang, Jie Cui, Qingyang Zhang, Debiao He, Chengjie Gu and Hong Zhong

***Abstract*—In edge-assisted cross-domain Industrial Internet of Things (IIoT), blockchain-based authentication is an effective way to build cross-domain trust and secure cross-domain data. However, existing authentication schemes still have serious challenges in terms of efficiency and security. In this paper, we propose a blockchain-based lightweight message authentication scheme. First, to address efficiency challenges, we build a blockchain-enabled edge-assisted lightweight authentication framework. This framework uses edge servers to assist smart devices in achieving cross-domain authentication and effectively reduce redundant interactions between entities. Second, to resolve the security challenges, we design a lightweight message authentication algorithm for cross-domain IIoT. The algorithm guarantees message security with low computational overhead and is suitable for multi-receiver cross-domain IIoT. The security proof and analysis demonstrate that the proposed scheme is secure under the random oracle model and can resist various attacks. The performance evaluation shows that our proposed scheme is superior in terms of computation and communication overhead when compared with other related schemes.**

***Index Terms*—Industrial Internet of Things (IIoT), consortium blockchain, cross-domain authentication, elliptic curve cryptography (ECC).**

I. Introduction

IN recent years, the introduction of edge computing [1], [2] into the Industrial Internet of Things (IIoT) [3], [4], [5] has shortened decision latency, simplified network topology, and optimized device management in industrial manufacturing [6], [7]. However, as manufacturing becomes more complex, producing a product often requires smart devices from multiple administrative domains (e.g., different smart factories) to collaborate in real time [8], [9].

Fig. 1 shows a typical data exchange scenario in the edge-assisted cross-domain IIoT. In this scenario, multiple smart devices can support the same IIoT service (e.g., the same production task) [10], [11], so this scenario is generally a multi-receiver scenario, i.e., a message has multiple receivers. These devices generate real-time data according to the service type, exchange data with the assistance of edge servers, and process the received data in time. However, the data faces many security issues throughout the transmission process [12], [13]. For example, malicious network attackers can intercept and tamper with real-time data. Once this tampered data is used, it could lead to disruptions in industrial production and economic losses. On the other hand, trust between multiple domains is difficult to build. That is, multiple administrative domains do not trust each other and are reluctant to share sensitive data.

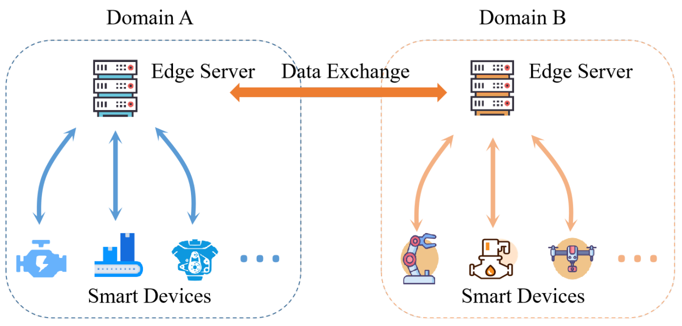


Fig. 1. Data exchange between two domains based on edge computing.

To secure cross-domain data and build trust between multiple administrative domains, many researchers have introduced consortium blockchain technology [14], [15], [16] and proposed cross-domain authentication schemes [17], [18]. However, existing blockchain-based cross-domain authentication schemes face several challenges.

- A blockchain-enabled authentication framework should be lightweight. Most existing blockchain-enabled frameworks have complex organizational structures. For example, the authentication framework is complex in the scheme proposed by Shen *et al.* [19]. Verifying the legitimacy of a message requires multiple interactions between multiple entities (smart device, authentication agent server, and blockchain). In the multiple-receiver IIoT, these interactions increase as the number of receivers increases. This generates huge communication overheads and non-negligible communication latency, which may eventually lead to conflicts between collaborating devices and disrupt the production process [20].
- An authentication algorithm should be lightweight. IIoT requires a high level of real-time data. For example, in some industrial control applications, feedback is required within $10 - 100$ milliseconds (ms) [21]. Most existing authentication schemes involve many time-consuming

F. Wang, J. Cui, Q. Zhang and H. Zhong are with the School of Computer Science and Technology, Anhui University, Hefei 230039, China, the Anhui Engineering Laboratory of IoT Security Technologies, Anhui University, Hefei 230039, China, and the Institute of Physical Science and Information Technology, Anhui University, Hefei 230039, China (e-mail: cuijie@mail.ustc.edu.cn).

C. Gu is with the Security Research Institute, New H3C Group, Hefei, 230088, China (e-mail: gu.chengjie@h3c.com).

D. He is with the School of Cyber Science and Engineering, Wuhan University, 430072, China (email: hedebiao@163.com).

cryptographic operations. In this case, smart devices with limited computing power cannot process massive amounts of real-time data in time, leading to significant delays in production decisions [13] and chaotic industrial production. In the multi-receiver IIoT, this challenge becomes more prominent because the time-consuming operation of most existing schemes increases linearly with the number of receivers.
- The security of the authentication algorithm should be more comprehensive. Most existing authentication algorithms do not consider data confidentiality and anonymity simultaneously. If data confidentiality is not guaranteed, attackers can easily obtain industry-critical data (e.g., business secrets), which can lead to significant financial losses [4], [22]. If data anonymity is not guaranteed, attackers can mark the real identity of a smart device and capture the data sent by that device for analysis, leading to a privacy leakage of the device or even the IIoT [23], [24].

To solve the above problems and balance the relationship between real-time and security, we design a blockchain-based lightweight message authentication scheme. In the proposed scheme, we use blockchain to build trust between multiple administrative domains.

The contributions of the proposed scheme are as follows:

- We construct a blockchain-enabled lightweight cross-domain authentication framework. The framework uses edge servers to assist smart devices in cross-domain communication. Compared to existing cross-domain authentication frameworks, the proposed framework involves fewer entities and requires fewer interactions between entities to complete authentication.
- We design a blockchain-based lightweight message authentication algorithm, which is composed of elliptic curve cryptography and hash functions. In addition, we implement our proposed scheme and compare its performance with those of three other related schemes. The theoretical analysis and simulation results show that our proposed scheme has a lower computational overhead and is superior to the multi-receiver cross-domain IIoT.
- The security proof shows that the proposed scheme satisfies confidentiality and unforgeability. In addition, we conduct a security analysis and compare the security with three other schemes. The results show that our proposed scheme is highly secure, especially considering data confidentiality and anonymity.

The remainder of this paper is organized as follows. Section II presents work related to authentication in IIoT. Section III discusses the preliminary knowledge. Section IV offers the specific details of the framework design. Section V presents details of the proposed scheme. Section VI focuses on the security proof and analysis of the proposed scheme. Section VII provides an experimental evaluation of the proposed scheme. Finally, Section VIII concludes the paper.

II. Related Work

In this section, we mainly introduce and analyze the work related to security authentication in IIoT.

In recent years, with the development of IIoT, the security of IIoT has also received more and more attention from industry and academia. Considering the limited computing power of smart devices in IIoT, Esfahani *et al.* [25] proposed a lightweight authentication scheme that enables device-to-device authentication. In this scheme, there are only hash and XOR operations, so it has the advantages of low computation and communication overhead. Regarding security, it can resist common attacks such as replay attacks. In 2020, Verma *et al.* [26] proposed an efficient proxy signature scheme for securing IIoT data, dramatically improving performance. However, neither of these two schemes guarantees the anonymity of messages.

In 2018, Esposito *et al.* [27] designed a message authentication scheme using group signature. In terms of efficiency, this scheme uses short group signatures to reduce computational overhead. Still, for security, the scheme mainly focuses on the integrity and anonymity of the message and does not consider the confidentiality of messages. Subsequently, Cui *et al.* [28] used proxy re-encryption techniques to design an anonymous message authentication scheme. This scheme introduces edge computing to effectively reduce messages' transmission delay and guarantees messages' integrity, anonymity, and confidentiality. These two schemes focus on the security of a single administrative domain, and trust between all entities is based on trust in a trusted authority (TA). However, in a cross-domain IIoT environment, each domain does not trust the other. To establish trust between each domain, the most common approach is to have multiple domains trust a specific domain's TA, or to establish a common TA for multiple domains; however, these two approaches are difficult to implement. Therefore, these schemes are not suitable for cross-domain IIoT.

More and more researchers [29], [30], [31], [32] have started using blockchain to solve the above problems. In [29], Guan *et al.* used the decentralized feature of blockchain to solve the centralized private key generator key escrow problem. The scheme uses blockchain as an information source for synchronizing user revocation lists, enabling fast detection of user revocations. However, this scheme only achieves data anonymity, not data confidentiality. In [30], Wang *et al.* used blockchain to achieve efficient conditional anonymity and key management, and the scheme has advantages in terms of communication cost and security. Still, this scheme does not take into account data confidentiality. In 2022, Wang *et al.* [33] abstracted smart devices in multiple domains into an undirected graph. They designed an authentication scheme using dynamic accumulator and digital signature techniques, which has good performance in terms of efficiency, but the scheme does not specifically consider data anonymity and confidentiality. To achieve both confidentiality and anonymity of data, some researchers have started to combine existing encryption and signature algorithms. For example, Shen *et al.* [19] designed a secure and practical cross-domain authentication scheme, which combines encryption and group signature technology. The scheme achieves data anonymity and confidentiality. However, in terms of efficiency, the scheme contains many time-consuming cryptographic operations, resulting in high computational and communication overheads. In terms of security, the scheme does not achieve unlinkability of data. In addition, the scheme uses blockchain to solve the trust problem between multiple domains, but the authentication framework is complex, resulting in many interactions between entities during the authentication process. Subsequently, Yang *et al.* [34] proposed a blockchain-based lightweight authentication scheme that offers significant advantages in terms of computational overhead. The scheme does not meet the data confidentiality. In addition, the above studies do not consider multi-receiver cross-domain IIoT scenarios. They also fail to consider that smart devices in different administrative domains are geographically dispersed and cannot directly cross-domain interaction, because these devices' communication and mobility capabilities are limited.

In summary, although some authentication schemes have been proposed, the existing schemes do not fully consider the complexity of the system framework, the lightweight of the authentication algorithm, the confidentiality and anonymity of the data in the multi-receiver IIoT scenario. Therefore, it is necessary and meaningful to design a secure and efficient authentication scheme for the multi-receiver cross-domain IIoT.

III. Preliminaries

In this section, to better understand our proposed scheme, we introduce the elliptic curve cryptosystem and blockchain used in the proposed scheme.

A. Elliptic Curve Cryptosystem

Let $\mathbb{F}_p$ be a finite field, which is determined by a prime number p. Let a set of elliptic curve point E over $\mathbb{F}_p$ be defined by the equation: $y^2 = x^3 + ax + b \bmod p$, where $a, b \in \mathbb{F}_p$. Let O be an infinity point, then O and other points on E make up an additive elliptic curve group $\mathbb{G}$ with the order q and generator P. The main properties of $\mathbb{G}$ are listed below:

- Scalar point multiplication: Let $P \in \mathbb{G}$ and $m \in \mathbb{Z}_q^*$, the scalar multiplication of E is defined as $m \cdot P = P + P + \cdots + P$ (m times). It is worth noting that the "+" here indicates the point addition.
- Elliptic Curve Discrete Logarithm problem (ECDLP): $x \in \mathbb{Z}_q^*$, $Q = xP$, where $P, Q \in \mathbb{G}$ on curve E. Given $Q = xP$, it is computational hard for a probabilistic polynomial-time (PPT) adversary to calculate x.
- Elliptic Curve Diffie-Hellman Problem (ECDHP): $x, y \in \mathbb{Z}_q^*$, and $X = xP$, $Y = yP$, where $X, Y \in \mathbb{G}$ on curve E. Given $X = xP$ and $Y = yP$, it is computational hard for a PPT adversary to calculate xyP.

B. Blockchain

A blockchain is essentially a distributed, shared, and tamper-proof database ledger [35]. Based on the degree of centralization [8], blockchain can be divided into three types: public blockchain, consortium blockchain, and private blockchain. In the proposed scheme, we focus on the following three characteristics of the blockchain:

- **Tamper proofing:** Data stored on the blockchain are tamper-proof and trustworthy.
- **Decentralization:** Blockchain is not dependent on third-party institutions and can maintain databases based on consensus protocols.
- **Smart contract:** A smart contract [36] is a computer program running on the blockchain that can be executed automatically and accurately according to a contract. Once a smart contract is deployed, it will not be modified.

Taking efficiency and security into account, we choose a consortium blockchain that can only be accessed by authorized organizations.

IV. Framework Design

In this section, we first describe the blockchain-enabled authentication framework. Then, we present the outline and the security model of the proposed scheme. Finally, we introduce the security objectives.

A. Blockchain-enabled Authentication Framework

In this subsection, to introduce the blockchain-enabled authentication framework more clearly, we present the entity overview and the communication process, respectively.

1) Entity Overview: We consider the scenario of implementing service-based cross-domain authentication in edge-assisted IIoT with the enablement of blockchain technology. As shown in Fig. 2, we divide the blockchain-enabled authentication framework into three layers based on functionality: the end device layer, the management layer, and the blockchain layer.

The end device layer is responsible for data sensing and processing, so it comprises all the smart devices (SD) in each domain.

The management layer is primarily responsible for managing all data in the domain, including data storing, forwarding, and querying. Therefore, this layer contains all the trusted authority (TA), the edge server (ES), and the blockchain domain agents (BCDA) in each domain. The TA, the ES, and the BCDA in this layer collaborate to assist smart devices with secure cross-domain communication and authentication.

The blockchain layer mainly builds trust between multiple administrative domains, and it is composed of BCDAs in the management layer. Specifically, these BCDAs in different domains form a consortium blockchain to maintain a ledger. This ledger contains pseudonym information for each SD, which is tamper-proof and can be used during cross-domain authentication.

The following is a detailed description of the various entities.

- **BCDA**: Each administrative domain has a BCDA, and all BCDAs form a consortium blockchain. The BCDA from different domains can issue smart contracts through negotiation. In each domain, BCDA can collaborate with

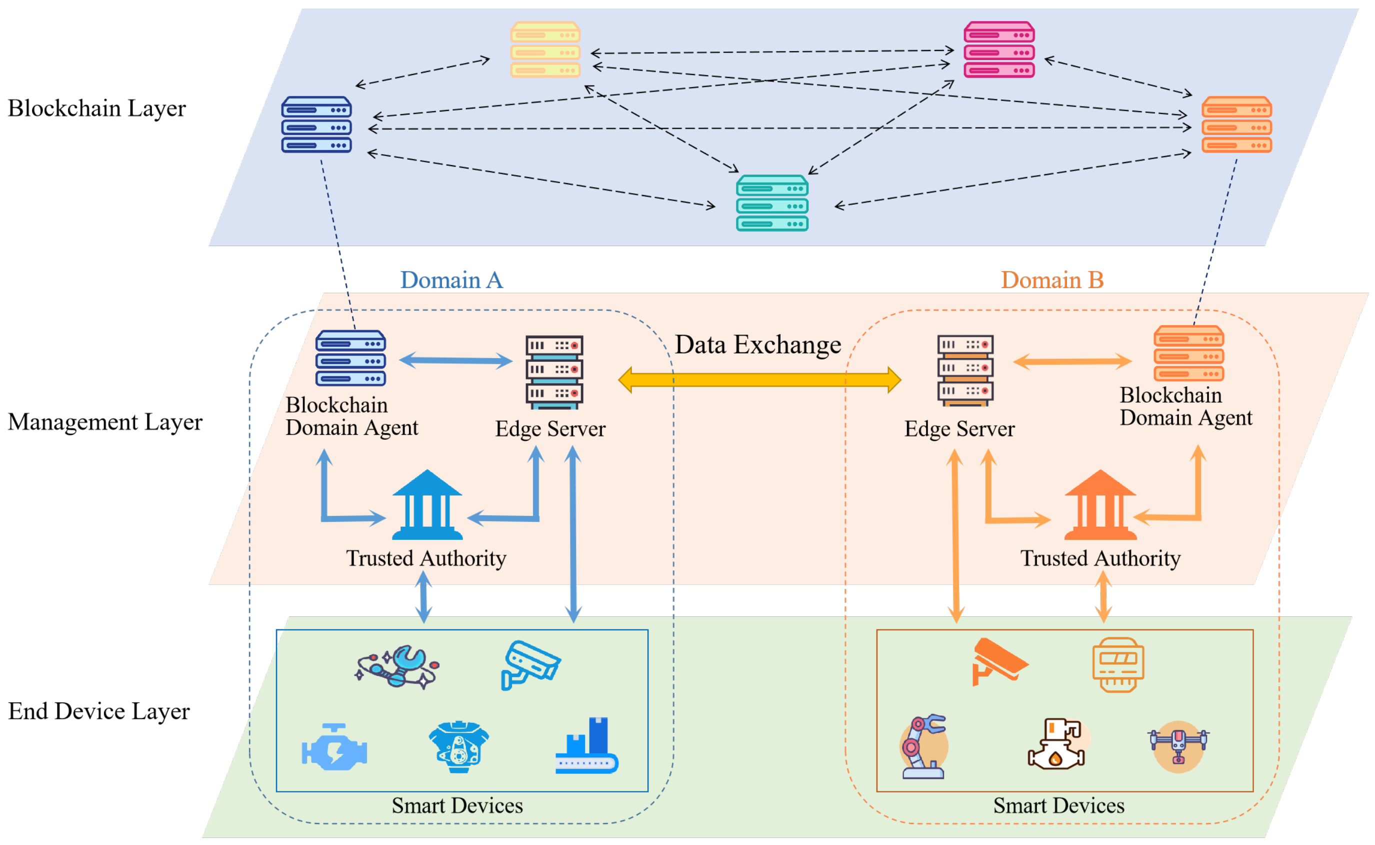


Fig. 2. Blockchain-enabled authentication framework.

TA to dynamically update the domain's latest information based on IIoT services and revoke illegal smart devices.

- **TA**: Each administrative domain has a TA. The TA is trusted to all entities in the domain. TA is mainly responsible for generating the security information and managing IIoT services. In addition, TA is the only entity that can trace messages source in its domain.
- **ES**: Each administrative domain has an ES. The ES is primarily responsible for storing and forwarding information from devices, TA, and other administrative domains. In addition, ES can assist SD in generating service-based keys and cooperate with BCDA to complete necessary data queries. Note that ES as a relay node is not required to be anonymous.
- **SD**: Each administrative domain has numerous SD. They have limited computing, mobility, and communication capabilities. Their main functions are to negotiate service-based keys, sign and encrypt messages to be sent, and decrypt and authenticate messages received. Note that SD's real identity needs to be anonymous in our proposed scheme.

2) Communication Process: In IIoT, SDs support different IIoT services. For a cross-domain service, the message sender distributes a piece of data; there may be multiple SDs in the message-receiving domain as receivers of that data. This communication is widely available currently and usually implemented in publish/subscribe systems [37], [38]. However, on the one hand, each domain is relatively independent and geographically dispersed [39], [40]. On the other hand, SDs have limited communication power and mobility. Therefore, SDs distributed in different domains cannot communicate with each other directly. In the blockchain-enabled authentication framework, we provide an ES in each administrative domain to assist SDs in cross-domain communication. That is, when SDs communicate across domains, the data needs to go through the ES.

Our proposed authentication framework is lightweight. For example, in scheme [19], when a smart device receives a message, this smart device needs to send the message to the authentication agent server. Then the authentication agent server needs to query the relevant information in the blockchain. Finally, the authentication agent server verifies the message and sends the verification result to the smart device. However, there is no authentication agent server in the framework we designed. When the message passes through the edge server, the edge server queries the relevant information in the blockchain. Then, the edge server forwards the query result and the message to the smart device. Finally, the smart device directly verifies the received messages. Therefore, our proposed scheme has fewer interactions between entities when performing message verification. In addition, in our proposed scheme, to complete a service-based cross-domain authentication, ES only needs to query the relevant data from the blockchain once, regardless of how many receivers there are. In contrast, in other schemes, the number of queries submitted to the blockchain increases with the number of receivers. Therefore, in a multi-receiver IIoT, our proposed scheme's advantage becomes even more significant.

B. Outline of the Proposed scheme

The proposed scheme consists of five algorithms, including *Setup, Service-Based Key Generation (SBKGen), Sign, Encrypt, Decrypt* and *Verify*, which are defined as follows.

1) *Setup*(1^λ): The algorithm is executed by TA. Given the random system security parameter λ, the TA outputs public system parameters $params$, system secret key msk and system public key P_{pub}.
2) *SBKGen*($serv_t, PK_{ES}^B, ID_{ES}^B, PK_{ES}^A, sk_{head}^B$): The algorithm is executed by a smart device SD_{head}^B, which generates the service-based key. Given the service $serv_t$, the public key PK_{ES}^B and identity ID_{ES}^B of the ES in the message receiving domain, the public key PK_{ES}^A of the ES in the message sending domain, and the SD_{head}^B's secret key sk_{head}^B, it outputs the corresponding $serv_t$-based secret key sk_{serv_t} and $serv_t$-based public key PK_{serv_t}. Note that the message sender uses PK_{serv_t} to encrypt the plaintext m to get the ciphertext c, and the message receiver uses sk_{serv_t} to decrypt the ciphertext c to obtain the plaintext m, so we subsequently call the sk_{serv_t} as the decryption key and the PK_{serv_t} as the encryption key.
3) *Sign*($sk_{i,j}^A, PID_{i,j}^A, PK_{ES}^B, ID_{ES}^B, PK_{ES}^A, PK_{serv_t}, m$): The algorithm is executed by the message sender SD_i^A. Given $\{PK_{ES}^B, ID_{ES}^B, PK_{ES}^A\}$, the system parameter $params$, signer's secret key $sk_{i,j}^A$, signer's pseudonym $PID_{i,j}^A$, the PK_{serv_t}, and a plaintext m, it outputs some parameters $\{h, u, U\}$ and the corresponding signature σ.
4) *Encrypt*($u, U, PK_{serv_t}, PK_{head}^B, h, \sigma, T_i, m$): The algorithm is executed by the message sender SD_i^A. Given the $\{u, U, PK_{serv_t}, PK_{head}^B, h\}$, signature σ, timestamp T_i, and plaintext m, it outputs the ciphertext c.
5) *Decrypt*(c, T_i, sk_{serv_t}): The algorithm is executed by the message receiver SD_k^B. Given the ciphertext c, the timestamp T_i, the $serv_t$-based secret key sk_{serv_t}, it output the intermediate parameter U, the corresponding plaintext m and signature σ.
6) *Verify*($m, \sigma, U, W, PK_{i,j}^A, sk_{serv_t}, P_{pub}, PID_{i,j}^A, ID_{ES}^B, PK_{ES}^B, PK_{ES}^A$): The algorithm is executed by the message receiver SD_k^B. Given parameters $\{m, \sigma, U, W, PK_{i,j}^A, sk_{serv_t}, P_{pub}, PID_{i,j}^A, ID_{ES}^B, PK_{ES}^B, PK_{ES}^A\}$, it outputs 1 if the input parameters are valid and 0 otherwise.

C. Security Model

In the security model, we divide the adversary into two categories: $\mathcal{A}_I$ and $\mathcal{A}_{II}$. Among them, $\mathcal{A}_I$ is the adversary that attacks the confidentiality of the scheme, and $\mathcal{A}_{II}$ is the adversary that attacks the unforgeability of the scheme.

1) Confidentiality

Definition 1: Our proposed scheme meets confidentiality if the probability that the adversary $\mathcal{A}_I$ could solve the ECDHP is negligible in any polynomial time. In our proposed scheme, confidentiality is defined as indistinguishability under the adaptive-chosen-ciphertext attack (IND-CCA2).

Game 1: The game is an interaction between the simulator $\mathcal{B}$ and the adversary $\mathcal{A}_I$ under IND-CCA2. The specific definition of this game is as follows:

- Setup: Given the security parameter λ, the simulator $\mathcal{B}$ outputs the system parameters $params$, system public key P_{pub}, system secret key msk, and then $\mathcal{B}$ sends $(P_{pub}, params)$ to the adversary $\mathcal{A}_I$ and keep msk secret.
- H_1-query: Adversary $\mathcal{A}_I$ sends a point $< X_i >$ to $\mathcal{B}$, where $X_i \in \mathbb{G}$. The $\mathcal{B}$ returns the corresponding hash value r_{H_1} to $\mathcal{A}_I$, where $r_{H_1} \in \mathbb{Z}_q^*$.
- H_2-query: Adversary $\mathcal{A}_I$ sends a point $< SPK >$ to $\mathcal{B}$, where $SPK \in \mathbb{G}$. The $\mathcal{B}$ returns the corresponding hash value r_{H_2} to $\mathcal{A}_I$, where $r_{H_2} \in \{0,1\}^*$.
- H_3-query: Adversary $\mathcal{A}_I$ sends $< ID_{ES}^B, PK_{ES}^B, PK_{ES}^A >$ to $\mathcal{B}$, the $\mathcal{B}$ returns the corresponding hash value r_{H_3} to $\mathcal{A}_I$, where $r_{H_3} \in \mathbb{Z}_q^*$.
- H_4-query: Adversary $\mathcal{A}_I$ sends $< K, T_i >$ to $\mathcal{B}$, where $K \in \mathbb{G}$ and T_i is a timestamp. The $\mathcal{B}$ returns the corresponding hash value r_{H_4} to $\mathcal{A}_I$, where $r_{H_4} \in \{0,1\}^*$.
- H_5-query: Adversary $\mathcal{A}_I$ sends a message $< m >$ to $\mathcal{B}$, where $m \in \{0,1\}^*$. The $\mathcal{B}$ returns the corresponding hash value r_{H_5} to $\mathcal{A}_I$, where $r_{H_5} \in \mathbb{Z}_q^*$.
- Extract encryption key query: Adversary $\mathcal{A}_I$ sends a service $< serv_t >$ to $\mathcal{B}$, and $\mathcal{B}$ returns the encryption key PK_{serv_t} corresponding to the $< serv_t >$ to $\mathcal{A}_I$.
- Extract decryption key query: Adversary $\mathcal{A}_I$ sends a service $< serv_t >$ to $\mathcal{B}$, and $\mathcal{B}$ returns the decryption key sk_{serv_t} corresponding to the $< serv_t >$ to $\mathcal{A}_I$.
- Extract verification key query: Adversary $\mathcal{A}_I$ sends a pseudonym $< PID_{i,j}^A >$ to $\mathcal{B}$, and $\mathcal{B}$ returns the verification key $PK_{i,j}^A$ corresponding to the $< PID_{i,j}^A >$ to $\mathcal{A}_I$.
- Extract signature key query: Adversary $\mathcal{A}_I$ sends a pseudonym $< PID_{i,j}^A >$ to $\mathcal{B}$, and $\mathcal{B}$ returns the signature key $sk_{i,j}^A$ corresponding to the $< PID_{i,j}^A >$ to $\mathcal{A}_I$.
- Encryption key replacement query: Adversary $\mathcal{A}_I$ sends a service $serv_t$ and a valid new encryption key $PK_{serv_t}^{'}$ to $\mathcal{B}$, $\mathcal{B}$ replaces the original encryption key PK_{serv_t} with the $PK_{serv_t}^{'}$.
- Encryption query: Adversary $\mathcal{A}_I$ sends a encryption query for a message m to $\mathcal{B}$, then the $\mathcal{B}$ returns the corresponding encryption result δ to $\mathcal{A}_I$.
- Decryption query: Adversary $\mathcal{A}_I$ sends a decryption query for a message δ to $\mathcal{B}$, then the $\mathcal{B}$ returns the corresponding decryption result m to $\mathcal{A}_I$.

Challenge: After the above queries are over, adversary $\mathcal{A}_I$ sends two plaintext $\{m_0, m_1\}$ to $\mathcal{B}$. The $\mathcal{B}$ randomly chooses a bit $d \in \{0,1\}$, calculates the encryption message δ corresponding to m_d, and final returns the δ to $\mathcal{A}_I$. $\mathcal{A}_I$ asks $\mathcal{B}$ some queries. However, the $\mathcal{A}_I$ cannot query for the decryption key of δ and cannot query for the decryption result of δ. Finally, the $\mathcal{A}_I$ outputs a bit $d^{'} \in \{0,1\}$. If $d^{'} = d$, $\mathcal{A}_I$ wins the game; otherwise, $\mathcal{A}_I$ fails.

2) Unforgeability

Definition 2: Our proposed scheme meets unforgeability if the probability that the adversary $\mathcal{A}_{II}$ could solve the ECDLP is negligible in any polynomial time. In our proposed scheme, the unforgeability is defined as existential unforgeability under chosen message attack (EUF-CMA).

Game 2: The game is an interaction between the simulator $\mathcal{B}$ and the adversary $\mathcal{A}_{II}$ under EUF-CMA. The specific definition of this game is as follows:

- Setup, Hash query, Extract signature key query, Extract verification key query: Adversary $\mathcal{A}_{II}$ and $\mathcal{B}$ perform the same operations as in Game 1.
- Verification key replacement query: Adversary $\mathcal{A}_{II}$ sends a pseudonym $PID_{i,j}^{A}$ and a valid new verification key $PK_{i,j}^{A'}$ to $\mathcal{B}$, $\mathcal{B}$ replaces the original verification key $PK_{i,j}^{A}$ with the $PK_{i,j}^{A'}$.
- Sign query: Adversary $\mathcal{A}_{II}$ sends a sign query for a message m to $\mathcal{B}$, then the $\mathcal{B}$ returns the corresponding sign result σ and some necessary parameters to $\mathcal{A}_{II}$.
- Verify signature query: Adversary $\mathcal{A}_{II}$ sends a verify signature query for a signature σ to $\mathcal{B}$, then the $\mathcal{B}$ returns the corresponding verification result to $\mathcal{A}_{II}$.

Forgery: After the above queries are over, adversary $\mathcal{A}_{II}$ returns a forged signature pair (m, σ). If the signature pair is valid, then $\mathcal{A}_{II}$ wins the game; otherwise, $\mathcal{A}_{II}$ fails. Note that in the forgery process, $\mathcal{A}_{II}$ cannot query for the signature key.

D. Security Objectives

To protect the security of the multi-receiver IIoT, the proposed scheme needs to meet the following security objectives.

1) **Message confidentiality:** To prevent privacy-sensitive data from being obtained by malicious network attackers, message senders should encrypt messages, and the encrypted messages can only be decrypted by legal message receivers.
2) **Message integrity and authentication:** To guarantee the security of the IIoT system, it is necessary to ensure the integrity of the messages and ensure that the message receiver can verify the messages' validity after receiving them. If a message has been tampered with, the receiver should detect it in time.
3) **Message anonymity:** To guarantee the privacy of smart devices, the smart devices' real identity should remain anonymous from malicious network attackers and third parties. Other than the TA in the same domain as the smart device, no other entities can obtain the real identity of the smart device.
4) **Un-linkability:** To protect privacy, malicious network attackers and third parties cannot link two messages generated by two pseudonyms.
5) **Traceability:** Suppose a message is found to be illegal. In that case, the TA in the same administrative domain as the smart device can extract the pseudonym through messages and then should be able to trace the smart device's real identity corresponding to the pseudonym.
6) **Identity revocation:** When a smart device is found to have sent an illegal message, the TA in the same administrative domain should have the ability to revoke the identity of the smart device.
7) **Resistance to attacks:** To guarantee the security of IIoT system, the proposed scheme should be able to withstand various common attacks such as the replay attack, the modification attack, and the impersonation attack.

V. Proposed Scheme

In this section, we describe our scheme in detail. As shown in Fig. 3, the proposed scheme can be divided into three main phases: system initialization, service-based key negotiation, message signing and verification. The message signing and verification phase contains two parts: message signing and encryption, message decryption and verification. In addition, the proposed scheme supports illegal message traceability and identity revocation. Therefore, to introduce the proposed scheme more clearly, in this section, we present it in the following five parts: system initialization, service-based key generation, message signing and encryption, message decryption and verification, illegal message tracing and identity revocation. The notations used in this process are shown in Table I.

TABLE I
NOTATIONS AND DEFINITIONS USED

Notations	Definitions
$BCDA^X$	Blockchain domain agent in domain X
TA^X	Trusted authority in domain X
ES^X	Edge server in domain X
SD_i^X	i-th smart device in domain X
RID_i^X	Real identity of SD_i^X
$PID_{i,j}^X$	j-th pseudonym of SD_i^X
$VP_{i,j}^X$	Validity period of $PID_{i,j}^X$
msk^X	Master secret key of TA^X
P_{pub}^X	Master public key of TA^X
sk_{ES}^X	Secret key of ES^X
PK_{ES}^X	Public key of ES^X
$sk_{i,j}^X$	j-th secret key of SD_i^X
$PK_{i,j}^X$	j-th public key of SD_i^X
$serv_t$	t-th service
sk_{serv_t}	$serv_t$-based secret key
PK_{serv_t}	$serv_t$-based public key
$Sig(\cdot)$	ECDSA signature algorithm
$Ver(\cdot)$	Verify the ECDSA signature
m	Plaintext
c	Ciphertext
σ	Signature
T_i	Timestamp
H_1, H_2, H_3, H_4, H_5	Five secure hash functions

A. System Initialization

When a new administrative domain X is added to the IIoT system, the blockchain domain agent $BCDA^X$ joins

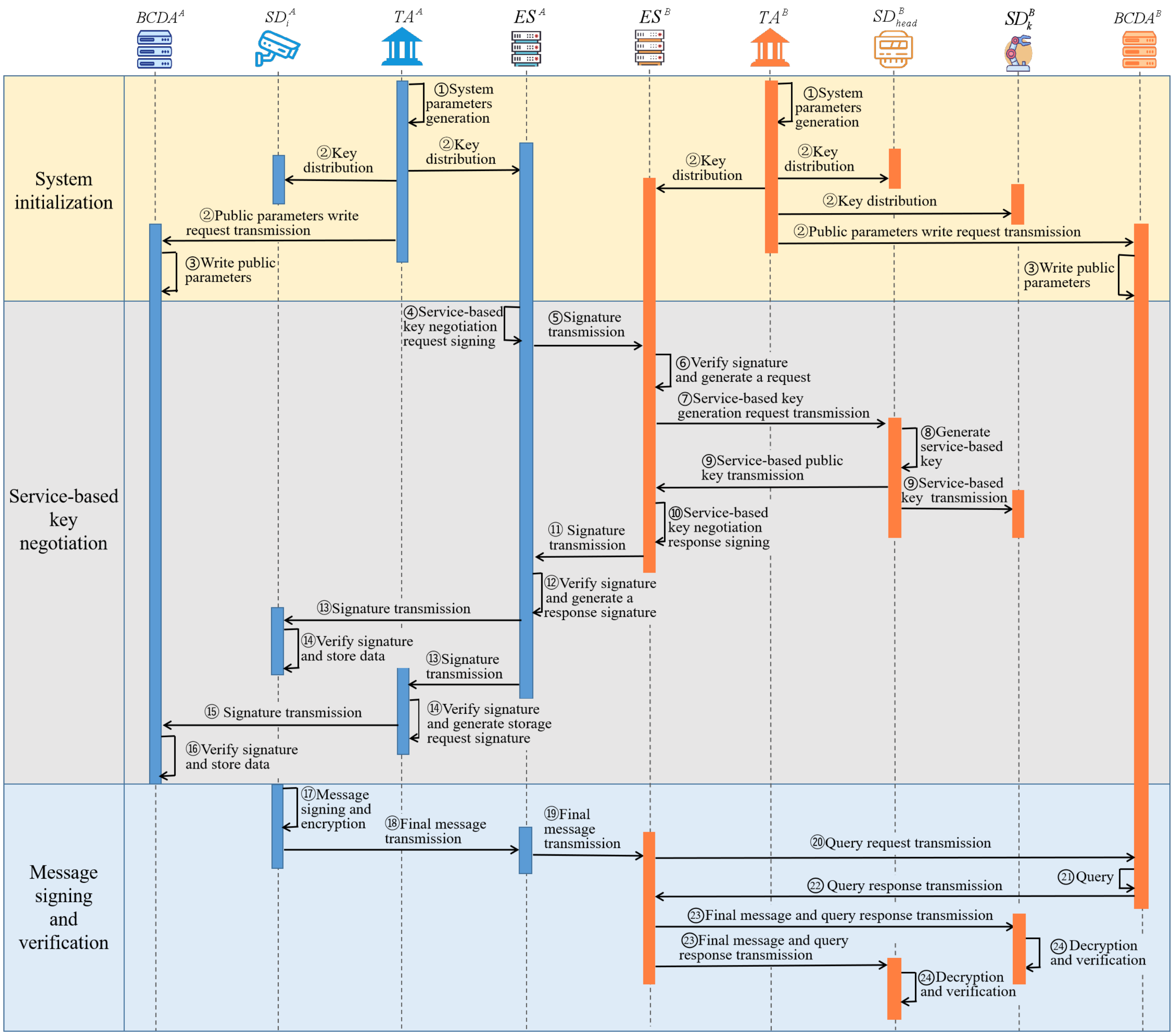

BCDA^A
SD_i^A
TA^A
ES^A
ES^B
TA^B
SD^B_head
SD^B_k
BCDA^B
System initialization
Service-based key negotiation
Message signing and verification
①System parameters generation
②Key distribution
②Public parameters write request transmission
③Write public parameters
④Service-based key negotiation request signing
⑤Signature transmission
⑥Verify signature and generate a request
⑦Service-based key generation request transmission
⑧Generate service-based key
⑨Service-based public key transmission
⑨Service-based key transmission
⑩Service-based key negotiation response signing
⑪ Signature transmission
⑫Verify signature and generate a response signature
⑬Signature transmission
⑭Verify signature and store data
⑭Verify signature and generate storage request signature
⑮ Signature transmission
⑯Verify signature and store data
⑰Message signing and encryption
⑱Final message transmission
⑲Final message transmission
⑳Query request transmission
㉑Query
㉒ Query response transmission
㉓Final message and query response transmission
㉔Decryption and verification

Fig. 3. Message authentication process overview.

the blockchain according to the strategy configured in the blockchain. Given the public parameters $(p, q, E, \mathbb{G}, \mathbb{Z}_q^*)$ for administrative domain X, then TA^X initializes the domain X as per the following steps.

1) The TA^X selects a random number $msk^X \in \mathbb{Z}_q^*$ as the system master secret key in domain X and computes the corresponding public key $P_{pub}^X = msk^X \cdot P$.
2) TA from different domains negotiate several secure one-way hash functions: $H_1 : \mathbb{G} \to \mathbb{Z}_q^*$, $H_2 : \mathbb{G} \to \{0,1\}^*$, $H_3 : \{0,1\}^* \times \mathbb{G} \times \mathbb{G} \to \mathbb{Z}_q^*$, $H_4 : \mathbb{G} \times \{0,1\}^* \to \{0,1\}^*$, $H_5 : \{0,1\}^* \to \mathbb{Z}_q^*$. Then, the system parameters $params$ will be published in IIoT, which include $\{q, \mathbb{G}, E, P, P_{pub}^X, \mathbb{Z}_q^*, H_1, H_2, H_3, H_4, H_5\}$.
3) TA^X selects a random number $sk_{ES}^X \in \mathbb{Z}_q^*$ as the secret key for ES^X and computes the corresponding public key $PK_{ES}^X = sk_{ES}^X \cdot P$. Then, the TA^X sends $\{sk_{ES}^X, PK_{ES}^X\}$ to ES^X via a secure channel.
4) When a new smart device is added to administrative domain X, the TA^X first selects a real identity RID_i^X for the smart device SD_i^X. Then, the TA^X selects a random number $r_j \in \mathbb{Z}_q^*$ and computes $sk_{i,j}^X = H_1(r_j \cdot P_{pub}) + msk^X$ as a secret key for SD_i^X. To ensure the anonymity of a message, TA^X computes pseudonym $PID_{i,j}^X = RID_i^X \oplus H_2(sk_{i,j}^X \cdot P_{pub}^X)$ and calculates public key $PK_{i,j}^X = sk_{i,j}^X \cdot P$ for SD_i^X. Subsequently, the TA^X sends $\{sk_{i,j}^X, PK_{i,j}^X, PID_{i,j}^X, VP_{i,j}^X\}$ to SD_i^X via a secure channel, where $VP_{i,j}^X$ indicates the validity period of the $PID_{i,j}^X$. Noting that $sk_{i,j}^X$ and $PID_{i,j}^X$ are updated every period of time, so TA^X will generate multiple set of data $(sk_{i,1}^X, PK_{i,1}^X, PID_{i,1}^X, VP_{i,1}^X)$, $(sk_{i,2}^X, PK_{i,2}^X, PID_{i,2}^X, VP_{i,2}^X)$, $(sk_{i,3}^X, PK_{i,3}^X, PID_{i,3}^X, VP_{i,3}^X)$, ... , $(sk_{i,n}^X, PK_{i,n}^X, PID_{i,n}^X, VP_{i,n}^X)$ to SD_i^X.

5) TA^X sends administrative domain X initial parameters $\{P_{pub}^X, PK_{ES}^X\}$ to $BCDA^X$, then the $BCDA^X$ triggers a smart contract to write these initial parameters into the blockchain.

B. Service-Based Key Negotiation

Suppose the message sender is the entity in domain A, and the receiver is in domain B. Given a new service (e.g., $serv_t$), entities from domains A and domain B need to negotiate a $serv_t$-based key before exchanging information. In our proposed scheme, the blockchain creates a list for each service, and these lists are used to query information about valid smart devices corresponding to services. For example, if the service is $serv_t$, the corresponding list is $List_{serv_t}$. And the $List_{serv_t}$ stores all valid pseudonyms, the public key, and the validity period of all smart devices that support the service $serv_t$.

It is worth noting that there are multiple message receivers for the same service. However, in the service-based key negotiation process, the $serv_t$-based public key PK_{serv_t} and the $serv_t$-based secret key sk_{serv_t} are generated by a certain smart device, which has relatively strong computing power and low computing density among all message receivers. Assume that the smart device SD_{head}^B is the generator of PK_{serv_t} and sk_{serv_t}. Negotiating the $serv_t$-based key as per the following steps.

1) The edge server ES^A sends a request $Sig_{sk_{ES}^A}(serv_t)$ for key negotiation to edge server ES^B.
2) Upon receiving the request $Sig_{sk_{ES}^A}(serv_t)$, the ES^B verifies the validity of $Sig_{sk_{ES}^A}(serv_t)$ by computing $Ver_{PK_{ES}^A}(Sig_{sk_{ES}^A}(serv_t))$. If passed, ES^B sends $serv_t$ to the corresponding smart device SD_{head}^B.
3) Upon receiving $serv_t$, SD_{head}^B selects a random number $d \in \mathbb{Z}_q^*$ and computes a $serv_t$-based public key $PK_{serv_t} = d \cdot P$. Then the smart device SD_{head}^B uses its secret key sk_{head} to calculate $serv_t$-based secret key $sk_{serv_t} = d + sk_{head}^B \cdot h$, where $h = H_3(ID_{ES}^B, PK_{ES}^B, PK_{ES}^A)$. Subsequently, SD_{head}^B sends PK_{serv_t} to ES^B and sends $\{sk_{serv_t}, PK_{serv_t}\}$ to other smart devices in administrative domain B that support the same service via a secure channel.
4) Upon receiving PK_{serv_t}, the edge server ES^B generates signature $Sig_{sk_{ES}^B}(PK_{serv_t}, PK_{head}^B)$ and sends it to ES^A.
5) Once ES^A receives the signature $Sig_{sk_{ES}^B}(PK_{serv_t}, PK_{head}^B)$, it first verifies the validity of the signature by computing $Ver_{PK_{ES}^B}(Sig_{sk_{ES}^B}(PK_{serv_t}, PK_{head}^B))$. If passed, ES^A sends $Sig_{sk_{ES}^A}(ID_{ES}^B, PK_{ES}^B, PK_{serv_t}, PK_{head}^B)$ to corresponding smart devices and trusted authority TA^A in administrative domain A.
6) Once SD receives $Sig_{sk_{ES}^A}(ID_{ES}^B, PK_{ES}^B, PK_{serv_t}, PK_{head}^B)$, it first verifies the legitimacy of the signature, if it is not legitimate, directly discard; otherwise, store the corresponding $serv_t$-based data.
7) The TA^A submits a signature $Sig_{msk^A}(Store\_req, serv_t, PIDS, SDPKS, VPS)$ to $BCDA^A$, where $Store\_req$ indicates a storage request, $PIDS$ indicates a set of all valid pseudonyms based on $serv_t$, $SDPKS$ denotes the public key set corresponding to $PIDS$, VPS indicates the validity period set corresponding to $PIDS$. When the $BCDA^A$ receives the signature, it verifies whether the signature is valid. If it is invalid, $BCDA^A$ discards the request directly; otherwise, $BCDA^A$ stores $\{PIDS, SDPKS, VPS\}$ into $List_{serv_t}$.

C. Message Signing And Encryption

Assume that the smart device SD_i^A in administrative domain A is the message sender and the smart device SD_k^B in administrative domain B is the message receiver. To ensure the security of messages, SD_i^A needs to sign and encrypt the message before sending them, as explained below.

1) The message sender SD_i^A selects two random numbers $w, u \in \mathbb{Z}_q^*$, and computes $W = w \cdot P$, $U = u \cdot P$ respectively. Then the SD_i^A calculates $h_3 = H_3(PID_{i,j}^A, W, U)$ and $h = H_3(ID_{ES}^B, PK_{ES}^B, PK_{ES}^A)$.
2) When a smart device SD_i^A wants to send a plaintext m, it first selects current timestamp T_i and computes signature $\sigma = sk_{i,j}^A \cdot (H_5(m) \cdot w + (h + h_3) \cdot u)^{-1}$. Then it calculates ciphertext $c = (s, Y_i^A)$, where $s = H_4(U, T_i) \oplus (m||\sigma)$, $Y_i^A = u(PK_{serv_t} + h \cdot PK_{head}^B)$.
3) SD_i^A sends the final message $\delta = (c, W, PID_{i,j}^A, T_i)$ to SD_k^B through ES^A and ES^B.

Remark 1: In the message signing and encryption process, some data (e.g., $w, u, W, U, h_3, h, Y_i^A$) generation does not require online messages m. Therefore, when the smart device is idle or computational density is not high, the smart device can generate these data in advance through pre-processing and store them for generating online signatures in the future. Using this pre-processing approach can effectively improve the signing efficiency of the smart device.

D. Message Decryption And Verification

Upon receiving a message $\delta' = (c', W', PID_{i,j}^{A'}, T_i')$, ES^B queries the blockchain to check if the $PID_{i,j}^{A'}$ is valid, and if so, forwards the corresponding data to SD_k^B. SD_k^B first decrypts the ciphertext c' by using sk_{serv_t} and then verifies the integrity of the received signature σ'. The main process is divided into the following steps.

1) When ES^B receives the final message δ', it first queries the blockchain whether the $PID_{i,j}^{A'}$ exists in $List_{serv_t}$ and has not expired. If the result of the query is that the $PID_{i,j}^{A'}$ does not exist or has expired, ES^B will directly discard the final message; otherwise, ES^B will obtain the $PK_{i,j}^{A'}$ corresponding to the $PID_{i,j}^{A'}$ from the blockchain. Finally, ES^B sends the δ' and $PK_{i,j}^{A'}$ to the corresponding smart device SD_k^B.
2) Upon receiving this message δ', SD_k^B checks the freshness of T_i'. Assume that the time of receiving message is T. If $\Delta T \geq T - T_i'$, SD_k^B continues; otherwise, SD_k^B discards the message δ'.
3) To obtain plaintext m and signature σ, SD_k^B computes $U' = Y_i^{A'} \cdot sk_{serv_t}^{-1}$, and then calculates $m'||\sigma' = H_4(U', T_i) \oplus s'$.

4) To verify the validity and integrity of signature $\sigma^{'}$, SD_k^B computes $h_3^{'} = H_3(PID_{i,j}^{A'}, W^{'}, U^{'})$ and then calculates $h = H_3(ID_{ES}^B, PK_{ES}^B, PK_{ES}^A)$. Subsequently, SD_k^B checks whether the formula $PK_{i,j}^{A'} = \sigma^{'} \cdot (H_5(m^{'}) \cdot W^{'} + (h + h_3^{'}) \cdot U^{'})$ holds true or not. If not, SD_k^B rejects the $\delta^{'}$; otherwise, the $\delta^{'}$ be considered legal.

Due to $PK_{serv_t} = d{\cdot}P$, $sk_{serv_t} = d{+}sk_{head}^B{\cdot}h$, $W = w{\cdot}P$, $PK_{i,j}^A = sk_{i,j}^A \cdot P$, $U = u \cdot P$, the correctness of the $U^{'}$ can be ensured using the below formula.

$$\begin{aligned} U^{'} &= Y_i^{A'} \cdot sk_{serv_t}^{-1} \\ &= u(PK_{serv_t} + h \cdot PK_{head}^B) \cdot sk_{serv_t}^{-1} \\ &= u(d \cdot P + sk_{serv_t} \cdot P - d \cdot P) \cdot sk_{serv_t}^{-1} \\ &= (u \cdot sk_{serv_t} \cdot P) \cdot sk_{serv_t}^{-1} \\ &= u \cdot P \\ &= U \end{aligned} \tag{1}$$

The correctness of the verification can be ensured using the below formula.

$$\begin{aligned} &\sigma^{'} \cdot (W^{'} \cdot H_5(m^{'}) + (h + h_3^{'}) \cdot U^{'}) \\ =&sk_{i,j}^A \cdot (w \cdot H_5(m) + (h + h_3) \cdot u)^{-1} \\ &\cdot (W \cdot H_5(m) + (h + h_3) \cdot U) \\ =&sk_{i,j}^A \cdot (w \cdot H_5(m) + (h + h_3) \cdot u)^{-1} \\ &\cdot (w \cdot H_5(m) + (h + h_3) \cdot u) \cdot P \\ =&sk_{i,j}^A \cdot P \\ =&PK_{i,j}^A \end{aligned} \tag{2}$$

Remark 2: The pre-processing approach can also improve message decryption and verification efficiency like the message signing and encryption process. When the message has not yet arrived, and the smart device is idle or computational density is not high, some data (e.g., h) can be pre-processed offline and stored for future use in decrypting and verifying received messages.

E. Illegal Message Tracing And Identity Revocation

In the proposed scheme, if it is found that a smart device in the administrative domain X has published an illegal message, then TA^X traces the source of the illegal messages and revokes the identity of the smart device as per the following steps.

1) TA^X queries the $PK_{i,j}^X$ corresponding to the $PID_{i,j}^X$, and then computes $temp = msk^X \cdot PK_{i,j}^X$, $RID_i^X = PID_{i,j}^X \oplus H_2(temp)$.
2) TA^X stops regenerating the pseudonym $PID_{i,j}^X$ and secret key $sk_{i,j}^X$ corresponding to the RID_i^X. Then the TA^X submits a signature $Sig_{msk^X}(Del\_req, PIDS_i^X)$ to the $BCDA^X$ to delete the $PIDS_i^X$, where Del_req indicates delete request and $PIDS_i^X$ indicates a set of all pseudonyms corresponding to the RID_i^X.
3) Upon receiving the signature $Sig_{msk^X}(Del\_req, PIDS_i^X)$, $BCDA^X$ first verifies the validity of the signature by computing $Ver_{P_{pub}^X}(Sig_{msk^X}(Del\_req, PIDS_i^X))$. If the signature is valid, $BCDA^X$ will delete the $\{PIDS_i^X$, PKS_i^X, $VPS_i^X\}$ in corresponding list, where PKS_i^X denotes the public key set corresponding to $PIDS_i^X$, VPS_i^X indicates the validity period set corresponding to $PIDS_i^X$.

VI. Security Proof And Analysis

In this section, we first demonstrate that our proposed scheme is secure through security proof. Then we show that our proposed scheme can resist various common attacks through security analysis.

A. Security Proof

This subsection will perform detailed security proof to prove the security of our proposed scheme.

1) Confidentiality

Theorem 1. In the random oracle model, if an adversary $\mathcal{A}_I$ with probabilistic polynomial time executes **Game 1** and wins the game with a non-negligible probability ε_I, then the simulator $\mathcal{B}$ with probabilistic polynomial time can solve the ECDHP problem with a non-negligible probability no less than $(1 - \frac{q_{dsk}}{2^\lambda})\frac{\varepsilon_I}{e(q_{enc}+q_{dsk})}$, where q_{dsk} denotes the maximum times of extracting decryption key queries, q_{enc} denotes the maximum times of encryption queries, λ denotes secure parameter, e denotes natural logarithm base.

Proof. If there is an adversary $\mathcal{A}_I$ that can break the proposed scheme with a non-negligible probability ε_I, then we can construct a simulator $\mathcal{B}$ based on $\mathcal{A}_I$, and the $\mathcal{B}$ can solve the ECDHP run by $\mathcal{A}_I$ as a subroutine with non-negligible probability. Given a group $\mathbb{G}$ and an ECDHP instance $\{P, aP, bP | a, b \in \mathbb{Z}_q^*\}$, $\mathcal{B}$ simulates oracles queried by $\mathcal{A}_I$ as follows.

Setup: Upon receiving the setup query from $\mathcal{A}_I$, the $\mathcal{B}$ sends the public parameters $\{E, \mathbb{G}, P, P_{pub}, H_i(i = 1, 2, 3, 4, 5)\}$ to $\mathcal{A}_I$.

H_1**-query:** For the query, $\mathcal{B}$ presets a map Map_{H1}, and the Map_{H1} is empty at the beginning. When the adversary $\mathcal{A}_I$ makes an H_1 query with $< X_i >$, where X_i denotes an elliptic curve point, $\mathcal{B}$ checks whether the Map_{H1} has the key $< X_i >$. If so, $\mathcal{B}$ finds the corresponding value and returns it to $\mathcal{A}_I$. Otherwise, $\mathcal{B}$ chooses a random number $r_{H_1} \in \mathbb{Z}_q^*$, and r_{H_1} should satisfy $r_{H_1} \notin Map_{H1}$. Then the $\mathcal{B}$ sets the value $Map_{H1}(< X_i >) = r_{H_1}$, and returns r_{H_1} to $\mathcal{A}_I$.

H_2**-query:** For the query, $\mathcal{B}$ presets a map Map_{H2}, and the Map_{H2} is empty at the beginning. When the adversary $\mathcal{A}_I$ makes an H_2 query with $< SPK >$, where SPK denotes an elliptic curve point, $\mathcal{B}$ checks whether the Map_{H2} has the key $< SPK >$. If so, $\mathcal{B}$ finds the corresponding value and returns it to $\mathcal{A}_I$. Otherwise, $\mathcal{B}$ chooses a random bit-string $r_{H_2} \in \{0,1\}^*$, and r_{H_2} should satisfy $r_{H_2} \notin Map_{H2}$. Then the $\mathcal{B}$ sets the value $Map_{H2}(< SPK >) = r_{H_2}$, and returns r_{H_2} to $\mathcal{A}_I$.

H_3**-query:** For the query, $\mathcal{B}$ presets a map Map_{H3}, and the Map_{H3} is empty at the beginning. When the adversary $\mathcal{A}_I$ makes an H_3 query with $< ID_{ES}^B, PK_{ES}^B, PK_{ES}^A >$, $\mathcal{B}$ checks whether the Map_{H3} has the key $< ID_{ES}^B, PK_{ES}^B, PK_{ES}^A >$. If so, $\mathcal{B}$ finds the corresponding value and returns it to $\mathcal{A}_I$.

Otherwise, $\mathcal{B}$ chooses a random number $r_{H_3} \in \mathbb{Z}_q^*$, and r_{H_3} should satisfy $r_{H_3} \notin Map_{H3}$. Then the $\mathcal{B}$ sets the value $Map_{H3}(< ID_{ES}^B, PK_{ES}^B, PK_{ES}^A >) = r_{H_3}$, and returns r_{H_3} to $\mathcal{A}_I$.

H_4**-query:** For the query, $\mathcal{B}$ presets a map Map_{H4}, and the Map_{H4} is empty at the beginning. When the adversary $\mathcal{A}_I$ makes an H_4 query with $< K, T_i >$, $\mathcal{B}$ checks whether the Map_{H4} has the key $< K, T_i >$. If so, $\mathcal{B}$ finds the corresponding value and returns it to $\mathcal{A}_I$. Otherwise, $\mathcal{B}$ chooses a random bit-string $r_{H_4} \in \{0,1\}^*$, and r_{H_4} should satisfy $r_{H_4} \notin Map_{H4}$. Then the $\mathcal{B}$ sets the value $Map_{H4}(< K, T_i >) = r_{H_4}$, and returns r_{H_4} to $\mathcal{A}_I$.

H_5**-query:** For the query, $\mathcal{B}$ presets a map Map_{H5}, and the Map_{H5} is empty at the beginning. When the adversary $\mathcal{A}_I$ makes an H_5 query with $< m >$, where $m \in \{0,1\}^*$. $\mathcal{B}$ checks whether the Map_{H5} has the key $< m >$. If so, $\mathcal{B}$ finds the corresponding value and returns it to $\mathcal{A}_I$. Otherwise, $\mathcal{B}$ chooses a random number $r_{H_5} \in \mathbb{Z}_q^*$, and r_{H_5} should satisfy $r_{H_5} \notin Map_{H5}$. Then the $\mathcal{B}$ sets the value $Map_{H5}(< m >) = r_{H_5}$, and returns r_{H_5} to $\mathcal{A}_I$.

Extract encryption key query: For the query, $\mathcal{B}$ presets a map Map_{EK}, and the Map_{EK} is empty at the beginning. When the adversary $\mathcal{A}_I$ makes an extract encryption key query with a service $< serv_t >$, $\mathcal{B}$ checks whether the Map_{EK} has the key $< serv_t >$. If so, $\mathcal{B}$ finds the corresponding value and returns it to $\mathcal{A}_I$. Otherwise, the $\mathcal{B}$ chooses a random number $c_t \leftarrow \{0,1\}$, and $Pr[c_t = 1] = \frac{1}{q_{enc}+q_{dsk}+1}$, where the "1" in the denominator denotes that $\mathcal{A}_I$ has selected one service as the challenge service. If $c_t = 0$, the $\mathcal{B}$ chooses two random numbers $sk_{serv_t}, r_H \in \mathbb{Z}_q^*$, then the $\mathcal{B}$ computes $PK_{serv_t} = sk_{serv_t} \cdot P - r_H \cdot PK_{head}$, where the PK_{serv_t} should satisfy $PK_{serv_t} \notin Map_{EK}$. Final the $\mathcal{B}$ sets $Map_{EK}(< serv_t >) = PK_{serv_t}$, and returns PK_{serv_t} to $\mathcal{A}_I$. At the same time, the $\mathcal{B}$ sets $Map_{dsk}(< serv_t >) = sk_{serv_t}$. If $c_t = 1$, let $PK_{serv_t} = r_{know} \cdot P$, where the $r_{know} \in \mathbb{Z}_q^*$ that $\mathcal{B}$ already knows. The PK_{serv_t} should satisfy $PK_{serv_t} \notin Map_{EK}$. Otherwise, $\mathcal{B}$ re-chooses r_{know}, then $\mathcal{B}$ sets the value $Map_{EK}(< serv_t >) = PK_{serv_t}$, and final returns PK_{serv_t} to $\mathcal{A}_I$.

Extract decryption key query: For the query, $\mathcal{B}$ presets a map Map_{dsk}, and the Map_{dsk} is empty at the beginning. When the adversary $\mathcal{A}_I$ makes an extract decryption key query with a service $< serv_t >$, $\mathcal{B}$ checks whether the Map_{dsk} has the key $< serv_t >$. If so, $\mathcal{B}$ finds the corresponding value and returns it to $\mathcal{A}_I$. Otherwise, the $\mathcal{B}$ first conducts extract encryption key query and obtains the corresponding response $< PK_{serv_t} >$. If $c_t = 0$, it means that $\mathcal{B}$ has been set $Map_{dsk}(< serv_t >) = sk_{serv_t}$, and then the $\mathcal{B}$ returns the corresponding sk_{serv_t} to $\mathcal{A}_I$. If $c_t = 1$, $\mathcal{B}$ will abort the process.

Extract verification key query: For the query, $\mathcal{B}$ presets a map Map_{VK}, and the Map_{VK} is empty at the beginning. When the adversary $\mathcal{A}_I$ makes an extract verification key query with a pseudonym $< PID_{i,j}^A >$, $\mathcal{B}$ checks whether the Map_{VK} has the key $< PID_{i,j}^A >$. If so, $\mathcal{B}$ finds the corresponding value and returns it to $\mathcal{A}_I$. Otherwise, the $\mathcal{B}$ chooses a random number $g_{i,j} \leftarrow \{0,1\}$, and $Pr[g_{i,j} = 1] = \frac{1}{q_s+q_{ssk}+1}$ where the "1" in the denominator denotes that $\mathcal{A}_I$ has selected one pseudonym as the challenge pseudonym, q_{ssk} denotes the maximum times of extracting signature key queries, q_s denotes the maximum times of sign queries. If $g_{i,j} = 0$, the $\mathcal{B}$ chooses a random numbers $sk_i \in \mathbb{Z}_q^*$, then the $\mathcal{B}$ computes $PK_{i,j}^A = sk_{i,j}^A \cdot P$, where the $PK_{i,j}^A$ should satisfy $PK_{i,j}^A \notin Map_{VK}$. Final the $\mathcal{B}$ sets $Map_{VK}(< PID_{i,j}^A >) = PK_{i,j}^A$, and returns $PK_{i,j}^A$ to $\mathcal{A}_I$. At the same time, the $\mathcal{B}$ sets $Map_{ssk}(< PID_{i,j}^A >) = sk_i$. If $g_{i,j} = 1$, let $PK_{i,j}^A = r_{know2} \cdot P$, where the $r_{know2} \in \mathbb{Z}_q^*$ that $\mathcal{B}$ already knows. The $PK_{i,j}^A$ should satisfy $PK_{i,j}^A \notin Map_{VK}$. Otherwise, $\mathcal{B}$ re-chooses r_{know2}, then $\mathcal{B}$ sets the value $Map_{VK}(< PID_{i,j}^A >) = PK_{i,j}^A$, and final returns $PK_{i,j}^A$ to $\mathcal{A}_I$.

Extract signature key query: For the query, $\mathcal{B}$ presets a map Map_{ssk}, and the Map_{ssk} is empty at the beginning. When the adversary $\mathcal{A}_I$ makes an extract signature key query with a pseudonym $< PID_{i,j}^A >$, $\mathcal{B}$ checks whether the Map_{ssk} has the key $< PID_{i,j}^A >$. If so, $\mathcal{B}$ finds the corresponding value and returns it to $\mathcal{A}_I$. Otherwise, the $\mathcal{B}$ first conducts extract verification key query and obtains the corresponding response $< PK_{i,j}^A >$. If $g_{i,j} = 0$, it means that $\mathcal{B}$ has been set $Map_{ssk}(< PID_{i,j}^A >) = sk_i$, and then the $\mathcal{B}$ returns the corresponding $sk_{i,j}^A$ to $\mathcal{A}_I$. If $g_{i,j} = 1$, $\mathcal{B}$ will abort the process.

Encryption key replacement query: For the query, the adversary $\mathcal{A}_I$ can randomly choose a new encryption key $PK_{serv_t}^{'}$ to replace the original encryption key PK_{serv_t} corresponding to $serv_t$.

Encryption query: When the adversary $\mathcal{A}_I$ makes an encryption query with $< m, ID_{ES}^A, ID_{ES}^B, PID_{i,j}^A, serv_\alpha, T_i >$ (assume that $\mathcal{A}_I$ has already executed extract encryption key query with $< serv_\alpha >$), $\mathcal{B}$ queries the PK_{serv_α} corresponding to $serv_\alpha$. If $c_\alpha = 1$, $\mathcal{B}$ will abort the process; otherwise, $\mathcal{B}$ obtains $sk_{i,j}^A$ and PK_{serv_α} by querying Map_{ssk} and Map_{EK} respectively. Then, $\mathcal{B}$ runs the $Sign$ algorithm and $Encrypt$ algorithm to generate the final encryption message $\delta = (c = (s, Y_i), W, PID_{i,j}^A, PK_{i,j}^A, T_i)$, final the $\mathcal{B}$ sends δ to $\mathcal{A}_I$.

Decryption query: When the adversary $\mathcal{A}_I$ makes a decryption query with $< \delta, ID_{ES}^A, ID_{ES}^B, serv_\alpha >$ (assume that $\mathcal{A}_I$ has already executed extract encryption key query with $< serv_\alpha >$), $\mathcal{B}$ queries the PK_{serv_α} corresponding to $serv_\alpha$. If $c_\alpha = 0$, $\mathcal{B}$ obtains sk_{serv_α} by querying Map_{dsk}. Then $\mathcal{B}$ runs the $Decrypt$ algorithm to obtain m, final the $\mathcal{B}$ returns the m to $\mathcal{A}_I$, if the δ is invalid, the $\mathcal{B}$ will return $\perp$. If $c_\alpha = 1$, $\mathcal{B}$ obtains r_H, r_{H_3} by querying Map_{H3}, $\mathcal{B}$ obtains r_{H_4} by querying Map_{H4}. Then the $\mathcal{B}$ calculates $m||\sigma = s \oplus r_{H_4}$ and obtains r_{H_5} by querying Map_{H5}, if $PK_{i,j}^A = \sigma(r_{H_5} \cdot W + (r_H + r_{H_3}) \cdot U)$, $\mathcal{B}$ returns the m to $\mathcal{A}_I$. If δ is invalid, the $\mathcal{B}$ will return $\perp$. If there is no value for $< serv_\alpha >$ in Map_{EK}, it means that encryption key was replaced. Then the $\mathcal{B}$ obtains $r_H^{'}$, $r_{H_3}^{'}$ by querying Map_{H3} and obtains $r_{H_4}^{'}$ by querying Map_{H4}. Next, the $\mathcal{B}$ calculates $m||\sigma = s \oplus r_{H_4}^{'}$ and obtains $r_{H_5}^{'}$ by querying Map_{H5}, if $PK_{i,j}^A = \sigma(r_{H_5}^{'} \cdot W + (r_H^{'} + r_{H_3}^{'}) \cdot U)$, $\mathcal{B}$ returns the m to $\mathcal{A}_I$. Otherwise, $\mathcal{B}$ outputs $\perp$.

Challenge: Adversary $\mathcal{A}_I$ outputs a challenged pseudonym $PID_{e,f}^A$, a challenged service $serv_\alpha$ and two plaintext m_0, m_1.

$\mathcal{B}$ executes extract encryption key query on $serv_\alpha$, and obtains the value PK_{serv_α}. If $c_\alpha = 0$, $\mathcal{B}$ aborts the process. Otherwise, $\mathcal{B}$ chooses random number $a, b, w \in \mathbb{Z}_q^*$, sets $U = a \cdot P$ and $PK_{head}^B = b \cdot P$. Then the $\mathcal{B}$ obtains r_H, r_{H_3} by querying Map_{H3} and obtains r_{H_4}, r_{H_5} by querying Map_{H4}, Map_{H5} respectively. Next, $\mathcal{B}$ computes $\sigma = sk_{e,f}^A \cdot (r_{H_5} \cdot w + (r_H + r_{H_3}) \cdot a)^{-1}$, $s = r_{H_4} \oplus (m_d || \sigma)$, $Y_i^A = a \cdot (PK_{serv_\alpha} + r_H \cdot PK_{head}^B)$. Finally, $\mathcal{B}$ sends the challenged message $\delta = (c = (s, Y_i^A), W, PID_{e,f}^A, PK_{e,f}^A, T_i)$ to $\mathcal{A}_I$.

After Adversary $\mathcal{A}_I$ executes the above-mentioned query of the probability polynomial time and outputs the guess value $d^{'} \in \{0, 1\}$. If $d^{'} = d$, $\mathcal{B}$ outputs $abP = \frac{1}{r_H}(Y_i^A - r_{know} U)$ as the valid solution of the ECDHP, where $Y_i^A = a(PK_{serv_\alpha} + r_H \cdot PK_{head}^B) = (r_{know} + r_H \cdot b) \cdot U$, $U = aP$, $r_H = H_3(ID_{ES}^B, PK_{ES}^B, PK_{ES}^A)$. Otherwise, $\mathcal{B}$ did not solve the ECDHP.

If the simulator $\mathcal{B}$ does not abort during the simulation, and the adversary $\mathcal{A}_I$ broke the confidentiality of the scheme with a non-negligible probability ε_I, then $\mathcal{B}$ outputs the valid solution of ECDHP. Assume that $\varepsilon_I^{'}$ denotes $\mathcal{A}_I$ did not perform extract decryption key query for the challenged service, then $Pr[\varepsilon_I^{'}] = 1 - \frac{q_{dsk}}{2^\lambda}$; $\varepsilon_I^{''}$ denotes $\mathcal{B}$ did not abort during the query phase, then $Pr[\varepsilon_I^{''}] = (1-\tau)^{q_{enc}+q_{dsk}}$; $\varepsilon_I^{'''}$ denotes $\mathcal{B}$ did not abort during the challenge phase, then $Pr[\varepsilon_I^{'''}] = \tau$.

The probability that $\mathcal{B}$ does not abort in the entire simulation process is $Pr[\varepsilon_I^{'} \wedge \varepsilon_I^{''} \wedge \varepsilon_I^{'''}] = (1 - \frac{q_{dsk}}{2^\lambda})(1-\tau)^{q_{enc}+q_{dsk}}\tau$, where $\tau = \frac{1}{q_{enc}+q_{dsk}+1}$. When $q_{enc} + q_{dsk}$ is large enough, $(1 - \frac{1}{q_{enc}+q_{dsk}+1})^{q_{enc}+q_{dsk}+1}$ tends to e^{-1}. Therefore, the probability that $\mathcal{B}$ does not abort during the simulation is at least $(1 - \frac{q_{dsk}}{2^\lambda})\frac{1}{e(q_{enc}+q_{dsk})}$.

In summary, if simulator $\mathcal{B}$ does not abort during the simulation process, and adversary $\mathcal{A}_I$ break the confidentiality of the scheme with a non-negligible probability ε_I, then the simulator $\mathcal{B}$ with probabilistic polynomial time outputs the valid solution of ECDHP with a non-negligible probability no less than $(1 - \frac{q_{dsk}}{2^\lambda})\frac{\varepsilon_I}{e(q_{enc}+q_{dsk})}$.

2) Unforgeability

Theorem 2. In the random oracle model, if an adversary $\mathcal{A}_{II}$ with probabilistic polynomial time executes **Game 2** and wins the game with a non-negligible probability ε_{II}, then the simulator $\mathcal{B}$ with probabilistic polynomial time can solve the ECDLP problem with a non-negligible probability no less than $(1 - \frac{q_{ssk}}{2^\lambda})\frac{\varepsilon_{II}}{e(q_s+q_{ssk})}$, where q_{ssk} denotes the maximum times of extracting signature key queries, q_s denotes the maximum times of sign queries.

Proof. If there is an adversary $\mathcal{A}_{II}$ that can break our proposed scheme with non-negligible probability ε_{II}, then we can construct a simulator $\mathcal{B}$ based on $\mathcal{A}_{II}$, and the $\mathcal{B}$ can solve the ECDLP run by $\mathcal{A}_{II}$ as a subroutine with a non-negligible probability. Given a group $\mathbb{G}$ and an ECDLP instance $\{P, Q = bP | b \in \mathbb{Z}_q^*, P \in \mathbb{G}, Q \in \mathbb{G}\}$, $\mathcal{B}$ simulates oracles queried by $\mathcal{A}_{II}$ as follows.

Adversary $\mathcal{A}_{II}$ executes setup, H_1, H_2, H_3, H_4, H_5 query, extract verification key query and extract signature key query of the random oracle in **Theorem 1**.

Verification key replacement query: For the query, the adversary $\mathcal{A}_{II}$ can randomly choose a new verification key $PK_{i,j}^{A'}$ to replace the original verification key $PK_{i,j}^A$ corresponding to $PID_{i,j}^A$.

Sign-query: For the query, $\mathcal{B}$ presets a map Map_{sig}, and the Map_{sig} is empty at the beginning. When the adversary $\mathcal{A}_{II}$ makes a signing query with $< m, PID_{e,f}^A >$ (assume that $\mathcal{A}_{II}$ has already executed extract verification key query with $< PID_{e,f}^A >$), $\mathcal{B}$ first queries the value $sk_{e,f}^A$ of Map_{ssk}. If $g_{e,f} = 1$, $\mathcal{B}$ aborts the process. Otherwise, $\mathcal{B}$ chooses some random numbers $w, u \in \mathbb{Z}_q^*$, then $\mathcal{B}$ obtains r_H, r_{H_3} by executing H_3-query and obtains r_{H_5} by executing H_5-query. Then, $\mathcal{B}$ runs the $Sign$ algorithm and $Encrypt$ algorithm to generate the final encryption message $\delta = (c = (s, Y_i), W, PID_{e,f}^A, PK_{e,f}^A, T_i)$, final the $\mathcal{B}$ sends δ to $\mathcal{A}_{II}$.

Verify signature query: When the adversary $\mathcal{A}_{II}$ makes a verify signature query with $\delta = (c = (s, Y_i), W, PID_{e,f}^A, T_i)$ (assume that $\mathcal{A}_{II}$ has already executed extract verification key query with $< PID_{e,f}^A >$), $\mathcal{B}$ queries the $PK_{e,f}^A$ corresponding to $PID_{e,f}^A$. If $g_{e,f} = 0$, $\mathcal{B}$ runs the $Decrypt$ algorithm to obtain m, final the $\mathcal{B}$ returns the m to $\mathcal{A}_{II}$. If the δ is invalid, the $\mathcal{B}$ will return $\perp$. If $g_{e,f} = 1$, $\mathcal{B}$ obtains r_H, r_{H_3}, r_{H_4} by querying Map_{H3}, Map_{H3} and Map_{H4} respectively. Then the $\mathcal{B}$ calculates $m||\sigma = s \oplus r_{H_4}$ and obtains r_{H_5} by querying Map_{H5}, if $PK_{e,f}^A = \sigma(r_{H_5} \cdot W + (r_H + r_{H_3}) \cdot U)$, $\mathcal{B}$ returns the m to $\mathcal{A}_{II}$. If δ is invalid, the $\mathcal{B}$ will return $\perp$. If there is no value for $< PID_{e,f}^A >$ in Map_{VK}, it means that verification key was replaced. Then the $\mathcal{B}$ obtains $r_H^{'}$, $r_{H_3}^{'}$, $r_{H_4}^{'}$ by querying Map_{H3}, Map_{H3} and Map_{H4} respectively. Next, the $\mathcal{B}$ calculates $m||\sigma = s \oplus r_{H_4}^{'}$ and obtains $r_{H_5}^{'}$ by querying Map_{H5}, if $PK_{e,f}^A = \sigma(r_{H_5}^{'} \cdot W + (r_H^{'} + r_{H_3}^{'}) \cdot U)$, $\mathcal{B}$ returns the m to $\mathcal{A}_{II}$. Otherwise, $\mathcal{B}$ outputs $\perp$.

Forgery: After executes the above-mentioned query of the probability polynomial time, adversary $\mathcal{A}_{II}$ outputs a forged signature σ. If $\mathcal{A}_{II}$ successfully forges the signature and $g_{e,f} = 1$, then $\mathcal{B}$ outputs $b = r_{know2}$ as the valid solution of ECDLP. Otherwise, $\mathcal{B}$ did not solve ECDLP.

Assume that $\varepsilon_{II}^{'}$ denotes $\mathcal{A}_{II}$ did not perform extract signature key query for the challenge pseudonym $PID_{e,f}^A$, then $Pr[\varepsilon_{II}^{'}] = 1 - \frac{q_{ssk}}{2^\lambda}$. Assume that $\varepsilon_{II}^{''}$ denotes $\mathcal{B}$ did not abort during querying phase, then $Pr[\varepsilon_{II}^{''}] = (1-\tau)^{q_s+q_{ssk}}$. Assume that $\varepsilon_{II}^{'''}$ denotes $\mathcal{B}$ did not abort in the forgery phase, then $Pr[\varepsilon_{II}^{'''}] = \tau$. Hence, the probability that $\mathcal{B}$ does not abort in the entire simulation process is $Pr[\varepsilon_{II}^{'} \wedge \varepsilon_{II}^{''} \wedge \varepsilon_{II}^{'''}] = (1 - \frac{q_{ssk}}{2^\lambda})(1-\tau)^{q_s+q_{ssk}}\tau$, where $\tau = \frac{1}{q_s+q_{ssk}+1}$. When $q_s + q_{ssk}$ is large enough, $(1 - \frac{1}{q_s+q_{ssk}+1})^{q_s+q_{ssk}+1}$ tends to e^{-1}. Therefore, the probability that $\mathcal{B}$ does not abort during the simulation is at least $(1 - \frac{q_{ssk}}{2^\lambda})\frac{1}{e(q_s+q_{ssk})}$.

In summary, if simulator $\mathcal{B}$ does not abort during the simulation process, and adversary $\mathcal{A}_{II}$ broke the unforgeability of the scheme with a non-negligible probability ε_{II}, then the simulator $\mathcal{B}$ with probabilistic polynomial time outputs the valid solution of ECDLP with a non-negligible probability of no less than $(1 - \frac{q_{ssk}}{2^\lambda})\frac{\varepsilon_{II}}{e(q_s+q_{ssk})}$. □

B. Security Analysis

This subsection gives a detailed analysis of various security features that our proposed scheme satisfies.

1) Message confidentiality: According to **Theorem 1**, we know that no polynomial adversary can challenge success if the ECDHP is hard. Before sending a message, the smart device first encrypts the plaintext m by calculating $s = H_4(U, T_i) \oplus (m||\sigma)$. During transmission, messages are in ciphertext, and u, U are stored in the message sender. Through calculation or guessing, malicious network attackers and illegal receivers cannot obtain u or U. In addition, only the legitimate message receiver with the service-based secret key sk_{serv} can obtain U by computing $Y_i^A \cdot sk_{serv}^{-1}$, and further, obtain the plaintext m by calculating $H_4(U^{'}, T_i) \oplus s$. Therefore, the scheme can guarantee the confidentiality of the message.

2) Message integrity and authentication: According to the **Theorem 2**, it can be concluded that if the ECDLP is difficult to solve, then no adversary can forgery a legal signature within a given polynomial time. In addition, as long as the message and signature meet $PK_{i,j}^{A'} = \sigma^{'}(H_5(m^{'}) \cdot W^{'} + (h + h_3^{'}) \cdot U^{'})$, the signature $\sigma^{'}$ is proven to be valid. Therefore, the scheme can guarantee the integrity and authentication of the message.

3) Message anonymity: In the process of cross-domain authentication, the smart device SD_i^X does not use its real identity when signing and encrypting, but uses pseudonym $PID_{i,j}^X = RID_i^X \oplus H_2(sk_{i,j}^X \cdot P_{pub}^X)$ and $sk_{i,j}^X = H_1(r_j \cdot P_{pub}) + msk^X$, where $r_j \in \mathbb{Z}_q^*$. To obtain the real identity RID_i^X of SD_i^X, the pseudonym $PID_{i,j}^X$ and secret key $sk_{i,j}^X$ of SD_i^X must be obtained. On the one hand, the $sk_{i,j}^X$ is not transmitted on the network; on the other hand, according to the ECDLP, network attackers and message receivers cannot obtain the secret key $sk_{i,j}^X$ through the public key $PK_{i,j}^X$. Therefore, except for the pseudonym owner and TA^X, no other entity can obtain the real identity based on public information.

4) Un-linkability: In our proposed scheme, the smart device SD_i^X uses a pseudonym $PID_{i,j}^X$ to generate a signature. Note that generating $PID_{i,j}^X$ needs a random number r_j, and each random number is unique and un-linkability. So no adversary can link the two different signatures generated by two different pseudonyms from the same smart device. Therefore, our proposed scheme supports unlinkability.

5) Traceability: TA^X can extract the pseudonym $PID_{i,j}^X$ from the message sent by a smart device SD_i^X, and it can calculate the SD_i^X's real identity RID_i^X through the $PID_{i,j}^X$. If a smart device is found to send an illegal message, the IIoT system will feed the illegal message back to TA^X. TA^X first queries the public key $PK_{i,j}^X$ corresponding to $PID_{i,j}^X$, and then calculates $RID_i^X = H_2(msk^X \cdot PK_{i,j}^X) \oplus PID_{i,j}^X$ to obtain the real identity. Therefore, in our proposed scheme, TA can trace the source of any illegal message.

6) Identity revocation: In our proposed scheme, TA can cooperate with the blockchain to revoke illegal identity. If a smart device SD_i^X is found to publish an illegal message, TA^X first calculates the real identity RID_i^X corresponding to the illegal message and then stops generating the pseudonym $PID_{i,j}^X$. Subsequently, TA packs all the current pseudonyms into $PIDS_i^X$ and forwards the delete request for $PIDS_i^X$ to the blockchain. Finally, the blockchain can delete the corresponding information according to the delete request. If the illegal smart device regenerates a signature, ES can't find valid $PID_{i,j}^X$ in the blockchain, so the signature will be discarded. Therefore, our proposed scheme supports identity revocation.

7) Resistance to replay attack: In the proposed scheme, the message receiver should check whether the message has expired before verification. Assuming that T_1 represents the timestamp when the message is sent, T_2 represents the timestamp when the message is received, and ΔT represents the maximum delay time of the message. If $T_2 - T_1 \leq \Delta T$, the receiver will further authenticate the message; otherwise, the message will be considered expired, and the receiver will directly discard the message. Therefore, our proposed scheme can withstand replay attack.

8) Resistance to modification attack: According to the **Theorem 2**, we know that if an attacker modifies any parameter in $\{c, W, PID_{i,j}^A, PK_{i,j}^A, T_i\}$, the message receiver can determine that some parameters were modified by verifying that $PK_{i,j}^{A'} = \sigma^{'} \cdot (H_5(m^{'}) \cdot W^{'} + (h + h_3^{'}) \cdot U^{'})$ does not hold. Therefore, the proposed scheme can withstand the modification attack.

9) Resistance to impersonation attack: Suppose an attacker wants to impersonate a legitimate smart device. In that case, the attacker should have the ability to construct legitimate parameters $\{c, W, PID_{i,j}^A, PK_{i,j}^A, T_i\}$ and needs to ensure that the formula $PK_{i,j}^{A'} = \sigma^{'} \cdot (H_5(m^{'}) \cdot W^{'} + (h + h_3^{'}) \cdot U^{'})$ holds. According to the **Theorem 2**, it is impossible for an attacker to generate the above legitimate parameters. Therefore, the proposed scheme can withstand the impersonation attack.

C. Security Comparisons

We compare the security of our proposed scheme with three recently proposed authentication schemes for IIoT. In addition, we use SR-1, SR-2, SR-3, SR-4, SR-5, SR-6, SR-7, SR-8, and SR-9 to denote message confidentiality, message integrity and authentication, message anonymity, un-linkability, traceability, identity revocation, resistance to replay attack, resistance to modification attack, and resistance to impersonation attack, respectively. The results of the security comparison are shown in Table II.

It is worth noting that in [19], the confidentiality of the message is satisfied only after the message sender encrypts the message using the negotiated key. Still, during the subsequent interaction, the message sender uses this key all the time, so the un-linkability of the message is not satisfied. In [19] and [34], only the anonymity of the message senders is guaranteed, while in [28] and our proposed scheme, the anonymity of both the message senders and the message receivers are guaranteed. Because [28] and our proposed scheme are designed for a multi-receiver scenario, so the sender does not know which specific device the receiver is. However, [19] and [34] are not suited for a multi-receiver cross-domain IIoT scenario, so the sender is supposed to know exactly which device the receiver is and send the message directed.

TABLE II
SECURITY COMPARISON OF FOUR SCHEMES

	Cui *et al.* [28]	Shen *et al.* [19]	Yang *et al.* [34]	Our proposed
SR-1	✓	✓	×	✓
SR-2	✓	✓	✓	✓
SR-3	✓	✓	✓	✓
SR-4	✓	×	✓	✓
SR-5	✓	✓	✓	✓
SR-6	×	✓	✓	✓
SR-7	×	×	✓	✓
SR-8	✓	✓	✓	✓
SR-9	✓	✓	✓	✓

✓: The requirement is satisfied.
×: The requirement is not satisfied.

VII. PERFORMANCE EVALUATION

For a cross-domain IIoT scenario, the computing power of smart devices is limited, but the scenario requires a high level of real-time data. To demonstrate the feasibility of our proposed scheme, we evaluate the performance in terms of three aspects: computation overhead, communication overhead, and query latency.

A. Experimental Settings

1) Comparison Schemes Setting: To demonstrate the effectiveness of our proposed scheme, we compare it with three other schemes. To make the comparison more fair and convenient, in all schemes, we set up that all smart devices from different domains cannot communicate with each other directly and must communicate with each other through the edge server. Other corresponding changes are described in detail as follows.

- In Cui *et al.*'s scheme [28], SD_i^A signs the initial message, ES^B re-encrypts the message, and SD_k^B authenticates the received message.
- In Shen *et al.*'s scheme [19], SD_i^A and SD_k^B need to negotiate an encryption key before message authentication, and then SD_i^A sends the encrypted message to SD_k^B. So it takes three rounds of interaction between SD_i^A and SD_k^B to complete a message authentication. During these three rounds of interactions, the generation of the initial message and key negotiation is done by smart devices. The message signing, verification, and on-chain querying are all done by edge servers. Note that the encryption algorithm used in our experiments is Advanced Encryption Standard (AES).
- In Yang *et al.*'s scheme [34], SD_i^A signs the initial message, ES^B queries the necessary information from the blockchain, and SD_k^B authenticates the received message.

2) Experimental environment settings: To evaluate the performance of the proposed scheme in a real scenario, we use C++ to implement these schemes. In the experiment, the cryptographic tool library we use is Miracl Core [41]. And we choose the BLS12381 type curve, which provides 128-bit security level. In addition, all the common secure hash functions we use in this experiment first convert the input parameters to binary, then hash them using the SHA256 function and finally convert them to the required data type.

In our experiments, we use a PC running Ubuntu 18.04.3 operating system to simulate ES; this PC is equipped with an Intel Core i5-7500 CPU @3.4GHz and 16GB of memory. Since the computing power of SD is limited, we use a Raspberry Pi 4 to simulate the SD. This Raspberry Pi 4 runs Debian GNU/Linux 11 operating system and is equipped with a 1.5GHz CPU and 4GB of memory.

To facilitate testing the time spent on the corresponding operations in the blockchain, we deploy a blockchain platform. This platform uses the hyperledger fabric [42] and is deployed on nine PCs with the same performance. All nine PCs are running Ubuntu 18.04.3 operating system and have Intel Core i7-11700 CPU @2.50GHz and 16GB of memory. Among these nine PCs, one PC acts as an orderer node, and each of the other PCs acts as a BCDA in a certain administrative domain, respectively. We built different channels using different numbers of BCDAs to simulate consortium blockchains built by different numbers of domains. In addition, we use a PC equipped with an Intel Core i5-7500 CPU @3.4GHz and 16GB of memory as the client node. The on-chain operations we implement using the Go code.

It is worth noting that we assume that a cross-domain IIoT service has n receivers and the experimental results we show are the average of 200 experiments.

B. Computation Overhead

To compare the computational overhead more comprehensively, we first perform a theoretical analysis of the computational overhead and then compare the computational overhead by analyzing experimental simulation results.

1) Theoretical Analysis: To make the theoretical analysis more convenient, we define some notations for some time-consuming operations as follows. Note that according to the experimental setup, a message has n receivers. In addition, "-" in Table III indicates no time-consuming cryptographic operations.

- B: a bilinear pairing operation $e(g_1, g_2)$, where $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$.
- S_1: a scalar point multiplication operation $a \cdot P_1$, where $a \in \mathbb{Z}_q^*$ and $P_1 \in \mathbb{G}_1$.
- S_2: a scalar point multiplication operation $a \cdot P_2$, where $a \in \mathbb{Z}_q^*$ and $P_2 \in \mathbb{G}_2$.
- E_1: an exponential operation P_1^r, where $r \in \mathbb{Z}_q^*$ and $P_1 \in \mathbb{G}_1$.
- E_2: an exponential operation P_3^r, where $r \in \mathbb{Z}_q^*$ and $P_3 \in \mathbb{G}_T$.
- H: a hash-to-point operation $H(m) \in \mathbb{G}_1$, where $H(\cdot)$ is a secure hash function and $m \in \{0, 1\}^*$.

In each scheme, specific information on the time-consuming cryptographic operations on each entity is shown in Table III.

When there is no pre-processing in the authentication process, we can see that in scheme [28], most of the time-consuming cryptographic operations are performed by SD_i^A and SD_k^B, and the time-consuming cryptographic operations

TABLE III
TIME-CONSUMING CRYPTOGRAPHIC OPERATIONS IN THE FOUR SCHEMES

Scheme	Without Pre-processing				With Pre-processing			
	SD_i^A	ES^A	ES^B	SD_k^B	SD_i^A	ES^A	ES^B	SD_k^B
Cui *et al.* [28]	$10E_1 + 3E_2 + 3B + H$	-	E_1	$9E_1 + 4E_2 + 5B$	-	-	E_1	$9E_1 + 4E_2 + 2B$
Shen *et al.* [19]	$2nS_1$	$nS_1 + 2nE_2 + nS_2 + 3nB$	$nS_1 + 2nE_2 + nS_2 + 3nB$	$2S_1$	nS_1	$nS_1 + nS_2 + nE_2 + nB$	$nS_1 + nS_2 + nE_2 + nB$	S_1
Yang *et al.* [34]	$3nS_1$	-	-	$3S_1$	-	-	-	$3S_1$
Our proposed	$4S_1$	-	-	$4S_1$	-	-	-	$4S_1$

performed by SD_i^A and SD_k^B are $10E_1 + 3E_2 + 3B + H$ and $9E_1 + 4E_2 + 5B$ respectively. In scheme [19], we can see that the time-consuming cryptographic operations that need to be performed by entities other than the SD_k^B increase with the number of receivers. In scheme [34], the time-consuming cryptographic operation performed by the SD_i^A increases with the number of receivers, which is $3nS_1$. And the time-consuming cryptographic operation performed by the SD_k^B is $3S_1$. In our proposed scheme, the time-consuming cryptographic operations performed by the SD_i^A and SD_k^B are fixed, both being $4S_1$. That is, in our proposed scheme, the time-consuming cryptographic operations performed by SD_i^A and SD_k^B do not vary with the number of receivers.

Many operations can be pre-processed offline before the message is generated. As shown in Table III, with pre-processing, the time-consuming operations of the various schemes are effectively reduced. For example, in scheme [28], the time-consuming cryptographic operations performed by SD_i^A and SD_k^B are reduced by $10E_1 + 3E_2 + 3B + H$ and $(9E_1 + 4E_2 + 5B) - (9E_1 + 4E_2 + 2B) = 3B$, respectively. In scheme [19], the time-consuming cryptographic operations to be performed by SD_i^A, ES^A, ES^B and SD_k^B are reduced by nS_1, $(nE_2 + 2nB)$, $(nE_2 + 2nB)$ and S_1 respectively. Similarly, we find that in scheme [34], the time-consuming cryptographic operations to be performed by SD_i^A are reduced by $3nS_1$. In our proposed scheme, the time-consuming operations performed by SD_i^A are reduced by $4S_1$. It is worth noting that the time-consuming operation required for SD_k^B in our proposed scheme is $4S_1 - 3S_1 = 1S_1$ more than that required for SD_k^B in [34], because our proposed scheme guarantees data confidentiality while [34] does not, and the $1S_1$ is required for decryption operation in our proposed scheme. Data confidentiality is achieved using fewer cryptographic operations, which we consider to be an effective trade-off between security and efficiency.

2) Simulation Experimental Results: To illustrate that our proposed scheme is lightweight more concretely, we implement each scheme with and without pre-processing according to the experimental setting. When the number of receivers is 1, the experimental results are shown in Table IV and Fig. 4. Note that all cryptographic operations (e.g., integer addition and multiplication) are included in the simulation experiments. Therefore, in Table IV, when schemes support pre-processing, computational overhead exists for the SD_i^A in the scheme [34] and our proposed scheme. In addition, Fig. 5 compares the total time required for authentication in each scheme as the number of message receivers increases. The "-" in Table IV indicates no computational overhead.

Combining Table IV and Fig. 4, we can see that when the number of receivers is 1, the total time cost to complete an authentication process in [28] without pre-processing is $47.746 + 0.278 + 63.971 = 111.995$ ms, which is the highest among the four schemes. The reason is that resource-constrained smart devices perform many time-consuming cryptographic operations. Similarly, the scheme [19] contains many time-consuming cryptographic operations, and the total time required to complete an authentication is $2.523 + 10.394 + 10.387 + 2.496 = 25.800$ ms. Fig. 4 shows that the total time required to complete an authentication between [34] and our proposed scheme is very low, where [34] takes $3.900 + 3.552 = 7.452$ ms, which is the lowest time cost among the four schemes. In our proposed scheme, the time required is $(4.790 + 4.597) - 7.452 = 1.935$ ms more than that needed by [34], but this gap will be effectively reduced with pre-processing.

From Table IV, we find that the time cost of our scheme with pre-processing is saved about $((4.790+4.597)-(0.064+4.516))/(4.790+4.597) \approx 51.2\%$ relative to the case without pre-processing. We can see that with pre-processing, the time spent in our proposed scheme is about $(0.064 + 4.516)/(0.029 + 0.277 + 42.654) \approx 10.7\%$ of [28] and $(0.064 + 4.516)/(1.266 + 4.664 + 4.660 + 1.263) \approx 38.6\%$ of [19]. In addition, with pre-processing, the time cost of our proposed scheme is $(0.064+4.516)-(0.035+3.235) = 1.310$ ms more than [34]. The reason is that our proposed scheme achieves data confidentiality and requires a decryption operation while [34] does not.

In Fig. 5, we can see that the time spent in [19] grows fastest

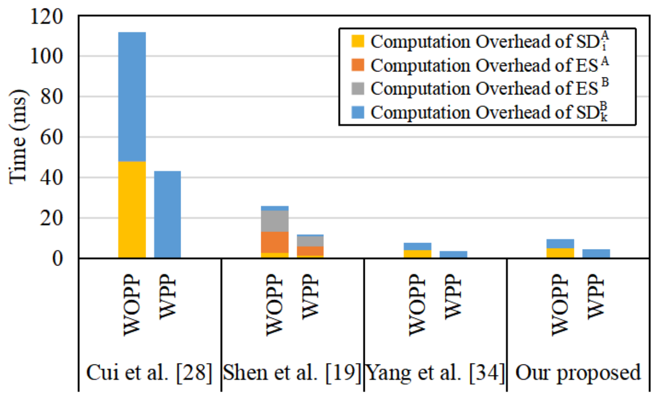


Fig. 4. Comparison of computation overhead when the number of message receivers is 1 (WPP: with pre-processing, WOPP: without pre-processing).

as the number of message receivers increases. The time spent in [28] and our proposed scheme does not increase with the number of message receivers, which is because both schemes are designed for multi-receiver IIoT scenarios. In addition, we find that the time consumption of [34] and our proposed scheme have been at a low level. However, when the number of message receivers exceeds 38, the time to be consumed by [34] will exceed that of our proposed scheme. Therefore, the computational overhead of our proposed scheme is better than other related schemes when the number of receivers exceeds 38.

In summary, our proposed scheme is lightweight and suitable for multi-receiver cross-domain IIoT.

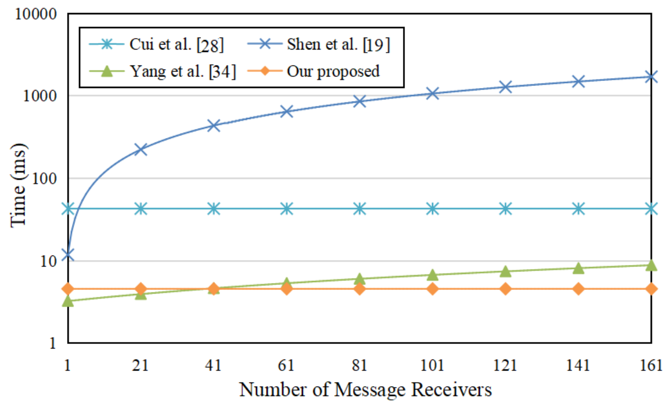


Fig. 5. Authentication time consumption for different number of message receivers (with pre-processing).

C. Communication Overhead

According to the experimental setting, to compare the communication overhead of each scheme, we first record the interactions between devices throughout the process from signature generation to signature authentication, then compute the size of communication packets. In the cryptographic tool library we use, the occupied space size for elements in $\mathbb{Z}_q^*$ is 48 bytes, and the occupied space size for elements in $\mathbb{G}_1$ is 97 bytes. We set the size of the message is 22 bytes and set the size of the timestamp is 16 bytes. In addition, we let I_{SD-ES}, I_{ES-ES}, I_{ES-BC} denote the interaction between the smart device and the edge server, the interaction between the edge server and the edge server, the interaction between the edge server and the blockchain respectively.

In our proposed scheme, the smart device SD_i^A sends data $\delta = (c, W, PID_{i,j}^A, T_i)$ to ES^A, where the size of this data is $(22 + 48 + 97 + 97 + 32 + 16) = 312$ bytes. The ES^A then forwards the received data to the ES^B, which is 312 bytes. Once the ES^B receives the data, it will send $PID_{i,j}^A$ of size 32 bytes to $BCDA^B$ and then get the $PK_{i,j}^A$ corresponding to the $PID_{i,j}^A$. Finally, the ES^B sends data $(c, W, PID_{i,j}^A, T_i, PK_{i,j}^A)$ to SD_k^B, where the size of this data is $(312 + 97) = 409$ bytes. In addition, there are n receivers for an IIoT service. Therefore, in our proposed scheme, the total communication packet size is $(312+312+32+97)+409n = 753+409n$ bytes and the total interaction is $(n+1)I_{SD-ES}$+I_{ES-ES}+$2I_{ES-BC}$.

We compute the communication overhead of other schemes using the same method. In Cui *et al.*'s scheme [28], the total communication packet size is about $1444 + 698n$ bytes, and the total interaction is $(n+1)I_{SD-ES}$+I_{ES-ES}. In Shen *et al.*'s scheme [19], the total communication packet size is about $1318n$ bytes, and the total interaction is $6nI_{SD-ES}$+$3nI_{ES-ES}$+$4nI_{ES-BC}$. In Yang *et al.*'s scheme [34], the total communication packet size is about $1816n$ bytes, and the total interaction is $2nI_{SD-ES}$+nI_{ES-ES}+$2nI_{ES-BC}$. First, we find that [28] has the lowest interactions since the authentication process of this scheme does not involve the blockchain. Second, we find that the total interactions of [34] and [19] are more than the total interactions of our proposed scheme because our proposed scheme leverages service-based ideas and is suitable for the multi-receiver scenario. In our proposed scheme, completing an authentication process only needs one signature, and the edge server only needs to submit one query request to the blockchain. In contrast, the other two related schemes require n signatures and more interactions between entities during the authentication processing (e.g., edge servers need to be queried n times in the blockchain). In addition, the comparison of the total communication packet size of the four schemes when the message receiver is 1 is shown in Fig. 6.

From Fig. 6, we can see that our proposed scheme has the smallest total communication packet size among the four schemes. Specifically, the total communication packet size of our proposed scheme is approximately 980 bytes less than that of [28], approximately 156 bytes less than that of [19], and approximately 654 bytes less than that of [34]. In addition, as the number of message receivers increases, the communication overhead advantage of our proposed scheme will become more significant. There are three reasons for this situation. The first reason is that the data transferred per interaction between the entities of our proposed scheme is short, and the second reason is that the total number of interactions between the entities of our proposed scheme is low. The last reason is that our proposed scheme is designed for the multi-receiver cross-domain IIoT scenario. In summary, our proposed scheme has a low communication overhead and is suitable for IIoT

TABLE IV
SIMULATION EXPERIMENTAL RESULTS OF THE FOUR SCHEMES (MS)

Scheme	Without Pre-processing				With Pre-processing			
	SD_i^A	ES^A	ES^B	SD_k^B	SD_i^A	ES^A	ES^B	SD_k^B
Cui *et al.* [28]	47.746	-	0.278	63.971	0.029	-	0.277	42.654
Shen *et al.* [19]	2.523	10.394	10.387	2.496	1.266	4.664	4.660	1.263
Yang *et al.* [34]	3.900	-	-	3.552	0.035	-	-	3.235
Our proposed	4.790	-	-	4.597	0.064	-	-	4.516

environments with high real-time data requirements.

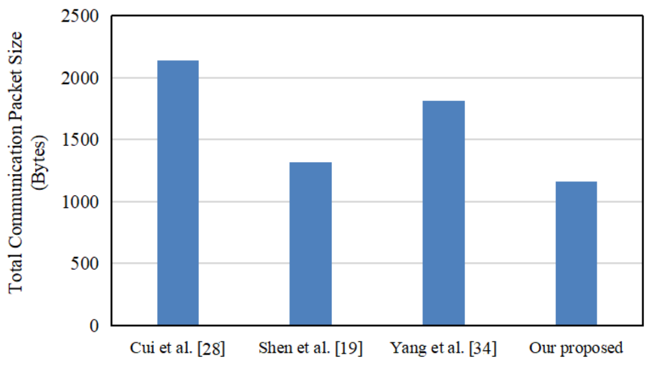


Fig. 6. Comparison of total communication packet size when the number of message receivers is 1.

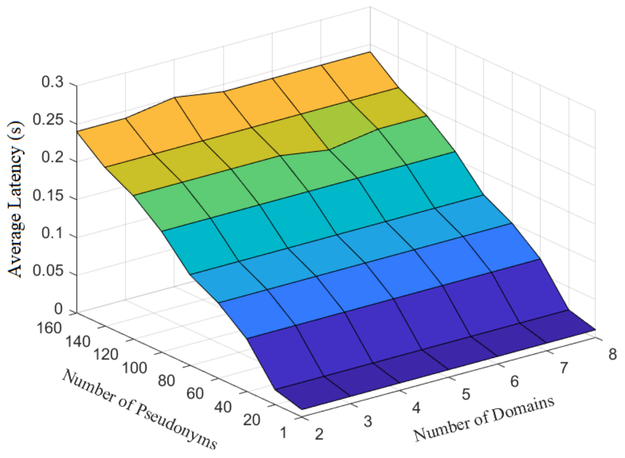


Fig. 7. Query latency.

D. Query Latency

In the process of message authentication, [28], [34], and our proposed scheme need to interact with the blockchain, where these interactions are mainly to perform query operations. Specifically, the ES submits a pseudonym to the blockchain; the blockchain performs the query operation and returns the information corresponding to that pseudonym. Therefore, we evaluate the on-chain query latency based on the blockchain platform settings mentioned in subsection VII-A. The results of the experiment are shown in Fig. 7.

From Fig. 7, we can see that when the number of pseudonyms in a batch query is constant, the average latency does not change significantly with the number of domains. For example, regardless of the number of administrative domains, the average latency is about 0.20 s when the number of pseudonyms for a batch query is 20. This is because the query operation is mainly performed in the local ledger. Therefore, the number of domains does not affect the query latency. In addition, we can see that the average query latency rises when the number of pseudonyms in a batch query increases. For example, when the number of pseudonyms for a batch query is 1, the average latency is about 0.01 s. And when the number of pseudonyms in the batch query is 160, the average latency is about 0.24 s.

Insight: To meet the IIoT requirements for real-time data, we should reduce the number of interactions with the blockchain. By analyzing the communication overhead, we find that for an IIoT service, our proposed scheme queries the blockchain only once during the authentication process, regardless of the number of receivers. However, in [28] and [34], the number of queries to the blockchain increases with the number of receivers. Therefore, our proposed scheme is applicable to multi-receiver cross-domain IIoT scenarios.

VIII. CONCLUSION

In this paper, we propose a blockchain-based lightweight message authentication scheme for multi-receiver cross-domain IIoT. The main goal of this scheme is to satisfy the security and efficiency requirements in cross-domain IIoT. Specifically, we first design a lightweight edge-assisted cross-domain authentication framework using blockchain, and then design a lightweight message authentication algorithm. Detailed security proofs and analysis demonstrate that the proposed scheme can resist various attacks. In addition, a comparison with related schemes shows that our proposed scheme is lightweight in terms of computational and communication overheads and suitable for multi-receiver cross-domain IIoT scenarios. In the future, we will design secure and efficient message authentication schemes for mobile smart devices in IIoT.

References

[1] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[2] J. Cui, L. Wei, H. Zhong, J. Zhang, Y. Xu, and L. Liu, "Edge computing in vanets-an efficient and privacy-preserving cooperative downloading scheme," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1191–1204, 2020.

[3] Y. Liao, E. de Freitas Rocha Loures, and F. Deschamps, "Industrial internet of things: A systematic literature review and insights," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4515–4525, 2018.

[4] M. Serror, S. Hack, M. Henze, M. Schuba, and K. Wehrle, "Challenges and opportunities in securing the industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 5, pp. 2985–2996, 2021.

[5] J. Cui, J. Lu, H. Zhong, Q. Zhang, C. Gu, and L. Liu, "Parallel key-insulated multi-user searchable encryption for industrial internet of things," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2021.

[6] W. Sun, J. Liu, and Y. Yue, "Ai-enhanced offloading in edge computing: When machine learning meets industrial iot," *IEEE Network*, vol. 33, no. 5, pp. 68–74, 2019.

[7] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, and D. O. Wu, "Edge computing in industrial internet of things: Architecture, advances and challenges," *IEEE Communications Surveys Tutorials*, vol. 22, no. 4, pp. 2462–2488, 2020.

[8] H. Zhang, X. Chen, X. Lan, H. Jin, and Q. Cao, "Btcas: A blockchain-based thoroughly cross-domain authentication scheme," *Journal of Information Security and Applications*, vol. 55, p. 102538, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S221421261931004X

[9] K. I.-K. Wang, X. Zhou, W. Liang, Z. Yan, and J. She, "Federated transfer learning based cross-domain prediction for smart manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 6, pp. 4088–4096, 2022.

[10] J. M. Mcginthy and A. J. Michaels, "Secure industrial internet of things critical infrastructure node design," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8021–8037, 2019.

[11] G. Li, J. Wu, J. Li, K. Wang, and T. Ye, "Service popularity-based smart resources partitioning for fog computing-enabled industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4702–4711, 2018.

[12] H. Xiong, Q. Mei, and Y. Zhao, "Efficient and provably secure certificateless parallel key-insulated signature without pairing for iiot environments," *IEEE Systems Journal*, vol. 14, no. 1, pp. 310–320, 2020.

[13] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, "Security and privacy challenges in industrial internet of things," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2015, pp. 1–6.

[14] B. Cao, Y. Li, L. Zhang, L. Zhang, S. Mumtaz, Z. Zhou, and M. Peng, "When internet of things meets blockchain: Challenges in distributed consensus," *IEEE Network*, vol. 33, no. 6, pp. 133–139, 2019.

[15] T. Meng, Y. Zhao, K. Wolter, and C.-Z. Xu, "On consortium blockchain consistency: A queueing network model approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 6, pp. 1369–1382, 2021.

[16] J. Cui, F. Ouyang, Z. Ying, L. Wei, and H. Zhong, "Secure and efficient data sharing among vehicles based on consortium blockchain," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2021.

[17] L. Xue, H. Huang, F. Xiao, and W. Wang, "A cross-domain authentication scheme based on cooperative blockchains functioning with revocation for medical consortiums," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022.

[18] X. Jiang, F. R. Yu, T. Song, Z. Ma, Y. Song, and D. Zhu, "Blockchain-enabled cross-domain object detection for autonomous driving: A model sharing approach," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 3681–3692, 2020.

[19] M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, and M. Guizani, "Blockchain-assisted secure device authentication for cross-domain industrial iot," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 942–954, 2020.

[20] S. Viswanathan, R. Tan, and D. K. Y. Yau, "Exploiting power grid for accurate and secure clock synchronization in industrial iot," in *2016 IEEE Real-Time Systems Symposium (RTSS)*, 2016, pp. 146–156.

[21] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4724–4734, 2018.

[22] X. Zhang, C. Xu, H. Wang, Y. Zhang, and S. Wang, "Fs-peks: Lattice-based forward secure public-key encryption with keyword search for cloud-assisted industrial internet of things," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, pp. 1019–1032, 2021.

[23] F. Tong, X. Chen, K. Wang, and Y. Zhang, "Ccap: A complete cross-domain authentication based on blockchain for internet of things," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 3789–3800, 2022.

[24] Q. Zhang, J. Wu, H. Zhong, D. He, and J. Cui, "Efficient anonymous authentication based on physically unclonable function in industrial internet of things," *IEEE Transactions on Information Forensics and Security*, pp. 1–1, 2022.

[25] A. Esfahani, G. Mantas, R. Matischek, F. B. Saghezchi, J. Rodriguez, A. Bicaku, S. Maksuti, M. G. Tauber, C. Schmittner, and J. Bastos, "A lightweight authentication mechanism for m2m communications in industrial iot environment," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 288–296, 2019.

[26] G. K. Verma, B. Singh, N. Kumar, M. S. Obaidat, D. He, and H. Singh, "An efficient and provable certificate-based proxy signature scheme for iiot environment," *Information Sciences*, vol. 518, pp. 142–156, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025520300074

[27] C. Esposito, A. Castiglione, F. Palmieri, and A. D. Santis, "Integrity for an event notification within the industrial internet of things by using group signatures," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3669–3678, 2018.

[28] J. Cui, F. Wang, Q. Zhang, Y. Xu, and H. Zhong, "Anonymous message authentication scheme for semitrusted edge-enabled iiot," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 12, pp. 12 921–12 929, 2021.

[29] J. Guan, Y. Wu, S. Yao, T. Zhang, X. Su, and C. Li, "Bsla: Blockchain-assisted secure and lightweight authentication for sgin," *Computer Communications*, vol. 176, pp. 46–55, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366421001997

[30] J. Wang, L. Wu, K.-K. R. Choo, and D. He, "Blockchain-based anonymous authentication with key management for smart grid edge computing infrastructure," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1984–1992, 2020.

[31] S. Guo, F. Wang, N. Zhang, F. Qi, and X. Qiu, "Master-slave chain based trusted cross-domain authentication mechanism in iot," *Journal of Network and Computer Applications*, vol. 172, p. 102812, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804520302836

[32] C. Feng, B. Liu, Z. Guo, K. Yu, Z. Qin, and K.-K. R. Choo, "Blockchain-based cross-domain authentication for intelligent 5g-enabled internet of drones," *IEEE Internet of Things Journal*, pp. 1–1, 2021.

[33] L. Wang, Y. Tian, and D. Zhang, "Toward cross-domain dynamic accumulator authentication based on blockchain in internet of things," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 4, pp. 2858–2867, 2022.

[34] Y. Yang, L. Wei, J. Wu, C. Long, and B. Li, "A blockchain-based multi-domain authentication scheme for conditional privacy preserving in vehicular ad-hoc network," *IEEE Internet of Things Journal*, pp. 1–1, 2021.

[35] H. Xiong, Y. Wu, C. Su, and K. hui Yeh, "A secure and efficient certificateless batch verification scheme with invalid signature identification for the internet of things," *Journal of Information Security and Applications*, vol. 53, p. 102507, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214212619307999

[36] Y. Yu, Y. Li, J. Tian, and J. Liu, "Blockchain-based solutions to security and privacy issues in the internet of things," *IEEE Wireless Communications*, vol. 25, no. 6, pp. 12–18, 2018.

[37] B. Oki, M. Pfluegl, A. Siegel, and D. Skeen, "The information bus: An architecture for extensible distributed systems," *SIGOPS Oper. Syst. Rev.*, vol. 27, no. 5, p. 58–68, dec 1993. [Online]. Available: https://doi.org/10.1145/173668.168624

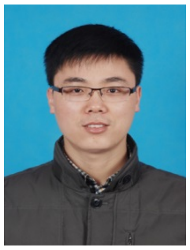
[38] S. Qian, W. Mao, J. Cao, F. L. Mouël, and M. Li, "Adjusting matching algorithm to adapt to workload fluctuations in content-based publish/subscribe systems," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 1936–1944.

[39] W. Zhang, D. Yang, W. Wu, H. Peng, N. Zhang, H. Zhang, and X. Shen, "Optimizing federated learning in distributed industrial iot: A multi-agent approach," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3688–3703, 2021.

[40] J. Lohmer and R. Lasch, "Production planning and scheduling in multi-factory production networks: a systematic literature review," *International Journal of Production Research*, vol. 59, no. 7, pp. 2028–

2054, 2021. [Online]. Available: https://doi.org/10.1080/00207543.2020.1797207
[41] "Miracl core," https://github.com/miracl/core.
[42] "hyperledger," https://github.com/hyperledger/fabric.

Fengqun Wang is currently a PhD student in the School of Computer Science and Technology, Anhui University, Hefei, China. His research interests include IoT security, blockchain and applied cryptography.



Jie Cui (Senior Member, IEEE) was born in Henan Province, China, in 1980. He received his Ph.D. degree in University of Science and Technology of China in 2012. He is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. His current research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). He has over 150 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Journal on Selected Areas in Communications, IEEE Transactions on Mobile Computing, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Network and Service Management, IEEE Transactions on Industrial Informatics, IEEE Transactions on Industrial Electronics, IEEE Transactions on Cloud Computing and IEEE Transactions on Multimedia), academic books and international conferences.

Qingyang Zhang was born in Anhui Province, China, in 1992. He received his B. Eng. degree and Ph.D. degree in computer science from Anhui University in 2021. He is currently a lecture of School of Computer Science and Technology at Anhui University. His research interest includes edge computing, computer systems, and security.

Debiao He received his Ph.D. degree in applied mathematics from School of Mathematics and Statistics, Wuhan University, Wuhan, China in 2009. He is currently a professor of the School of Cyber Science and Engineering, Wuhan University, Wuhan, China. His main research interests include cryptography and information security, in particular, cryptographic protocols. He has published over 100 research papers in refereed international journals and conferences, such as IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Security and Forensic, and Usenix Security Symposium. He is the recipient of the 2018 IEEE Sysems Journal Best Paper Award and the 2019 IET Information Security Best Paper Award. His work has been cited more than 10000 times at Google Scholar. He is in the Editorial Board of several international journals, such as Journal of Information Security and Applications, Frontiers of Computer Science, and Human-centric Computing & Information Sciences.

Chengjie Gu received his Ph.D. degree in Nanjing University of Posts and Telecommunications in 2012. From 2012 to 2017, he was an innovation team leader in the 38th Research Institute of CETC and conducted research and development in the communication and networking sector. Currently he is a president of security research institute in new H3C group. He is also studying for postdoctoral fellowship at the USTC. He is a high-level innovation leader of Anhui province and a cybersecurity expert of Zhejiang province in China. His research interest includes network security and trusted network architecture, etc.

Hong Zhong was born in Anhui Province, China, in 1965. She received her PhD degree in computer science from University of Science and Technology of China in 2005. She is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. Her research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). She has over 200 scientific publications in reputable journals (e.g. IEEE Journal on Selected Areas in Communications, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Mobile Computing, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Multimedia, IEEE Transactions on Vehicular Technology, IEEE Transactions on Network and Service Management, IEEE Transactions on Cloud Computing, IEEE Transactions on Industrial Informatics, IEEE Transactions on Industrial Electronics and IEEE Transactions on Big Data), academic books and international conferences.