Efficient Batch Authentication Scheme Based on Edge Computing in IIoT

Jie Cui, *Senior Member, IEEE*, Fengqun Wang, Qingyang Zhang, Chengjie Gu, and Hong Zhong

***Abstract*—In the industrial Internet of Things (IIoT) environment (e.g., a smart factory), smart devices with limited computing power can bring large amounts of privacy-sensitive data into insecure networks when they interact. If a network attacker intercepts and tampers with this data, it may cause chaos in production and even paralyze the entire IIoT system. Therefore, to ensure the regular operation of intelligent production, data receivers must authenticate the data before using them. However, existing message authentication schemes in the IIoT environment authenticate each message individually, which creates many redundant operations. Hence, to ensure data security among smart devices and reduce the computational overhead of data processing, we propose a batch authentication scheme based on edge computing in IIoT. Specifically, we design a lightweight batch authentication algorithm and use edge servers to assist smart devices in authenticating data, thus reducing the computational burden on smart devices and improving the efficiency of message authentication. The security analysis shows that the proposed scheme is secure in the random oracle model and meets the series of security requirements of the IIoT. In addition, we illustrate the efficiency of the scheme through experiments.**

***Index Terms*—Industrial Internet of Things (IIoT), batch authentication, edge computing, elliptic curve cryptography (ECC), hash chain.**

I. INTRODUCTION

IN RECENT years, the Internet of Things (IoT) [1], [2], [3] has gained a significant amount of attention in the industry because it provides a new way for people to communicate with things, making it possible for industrial production to achieve high yields with fewer risks [4]. The IoT terminology related to industrial processes and industrial infrastructure is referred to as industrial IoT (IIoT) [5], [6], [7].

Manuscript received 16 October 2021; revised 11 February 2022, 8 July 2022, and 8 September 2022; accepted 8 September 2022. Date of publication 13 September 2022; date of current version 7 March 2023. The work was supported in part by the National Natural Science Foundation of China under Grant 62272002 and Grant U1936220, in part by the Excellent Youth Foundation of Anhui Scientific Committee under Grant 2108085J31, and in part by the Special Fund for Key Program of Science and Technology of Anhui Province, China under Grant 202003A05020043. The associate editor coordinating the review of this article and approving it for publication was X. Fu. *(Corresponding author: Hong Zhong.)*

Jie Cui, Fengqun Wang, Qingyang Zhang, and Hong Zhong are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, the Institute of Physical Science and Information Technology, and the Anhui Engineering Laboratory of IoT Security Technologies, Anhui University, Hefei 230039, China (e-mail: zhongh@ahu.edu.cn).

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

Digital Object Identifier 10.1109/TNSM.2022.3206378

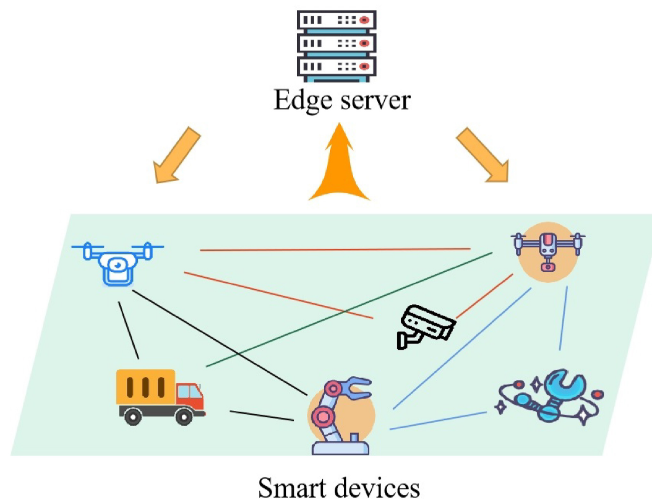


Fig. 1. IIoT system based on edge computing.

In an IIoT environment, many heterogeneous smart devices are deployed [8]. To flexibly allocate resources and intelligently optimize production methods, these smart devices need to share and process industrial data in real-time [9]. However, with the rapid development of IIoT and the increasing network scale, the number of smart devices and the amount of data generated are increasing dramatically [10], which imposes a heavy computational burden on resource-constrained smart devices [11]. In addition, the IIoT system brings a large amount of privacy-sensitive data into complex and insecure networks that are vulnerable to network attackers, resulting in data leakage or tampering. Therefore, ensuring the real-time and security of IIoT data becomes paramount [12], [13].

On the one hand, ensuring the real-time of data in the IIoT environment is crucial [14]. For example, video-based production line monitoring data [15] needs to be responded to on time; otherwise, it could lead to production lag and chaos. In a typical IIoT environment (e.g., a smart factory), smart devices directly process massive shared data [16], generating a large computational overhead that cannot meet the high real-time demand for the IIoT environment. Furthermore, if cloud computing [17], [18] with very strong computing power is used to assist smart devices in processing this shared data, additional data transmission overhead will be generated [19]. Therefore, some researchers have introduced edge computing [8], [20], which is closer to smart devices. Fig. 1 shows an IIoT system based on edge computing. In this system, smart devices with limited computing power and edge server with strong computing power are deployed. The edge server is responsible for collecting, filtering, analyzing, and forwarding the data. And data can be shared between smart devices with the same interest (e.g., production tasks). Although the edge server can assist smart devices in computing, its computational overhead is still very high in the face of massive amounts of shared data.

On the other hand, ensuring data security in the IIoT environment is vital. The reason is that if critical data are compromised, the network attacker can (a) control the smart devices in the IIoT system, (b) lead to chaos in production, (c) cause unnecessary economic loss, and (d) even cause safety accidents [21]. To ensure the security of the IIoT environment, some researchers have pointed out that the confidentiality and integrity of data need to be guaranteed first [22]. If data confidentiality is not guaranteed, network attackers can obtain sensitive data, leading to compromised industrial secrets. Suppose the integrity of the data is not guaranteed. In that case, network attackers can tamper with IIoT data without detection, which may lead to production chaos once smart devices use these tampered data. Second, the unlinkability [23] and anonymity [24] of data need to be guaranteed. If the unlinkability of data is not guaranteed, network attackers can infer some sensitive information from multiple data sent by the same smart device through techniques such as machine learning. If the anonymity of the data is not guaranteed, the real identity of the smart device can be exposed, and network attackers can launch targeted attacks on the smart devices. Therefore, some researchers have proposed solutions for IIoT message authentication [25], [26]. However, they can only authenticate received shared data one by one, which is only suitable for the IIoT environment with relatively low message density. For the IIoT environment with high message density and high data real-time requirements, such as emergency shutdown systems for smart devices, these schemes generate significant computational overhead that cannot be ignored.

A. Our Motivations

From the above analysis, we understand that it requires high security and real-time for the IIoT environment with high message density (e.g., inside a smart factory). However, this environment faces the following challenges: (1) malicious network attackers trying to obtain factory privacy-sensitive data and tamper with data, (2) the computing power of smart devices is limited, but the data that needs to be authenticated is massive, and (3) numerous redundant operations for authentication in existing IIoT schemes. Therefore, we are motivated to propose a security scheme that utilizes an edge server with high computing power to assist smart devices in message authentication and authenticate multiple messages in a batch.

B. Our Contribution

To solve the real-time and security issues in an IIoT system, such as in a smart factory, we propose an efficient batch authentication scheme in an IIoT environment based on edge computing.

The contributions of our proposed scheme are the following:

- First, we design an efficient batch authentication algorithm, which guarantees the confidentiality, integrity, unlinkability, and anonymity of the data. Moreover, we use edge servers to assist smart devices in message authentication, reducing the computational pressure on smart devices and improving authentication efficiency.
- Second, to reduce the cost of signing notification messages by the edge server, we design a lightweight signature algorithm based on the hash chain, which ensures data security and reduces the cost of edge servers signing notification messages.
- Finally, we demonstrate the security of our proposed scheme through security proof and analysis. In addition, we show the feasibility of applying our scheme in an IIoT system through experiments.

The remainder of this paper is organized as follows. Section II focuses on the existing work related to security in IIoT. Section III presents the system model and objectives of the proposed scheme. Section IV provides a detailed description of the proposed scheme. The security proof and analysis of the scheme are given in Section V. Section VI provides a detailed comparison and explanation in terms of the authentication performance through experimental data. Finally, Section VII presents the conclusions of the scope for future research.

II. Related Work

In this section, we introduce some message authentication schemes in IIoT and analyze them.

Due to the complexity of the network in the IIoT environment, the massiveness of data, and the limited computing power of smart devices, the data privacy issues faced by the IIoT environment are particularly prominent. To ensure the security of data, related researchers have proposed many solutions.

In 2018, Esposito *et al.* [27] adopted group signature technology to effectively ensure the confidentiality and integrity of data. On this basis, Cui *et al.* [19] introduced the proxy re-encryption technique to propose an authentication scheme that guarantees anonymity while guaranteeing data confidentiality and integrity. However, both of these schemes use bilinear pairing, which only applies to the IIoT scenarios with low data volume. In IIoT local area networks, due to the huge amount of data, the overhead is huge if the message authentication scheme is constructed using bilinear pairing, which may exhaust resources such as smart devices with limited computing power.

To address the above problem, some researchers used lightweight elliptic curve cryptography (ECC) in IIoT environments. For example, drone networks are often used in the IIoT, and Hussain *et al.* [28] found that some current schemes are not secure and inapplicable after analysis. To solve the existing problems, Hussain *et al.* applied ECC to the authentication scheme, allowing this scheme to meet security requirements while effectively improving authentication efficiency. In 2018, Li *et al.* [29] took into account the limited resources of smart device nodes in the IIoT environment and proposed a privacy-protected IIoT user authentication protocol based on ECC. This scheme greatly reduces the computation cost caused by verification. However, in the face of massive data in the IIoT environment, the above scheme can only verify the validity of one message at a time, which still consumes a large amount of computational overhead.

Although there are few relevant batch authentication schemes in the IIoT environment, batch authentication is already widely used in many areas of the IoT. For example, Xiong *et al.* [30] used ECC to design a lightweight authentication scheme that supports message receivers to authenticate the validity of multiple messages at a time. Still, in this scheme, smart devices directly perform batch authentication, putting much computational pressure on them. To achieve fast authentication of data uploaded by end devices without exposing the owner's sensitive data, Liu *et al.* [31] proposed an anonymous batch authentication scheme. This scheme can authenticate all end devices' information simultaneously and has confidentiality. In 2020, to protect data privacy when analyzing the smart grid users' data, Guo *et al.* [32] proposed a practical and lightweight aggregation scheme for the smart grid. In this scheme, to reduce the computational overhead of the system, the aggregation provider can perform batch authentication of the encrypted data. However, the computational overhead of the above batch authentication scheme is still relatively large. Faced with a huge amount of data and to reduce the cost of message authentication to guarantee real-time, Zhang *et al.* [33] proposed a batch authentication scheme for vehicular networks. Like many batch authentication schemes in vehicular networks, this research ensures the security of messages and reduces the overhead brought by message authentication. However, the above batch authentication schemes do not consider the time consumption caused by the edge server's signature of the notification message, nor do they provide specific signatures for the edge server.

III. System Model and Objectives

In this section, we introduce several aspects of the preliminaries, system model, assumptions, and design objectives to demonstrate the proposed scheme more clearly.

A. Preliminaries

In our proposed scheme, we use hash chains, elliptic curves. The following is a detailed introduction of these two technologies.

1) Hash Chain: The hash chain mainly uses the properties of the hash function. The specific operation is that the user chooses an initial data, then hashes the initial data several times, and finally connects the results obtained by each hash into a sequence, which is a hash chain. The hash chain's security relies on the hash function's one-way property.

A secure hash function $h(\cdot)$ should satisfy the following properties:

- $h(\cdot)$ inputs a message of arbitrary length, but outputs a fixed-length message.

$$seed \xrightarrow{h(\cdot)} S_1 \xrightarrow{h(\cdot)} S_2 \ldots\ldots S_{i-2} \xrightarrow{h(\cdot)} S_{i-1} \xrightarrow{h(\cdot)} S_i$$

Fig. 2. Hash chain.

- Given x as an input message to the $h(\cdot)$, it can obtain y easily, where $y = h(x)$. However, it is difficult to obtain x if given y.
- If $x' \neq x$, then $h(x') \neq h(x)$.

As is shown in Fig. 2, it is a hash chain of length i. And the *seed* is an initial seed value, which can be used to compute $S_i = h^i(seed)$. It's worth noting that, if given S_i, it is easy to obtain $S_{i+1} = h(S_i)$. However, it is very hard to obtain $S_{i-1} = h^{-1}(S_i)$.

In the proposed scheme, we use the properties of the hash chain to design a lightweight signature algorithm for edge servers to sign notification messages.

2) Elliptic Curve Cryptography: The elliptic curve cryptography (ECC) system is briefly summarized as follows:

Given a finite field F_q and a large prime number q greater than 3. And let an elliptic curve point E over F_q, which is expressed as $y^2 = x^3 + ax + b \pmod q$. Here, $a, b, x, y \in F_q$, and it should satisfy $4a^3 + 27b^2 \pmod q \neq 0$. Let O as a point at infinity, G_q as a cyclic group with the order q and P as a generator. The group G_q needs to have the following three properties:

- *Additive:* Suppose there are two points P and Q on the cyclic group G_q, and if these two points are not equal, R is obtained by computing $P + Q$. Here, R is the intersection of the line connecting P and Q with the elliptic curve. Also, if $P = -Q$, then $P + Q = 0$ is obtained.
- *Scalar point multiplication:* Suppose $P \in G_q$ and $n \in Z_q^*$, then we can get $n \cdot P = P + P + \cdots + P$.
- *Elliptic curve discrete logarithm problem (ECDLP):* ECC security is primarily based on ECDLP, which is said that given s and P, where the $s \in Z_q$ and $P \in G_q$, it is easy to compute $P_{pub} = s \cdot P$, where the $P_{pub} \in G_q$. However, given P and P_{pub}, it is hard to computer s.

In the proposed scheme, we use ECC to design a lightweight batch authentication algorithm to reduce the time overhead associated with the signature of smart devices.

B. System Model and Assumptions

As shown in Fig. 3, there are three entities in the IIoT system model: the key distribution center, the edge server, and some smart devices. Each type of entity and its assumptions are described in detail below.

1) *Key Distribution Center (KDC):* The KDC is a cluster of servers in the IIoT system. It assumes that the KDC is a fully trusted entity with strong storage and computational capabilities. The KDC can generate system parameters and is responsible for the distribution of keys between the edge server and smart devices. The KDC also can generate a seed and the corresponding parameters of the seed for the edge server to sign the notification messages. Finally, the KDC sends the system's public parameters to the IIoT

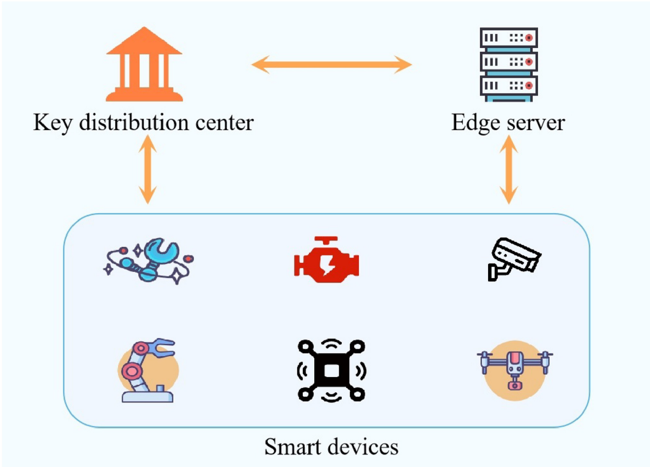


Fig. 3. System model.

system and some secret parameters to its corresponding entities through a secure channel. Note that KDC is the only entity that can trace the real identity of a smart device.

2) *Edge Server (ES):* The ES is a cluster of servers in the IIoT system. It is a server that belongs to an organization (e.g., a smart factory), responsible for assisting in the authentication of smart devices. The ES has good computational and storage capabilities. Note that the ES can communicate over a wider range than each smart device can communicate with each others. Therefore, if the ES can receive data sent from a smart device (data sender), all other smart devices (data receivers) that can receive data sent from that smart device (data sender) can also receive data sent from the ES.

3) *Smart Device (SD):* Smart devices are distributed in the IIoT system and have many interests. These smart devices usually have the poor computing power and limited storage capacity. They can generate shared data (e.g., production status) and dynamically adjust production methods by using data sent by other devices with the same interest. It is worth noting that to protect the IIoT system's privacy, smart devices in the IIoT should be anonymous [34].

C. Threat Model

The main adversaries considered in our proposed scheme are external network attackers and are not directly involved with the entities in the IIoT system. This type of adversary can launch both passive and active attacks. Specifically, when an adversary launches a passive attack, it mainly listens to the communication channels between entities in the IIoT system and tries to obtain confidential information (e.g., production decisions) about the IIoT system. When an adversary launches an active attack, he mainly accesses the communication channel between entities in the IIoT system and then intercepts, modifies, and replays the transmitted data through that channel.

TABLE I
NOTATIONS

Notations	Definitions
KDC	Key distribution center
ES	Edge server
SD_i	The i-th smart device
$PID_{i,j}$	Pseudonym of SD_i in j-th time slot
$sk_{i,j}$	Secret key corresponding to $PID_{i,j}$
$ek_{i,j}$	Encryption key corresponding to $PID_{i,j}$
gsk	Group secret key
h_1, h_2, h_3, h_4, h_5	Five one-way hash functions
T_i	Current timestamp
m_i	The plaintext generated by SD_i
M_i	The ciphertext after encrypting m_i
$\sigma_{i,j}$	Signature corresponding to the $PID_{i,j}$
VK_x	The x-th verification key
$E_{sk}(\cdot)$	Encrypt the plaintext by key sk
$D_{sk}(\cdot)$	Decrypt the ciphertext by key sk
$List_1$	The invalid-filter for storing invalid data
$List_2$	The valid-filter for storing valid data
$FinList$	The union of $List1$ and $List2$
$NMSign$	The signature of the notification message by ES

D. Design Objectives

In this section, we present the functional objectives and security objectives that can be met in the proposed scheme.

1) Functional Objectives:

- *Batch authentication:* In the proposed scheme, batch authentication is supported, which means that ES can simultaneously verify the legitimacy of a huge amount of data from different smart devices.

2) Security Objectives:

- *Integrity:* The verifier can confirm that the received data has not been tampered with by network attackers.
- *Confidentiality:* Even if a network attacker intercepts data via Internet, it cannot obtain the plaintext of the data.
- *Anonymity:* The real identity of the smart device is protected; no network attacker except for the KDC can obtain the real identity of the smart device through the messages sent by the device.
- *Unlinkability:* A network attacker cannot discover the correlation between two pseudonyms generated by the same smart device or between signatures generated by different pseudonyms of the same device.
- *Replay attack resistance:* Since the data in this scheme satisfies integrity, the timestamp in the data cannot be modified by a network attacker. Therefore, the verifier can verify the freshness of the data by the timestamp.

IV. PROPOSED SCHEME

This section describes the proposed scheme in the following phases: system parameter generation, pseudonym and secret key generation for SD, message encrypting and signing, batch authentication, generating notification messages, and message recovery. The notations used in this scheme are shown in Table I.

A. System Parameter Generation

In our scheme, to implement the functions of encryption, decryption, signing, and verification of messages, KDC needs to generate some system parameters.

1) The KDC chooses the parameters (G, q, P) of elliptic curve as the basis for generating system parameters.
2) The KDC selects five one-way hash functions: $h_1 : G \to \{0,1\}^*$, $h_2 : \{0,1\}^* \times G \to Z_q^*$, $h_3 : \{0,1\}^* \times G \times G \times \{0,1\}^* \times \{0,1\}^* \to Z_q^*$, $h_4 : G \times G \times Z_q^* \times \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \to Z_q^*$, $h_5 : \{0,1\}^* \to Z_q^*$.
3) Once a new SD wants to join the IIoT system, the KDC needs to assign a real identity to the SD, which be represented as $RID_i \in \{0,1\}^*$.
4) In the proposed scheme, we need to guarantee the confidentiality of data during transmission while also ensuring that other legitimate data receivers can decrypt the corresponding ciphertext. Therefore, the KDC needs to generate a group secret key $gsk \in Z_q^*$.
5) To ensure the security of the IIoT system, the SD needs to verify the validity of the messages it receives. However, the computing power of SD is limited and the messages received are massive. In the proposed scheme, we let ES with stronger computing power verify the message and then broadcast the result (notification message) to the IIoT system. To ensure the integrity of the notification message, the ES needs to generate some verification keys *VKS* for signing. In the proposed scheme, KDC generates a seed *seed*, which ES can use to generate verification keys. Assume that ES requires k verification keys and the x-th *VK* be represented as $VK_x = h_4^x(seed)$.
6) The KDC sets the system parameters $params = \{G, q, Z_q^*, P_{pub}, h_1, h_2, h_3, h_4, h_5\}$, then broadcasts them to IIoT system. Finally, the KDC sends *seed* to the ES via a secure channel and sends *RID*, *gsk*, and VK_k to the corresponding SD via a secure channel.

B. Pseudonym and Secret Key Generation for SD

In the proposed scheme, to ensure the anonymity of data, SD needs to use pseudonyms. To prevent pseudonyms from being forged, the pseudonyms are generated by KDC in the following steps.

1) To achieve data unlinkability, KDC needs to generate a series of pseudonyms. First, KDC randomly chooses a number $u_{i,j} \in Z_q{}^*$ and then computes $U_{i,j} = u_{i,j} \cdot P$. Finally, the *KDC* computes the *j*-th pseudonym
$$PID_{i,j} = RID_i \oplus h_1\big(s \cdot U_{i,j}\big). \tag{1}$$
2) To ensure that SD pseudonyms cannot be used arbitrarily in other smart device signatures, KDC needs to generate a corresponding unique key for each pseudonym to be used in future SD signatures. Therefore, the KDC computes the secret key
$$sk_{i,j} = s + u_{i,j}. \tag{2}$$
3) The KDC sends $\{PID_{i,j}, sk_{i,j}, h_{i,j}, U_{i,j}\}$ to smart device SD_i via a secure channel, where $h_{i,j} = h_2(PID_{i,j}, U_{i,j})$.

C. Message Encrypting and Signing

When SD_i needs to send its real-time data, it needs to encrypt the data and sign the corresponding ciphertext to protect data privacy in the IIoT system. Subsequently, SD_i sends the signed data to ES. Note that to ensure the confidentiality of data, we apply symmetric encryption to our scheme. Also, to enhance the security of the data, we use random numbers to generate the encryption key $ek_{i,j}$, which can achieve the effect of one secret at a time.

1) First, to generate the *i*-th SD's *j*-th secret key $ek_{i,j}$, the SD_i randomly chooses a number $r_{i,j} \in Z_q{}^*$. Then the SD_i computes $R_{i,j} = r_{i,j} \cdot P$. Finally, the SD_i sets the data encryption key as
$$ek_{i,j} = h_1\big(r_{i,j} \cdot gsk \cdot P\big). \tag{3}$$
It is worth noting that no real-time messages are required to generate these parameters, so the SD can generate these parameters ahead of time and store them for future encrypting or signing messages. In the proposed scheme, when SD_i network density is not high, it is possible to perform precalculations to compute $R_{i,j}$ and then store them. These parameters are required for encrypting and signing messages.
2) Since IIoT requires high real-time data and SD_i is a device with limited computational power, the encryption algorithm should be lightweight. In the proposed scheme, we use symmetric encryption to encrypt the real-time data m_i. And the ciphertext M_i be computed by SD_i as
$$M_i = E_{ek_{i,j}}(m_i). \tag{4}$$
3) To ensure the integrity and verifiability of the final message sent by SD_i, the SD_i first generates a timestamp T_i and then computes $h_{i,j}^* = h_3(PID_{i,j}, R_{i,j}, U_{i,j}, M_i, T_i)$.
4) The SD_i sets the signature as
$$\delta_{i,j} = sk_{i,j} \cdot h_{i,j}^* + r_{i,j} \cdot h_{i,j}. \tag{5}$$
And then the SD_i sends $\{\sigma_{i,j}, M_i, T_i, PID_{i,j}\}$ to IIoT environment, where $\sigma_{i,j} = (R_{i,j}, U_{i,j}, \delta_{i,j})$.

D. Batch Authentication

When ES receives some data sent by SD, it needs to verify the received data and then broadcast the verification result into the IIoT system. Noting that the data that ES receives are massive. To reduce the computing cost of verifying these data, in our scheme, ES can validate a batch of data simultaneously. Assume that after initial data filtering, there are still n pieces of data that need to be verified by ES.

1) When ES receives data, it first checks the timestamp T_i to determine whether the data is expired. If the T_i is not fresh, ES rejects the data.
2) The ES computes $h_{i,j} = h_2(PID_{i,j}, U_{i,j})$, $h_{i,j}^* = h_3(PID_{i,j}, R_{i,j}, U_{i,j}, M_i, T_i)$.
3) To effectively prevent non-repudiation attacks, ES applies the small exponential test technique to the process of batch authentication. ES randomly selects a vector $x = \{x_1, x_2, x_3, \ldots, x_n\}$, where $x_i \in [1, 2^l]$ and l is a small integer that requires little computational cost.

4) The ES determines whether the Eq. (6) holds by computation. If this equation holds, which means the *n* different data are legal. Then the ES performs the subsequent storage and broadcast operations.

$$\left(\sum_{i=1}^{n}(x_i\delta_{i,j})\right)P = \left(\sum_{i=1}^{n}\left(x_i h_{i,j}^*\right)\right)P_{pub} + \sum_{i=1}^{n}\left(x_i h_{i,j}^* U_{i,j}\right) + \sum_{i=1}^{n}(x_i h_{i,j} R_{i,j}). \tag{6}$$

The correctness of the Eq. (6) is as follows:

$$\begin{aligned}\left(\sum_{i=1}^{n}(x_i\delta_{i,j})\right)\cdot P &= \left(\sum_{i=1}^{n}\left(x_i\cdot\left(sk_{i,j}\cdot h_{i,j}^* + r_{i,j}\cdot h_{i,j}\right)\right)\right)\cdot P\\ &= \sum_{i=1}^{n}\left(x_i\cdot\left((s+u_{i,j})\cdot h_{i,j}^* + r_{i,j}\cdot h_{i,j}\right)\right)\cdot P\\ &= \sum_{i=1}^{n}\left(x_i\cdot\left(h_{i,j}^*\cdot(P_{pub}+U_{i,j}) + h_{i,j}\cdot R_{i,j}\right)\right)\\ &= \left(\sum_{i=1}^{n}\left(x_i h_{i,j}^*\right)\right)P_{pub}\\ &\quad+ \sum_{i=1}^{n}\left(x_i h_{i,j}^* U_{i,j}\right) + \sum_{i=1}^{n}\left(x_i h_{i,j} R_{i,j}\right).\end{aligned}$$

5) If the Eq. (6) does not hold, it proves that there are some invalid data in this batch of data. For a batch of data, it may only a few invalid data exist. Suppose ES chooses to abandon the whole batch of data because of this small amount of invalid data. In that case, this will lead to the waste of valid data and the transmission delay caused by legitimate SD sending valid data again. Therefore, to improve the efficiency of batch authentication, we apply the binary search technique to this scheme and use it to find invalid data to distinguish invalid data from valid data.

Suppose that after filtering through the timestamp T_i by ES, there are still *n* data that need batch authentication. First, ES arranges the received data into a list as $List = \{data_1, data_2, data_3, \ldots, data_n\}$ according to the order of the timestamps. Then the ES sets two empty lists as $List_1$ and $List_2$, which will be used to generate notification messages. To reduce the data length of an ES release notification message, the $List_1$ will store the hash value of invalid data, and the $List_2$ will store the hash value of valid data.

The specific steps of extracting valid and invalid data are shown in Algorithm 1. And the *batchAuthenticate*(*List*, *low*, *high*) denotes batch authentication of received data by ES.

E. Generating Notification Messages

In an IIoT environment, an SD needs to receive data from other SD to adjust its production state dynamically. However, SD has limited computing power in the face of numerous data. If the data is already verified by ES and needs to be re-verified by SD, this will cause a lot of additional computational overhead. Therefore, after ES validates the data in the IIoT environment, it is necessary to generate notification messages about valid and invalid data to assist SD in data validation. In our proposed scheme, the ES generates notification messages through the following steps:

Algorithm 1 $dataExtract(List, List_1, List_2, low, high)$

if *batchAuthenticate*(*List*, *low*, *high*) $==$ *true* **then**
  for $i = low$; $i < high$; i ++ **do**
    $List_2.append(h_4(List[i]))$
    $return\ List.remove(List[i])$
  end for
else
  if $low == high$ **then**
    $List_1.append(h_4(List[low]))$
    $return\ List.remove(List[low])$
  else
    $mid = (low + high)/2$
    $dataExtract(List, List_1, List_2, low, mid)$
    $dataExtract(List, List_1, List_2, mid, high)$
  end if
end if

1) SD needs to verify the validity of the notification message, so ES needs to sign the notification message before broadcasting it. In the proposed scheme, we design a lightweight signature algorithm based on a hash chain to reduce the computational overhead caused by ES's signature on notification messages. In the system parameter generation phase, ES gets *seed* distributed by KDC, and SD gets VK_k distributed by KDC.
2) Let the *FinList* denote the union of $List_1$ and $List_2$. Before sending a notification message to the IIoT system, ES should sign *FinList* in the following way:

$$\begin{aligned}&NMSign = (VK_{k-1}\oplus VK_k)\\ &\quad||(h_5(FinList||VK_{k-1}||VK_k||T_{NM}))||FinList||T_{NM},\end{aligned} \tag{7}$$

where T_{NM} denotes the notification message generation time. Subsequently, the ES broadcasts *NMSign* to the IIoT system.

Remark: The ES signature key is composed of a hash chain. When ES is signing, the signing keys are used in the order from the back to the front of the hash chain, and the previously used signing keys are discarded. SD has VK_k, but it is impossible to get VK_{k-1} unless it gets ES's latest signature. When SD gets ES's signature, ES chooses VK_{k-2} as the next signature key, so the hash chain-based signing algorithm is secure.

F. Message Recovery

In the proposed scheme, if the ES has verified the validity of the message sent by SD_i, then when SD_j receives a message sent by SD_i according to its interest, it only needs to spend little time to perform a simple query in the valid *NMSign* to determine the validity of the message from SD_i. Assume that the *k*-th *VK* saved in SD_j is VK'_k.

1) When the SD_j receives *NMSign*, it first checks the timestamp T_{NM} to determine whether the *NMSign* is expired, if the T_{NM} is not fresh, ES rejects the *NMSign*. Otherwise, SD_j calculates $VK'_{k-1} = (VK_{k-1}\oplus VK_k)\oplus VK'_k$.

TABLE II
FOUR POSSIBLE CASES OF QUERY RESULTS

Case	$List_1$	$List_2$	Validity
1	False	True	valid
2	True	False	invalid
3	True	True	false positive
4	False	False	has not verified

2) SD_j determines if $h_5(FinList||VK_{k-1}||VK_k||T_{NM}) = h_5(FinList||VK'_{k-1}||VK'_k||T'_{NM})$ is true. If that is true, then *NMSign* is proved to be valid.
3) SD_j computes the value of $h_4(\sigma_{i,j}, M_i, T_i, PID_{i,j})$.
4) SD_j queries the value of $h_4(\sigma_{i,j}, M_i, T_i, PID_{i,j})$ in the *FinList* to determine whether the message from SD_i is valid.
   After the SD_j query list *FinList*, the query results may appear in four different cases, which are shown in the Table II. For the first case, SD_i's message can be confirmed to be valid. For the second case, SD_i's message can be confirmed to be invalid. For the third case, the hash value of the message sent by SD_i appears not only in $List_2$, but also in $List_1$. It means a false positive happens, so the ES needs to confirm the SD_i's message again. For the last case, it means the ES has not yet validated the message from SD_i, so the SD_j should wait for the next *NMSign* be broadcast from the ES.
5) If the received message of SD_i is valid, the receiver SD_j computes $ek'_{i,j} = h_1(gsk \cdot R_{i,j})$ and $m_i = D_{ek'_{i,j}}(M_i)$ to obtain the plaintext m_i.

V. SECURITY PROOF AND ANALYSIS

In this section, we show the security satisfied by the proposed scheme in terms of security proof and analysis. Note that in our proposed scheme, symmetric encryption is mainly implemented using Advanced Encryption Standard (AES), so only the unforgeability of the signature is proven in the security proof.

A. Security Proof

We prove the security of the proposed scheme in the random oracle model. The simulator $\mathcal{B}$ and the adversary $\mathcal{A}$ define the security model by playing a game. In the game, the adversary $\mathcal{A}$ could make some queries by the follows:

- *Setup phase:* First, the simulator $\mathcal{B}$ generates a set of public system parameters and secret keys, and then sends the system public parameters to the adversary $\mathcal{A}$.
- *h_1 Oracle:* The simulator $\mathcal{B}$ selects a random number $\tau \in \{0,1\}^*$, and stores (m, τ) in the list L_{h_1}. Then, the simulator $\mathcal{B}$ sends τ to the adversary $\mathcal{A}$.
- *h_2 Oracle:* The simulator $\mathcal{B}$ selects a random number $\tau \in Z_q^*$, and stores (m, τ) in the list L_{h_2}. Then, the simulator $\mathcal{B}$ sends τ to the adversary $\mathcal{A}$. Note that query process for h_3 Oracle is similar to h_2 Oracle.
- *Sign Oracle:* If the simulator $\mathcal{B}$ gets a message m_i from adversary $\mathcal{A}$, the simulator $\mathcal{B}$ generates a data $\{R_{i,j}, U_{i,j}, \delta_{i,j}, M_i, T_i, PID_{i,j}\}$, and then, the simulator $\mathcal{B}$ sends it to the adversary $\mathcal{A}$.

An adversary $\mathcal{A}$ can break the proposed scheme Γ if the $\mathcal{A}$ generates a valid signed message. Let $Adv_{\Gamma}^{Auth}(\mathcal{A})$ present the probability of $\mathcal{A}$ breaking the proposed scheme.

Definition 1: The proposed scheme Γ for IIoT is secure if $Adv_{\Gamma}^{Auth}(\mathcal{A})$ is negligible for any polynomial $\mathcal{A}$.

We evaluate the security of the proposed scheme and prove that this scheme is secure under the random oracle model.

Theorem 1: Suppose Q denotes the number of queries to the random oracle by the adversary $\mathcal{A}$, and R denotes the number of queries to the sign oracle by the adversary $\mathcal{A}$. If the adversary $\mathcal{A}$ can break the scheme within a time period T, the simulator $\mathcal{B}$ can break ECDLP within a time period T', where the $T' < 120686QT/\varepsilon$ and $\varepsilon \geq 10(R+1)(R+Q)/q$.

Proof: Supposed that an adversary $\mathcal{A}$ has the ability to forge a message $\{R_{i,j}, U_{i,j}, \delta_{i,j}, M_i, T_i, PID_{i,j}\}$. We can construct a simulator $\mathcal{B}$ has the capability to solve the ECDLP with a non-negligible probability by utilizing the adversary $\mathcal{A}$ as a subroutine. Noting that the simulator $\mathcal{B}$ maintains L_{h_1}, L_{h_2} and L_{h_3}. Given an ECDLP instance $\{P, PK_{i,j} = sk_{i,j} \cdot P | sk_{i,j} \in Z_q^*\}$, $\mathcal{B}$ simulates oracles queried by $\mathcal{A}$ as follows.

Setup: The simulator $\mathcal{B}$ sends the system parameters $params = \{G, q, Z_q^*, P_{pub}, h_1, h_2, h_3\}$ to the adversary $\mathcal{A}$.

h_1 Oracle: When the adversary $\mathcal{A}$ makes a h_1 query with message μ, the simulator $\mathcal{B}$ determines whether a tuple $< \mu, \tau_{h_1} >$ exists in list L_{h_1}. If so, the simulator $\mathcal{B}$ sends $\tau_{h_1} = h_1(\mu)$ to the adversary $\mathcal{A}$; otherwise, the simulator $\mathcal{B}$ chooses a random bit-string $\tau_{h_1} \in \{0,1\}^*$, next, it inserts $< \mu, \tau_{h_1} >$ into L_{h_1} and sends $\tau_{h_1} = h_1(\mu)$ to the adversary $\mathcal{A}$.

h_2 Oracle: Once the adversary $\mathcal{A}$ makes a h_2 query with the message $< PID_{i,j}, U_{i,j} >$, the simulator $\mathcal{B}$ determines whether a tuple $< PID_{i,j}, U_{i,j}, \tau_{h_2} >$ exists in list L_{h_2}. If so, the simulator $\mathcal{B}$ sends $\tau_{h_2} = h_2(PID_{i,j}, U_{i,j})$ to the adversary $\mathcal{A}$. Otherwise, the simulator $\mathcal{B}$ chooses a random number $\tau_{h_2} \in Z_q^*$, next, it inserts $< PID_{i,j}, U_{i,j}, \tau_{h_2} >$ into list L_{h_2} and sends $\tau_{h_2} = h_2(PID_{i,j}, U_{i,j})$ to the adversary $\mathcal{A}$.

h_3 Oracle: Once the adversary $\mathcal{A}$ makes a h_3 query with the message $<PID_{i,j}, R_{i,j}, U_{i,j}, M_i, T_i>$, the simulator $\mathcal{B}$ determines whether a tuple $< PID_{i,j}, R_{i,j}, U_{i,j}, M_i, T_i, \tau_{h_3} >$ exists in list L_{h_3}. If so, the simulator $\mathcal{B}$ sends $\tau_{h_3} = h_3(PID_{i,j}, R_{i,j}, U_{i,j}, M_i, T_i)$ to the adversary $\mathcal{A}$. Otherwise, the simulator $\mathcal{B}$ chooses a random number $\tau_{h_3} \in Z_q^*$, inserts $< PID_{i,j}, R_{i,j}, U_{i,j}, M_i, T_i, \tau_{h_3} >$ into list L_{h_3} and sends $\tau_{h_3} = h_3(PID_{i,j}, R_{i,j}, U_{i,j}, M_i, T_i)$ to the adversary $\mathcal{A}$.

Sign query: When the adversary $\mathcal{A}$ uses $PID_{i,j}$ for a sign query on a message M_i, the simulator $\mathcal{B}$ queries $h_{i,j} = h_2(PID_{i,j}, U_{i,j})$, $h^*_{i,j} = h_3(PID_{i,j}, R_{i,j}, U_{i,j}, M_i, T_i)$ through lists L_{h_2} and L_{h_3} respectively. Then, the simulator $\mathcal{B}$ selects a random numbers $\delta_{i,j} \in Z_q^*$. Next, the $\mathcal{B}$ computes $R_{i,j} = (\delta_{i,j} \cdot P - h^*_{i,j} \cdot PK_{i,j}) \cdot h_{i,j}^{-1}$. Last, the simulator $\mathcal{B}$ sends $< M_i, R_{i,j}, \delta_{i,j} >$ to the adversary $\mathcal{A}$.

Analysis: Through forking lemma [35], the adversary $\mathcal{A}$ can construct two valid signatures $(R_{i,j}, \delta_{i,j} = h_{i,j} \cdot r_{i,j} + h^*_{i,j} \cdot sk_{i,j})$, $(R_{i,j}, \delta'_{i,j} = h_{i,j} \cdot r_{i,j} + h^{*'}_{i,j} \cdot sk_{i,j})$, and the simulator

$\mathcal{B}$ can get $sk_{i,j}$ by computing

$$\begin{aligned}&\frac{\delta_{i,j}-\delta'_{i,j}}{h^*_{i,j}-h^{*'}_{i,j}} \pmod q\\&=\left(\frac{h_{i,j}\cdot r_{i,j}+h^*_{i,j}\cdot sk_{i,j}-h_{i,j}\cdot r_{i,j}-h^{*'}_{i,j}\cdot sk_{i,j}}{h^*_{i,j}-h^{*'}_{i,j}}\right) \pmod q\\&=sk_{i,j}\end{aligned}\tag{8}$$

In summary, the simulator $\mathcal{B}$ can break the ECDLP within the time T', where $T' < 120686QT/\varepsilon$, where $\varepsilon \geq 10(R+1)(R+Q)/q$. Therefore, the scheme is secure under the random oracle model. ■

B. Security Analysis

Combining the threat model and security analysis, we demonstrate the security properties met by the scheme. It is worth noting that in the proposed scheme, we design two signature algorithms, and these two signature algorithms are executed by SD and ES, respectively. Moreover, according to the threat model, we can know that the malicious network attacker can intercept and tamper with data.

1) *Integrity:* For a smart device SD_i, according to Theorem 1, we can know that SD_i's signature cannot be forged because solving the ECDLP is hard. Therefore, if a network attacker launches an active attack, tempered with the data $\{\sigma_{i,j}, M_i, T_i, PID_{i,j}\}$ and then broadcasts the tampered data to the IIoT system, the ES can use Eq. 6 and binary search to quickly find this illegal data, so the scheme can guarantee the integrity of SD_i's signature.

   For ES, it sends notification messages signature $NMSign = (VK_{k-1} \oplus VK_k)||(h_4(FinList||VK_{k-1}||VK_k||T_{NM}))||FinList||T_{NM}$ to the IIoT system. If a network attacker launches an active attack, tampered with the *NMSign* and broadcasts the tampered *NMSign* to the IIoT system, the receiver SD_j can calculate $VK'_{k-1} = (VK_{k-1} \oplus VK_k) \oplus VK'_K$, and find out that $h_4(FinList||VK_{k-1}||VK_k||T_{NM}) = h_4(FinList||VK'_{k-1}||VK'_k||T'_{NM})$ is false in time. Therefore, the proposed scheme can guarantee the integrity of *ES*'s signature.
2) *Confidentiality:* For a smart device SD_i, in the proposed scheme, before SD_i sends a message, SD_i first encrypts the plaintext by symmetric encryption, then signs the ciphertext, and finally broadcasts the processed message. For ES, to process messages from SD_i and broadcast them to the IIoT system, SD_i's original data m_i always exists in ciphertext form.

   To sum up, from SD_i sending a message to SD_j receiving the corresponding message, the m_i in the whole process is always in the form of ciphertext. When a network attacker launches a passive attack or an active attack to obtain the message sent by SD_i, it cannot get the corresponding plaintext m_i because it does not have the encryption key $ek_{i,j}$. Therefore, the proposed scheme can guarantee the confidentiality of data m_i.
3) *Anonymity:* The anonymity implies that the signature of the SD_i is anonymous. Since there is only one ES in the system model, the signature of the ES does not need to be anonymous. In the proposed scheme, before SD_i broadcasts a message, it hides its real identity in a pseudonym. When a network attacker launches a passive attack or an active attack to obtain a message, it cannot get the real identity RID_i of the smart device SD_i unless it has a number $u_{i,j}$ and the system master key *s*. However, in our proposed scheme, $u_{i,j}$ and *s* are all stored in *KDC*, so it is not accessible to a network attacker. Therefore, the proposed scheme can ensure the anonymity of SD_i.
4) *Unlinkability:* The unlinkability implies that the signature of the SD_i is unlinkable. Since there is only one ES in the system model, the signature of the ES does not need to be unlinkable. In our proposed scheme, the pseudonym is obtained by calculating $PID_{i,j} = RID_i \oplus h_1(s \cdot u_{i,j} \cdot P)$, which contains a random number $u_{i,j}$, and this number uniquely corresponds to a secret key $sk_{i,j}$ and a pseudonym $PID_{i,j}$. In other words, each signature from SD_i corresponds to a random number and a pseudonym. There is no connection between these random numbers, and there is no connection between these pseudonyms. Therefore, when a network attacker launches a passive or active attack to obtain two messages generated by two different pseudonyms, it cannot link the two messages.
5) *Replay attack resistance:* When a network attacker launches an active attack and replays the message, the ES can verify the timestamp and find that the message is not within the validity period, then the ES will reject the message. Similarly, the ES signature for notification messages has the corresponding timestamp. Therefore, the proposed scheme is resistant to replay attacks.

VI. Performance Analysis

In this section, we use the experiment to prove the feasibility and superiority of our proposed scheme.

A. Experiment Setup

1) Experimental Environment: We use c++ code to implement the proposed scheme. The cryptographic library we use is Miracl Core [36], and we choose the BLS12381 curve (which provides 128-bit security level) to implement the basic operations of elliptic curves. In addition, the symmetric encryption we use is AES, and we use hashmap to implement *FinList*. As shown in Fig. 4, we use a PC to simulate the edge server in the proposed scheme. The operating system on this PC is Ubuntu 18.04.3 with an Intel Core i5-7500 CPU at 3.40GHz and 16GB of memory. In the proposed scheme, smart devices have limited computing power, so we use a Raspberry Pi 4 to simulate a smart device. The Raspberry Pi has a 1.5GHz CPU and 4GB memory. Here, the edge server and Raspberry Pi are connected to the same router via a wired network for data transmission stability. The router is a Gigabit router. It is worth noting that the PC does the

TABLE III
FOUR CASES IN IIoT

	Batch authentication	ES	ES signature	List	Message transfer path
case1	×	×	×	×	SD_i-SD_j
case2	√	×	×	×	SD_i-SD_j
case3	√	√	ECDSA	√	SD_i-ES-SD_j
Ours	√	√	HashSig	√	SD_i-ES-SD_j

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

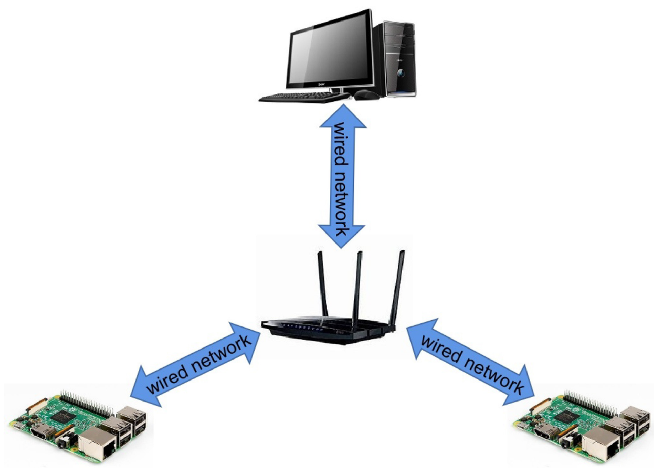


Fig. 4. Experimental network topology.

batch authentication of messages and the generation of notification messages in our experiments. Other operations, such as encryption, decryption, and signing of the initial messages, are done by the Raspberry Pi 4. The router does not participate in any computation and only provides the network to all devices.

2) Cases in IIoT: In the IIoT system, it is generally the case that smart devices communicate directly with each other and authenticate the received messages one by one. To prove that (1) batch authentication algorithm can improve the efficiency of message authentication; (2) using edge servers to assist smart devices in message authentication can significantly improve the efficiency of smart device message authentication; (3) hash chain-based signature algorithm is lightweight, we set four cases. As shown in Table III, case1 is a typical case in the IIoT system, where SD_j authenticates the received data one by one. To highlight the efficiency of batch authentication, in case2, SD_j performs batch authentication for the received messages. Furthermore, to demonstrate that the efficiency of message authentication can be improved with the assistance of ES, case3 lets ES perform batch authentication and sign the authentication results using the elliptic curve digital signature algorithm (ECDSA). Finally, to prove that the proposed hash chain-based signature (labeled as HashSig) is lightweight, we set case4. The only difference between case4 and case3 is using HashSig instead of ECDSA. Note that our proposed scheme is case4 (labeled as Ours). In addition, to further show that the proposed scheme is lightweight, we compare the proposed scheme with related schemes [19], [30].

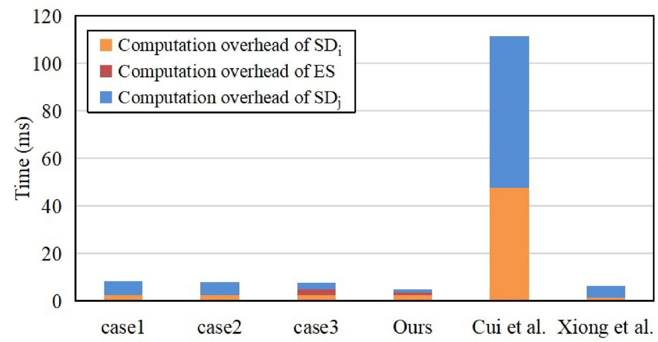


Fig. 5. Comparison of the total time overhead when the number of messages is 1.

B. Experimental Results

1) Advantages of Batch Authentication Scheme Based on Edge Computing: We conduct experiments on case1, case2, case3, and Ours, according to the experimental setup, and the results are as follows.

- For SD_i, it first generates some parameters related to encryption and signature, then encrypts and signs the generated original data. After testing, we obtain that SD_i spend total time in case1 is 2.623 ms, in case2 is 2.505 ms, in case3 is 2.509 ms and in Ours is 2.507 ms. The difference in the time cost by SD_i is negligible because the SD_i's operation is the same.
- The ES performs batch authentication of the received data and signs the authentication result. According to the experimental setup, only case3 and Ours use ES. When the number of authenticated data is 1, the total time of ES consumed in case3 is 2.292 ms and in Ours is 1.159 ms.
- In case1 and case2, SD_j authenticates the received messages and then decrypts them to obtain the original data. In case3 and Ours, SD_j performs authentication of the received message with the assistance of the authentication result signed by ES and performs decryption to get the original data. When the number of data is 1, the total time consumed by SD_j is 5.681 ms in case1, 5.557 ms in case2, 2.826 ms in case3, and 1.250 ms in Ours.

When the number of messages is 1, the total time spent in case1 is about $2.623 + 5.681 = 8.304$ ms. We calculate the total time for other cases using the same way and represent it in Table IV, Fig. 5, and Fig. 6.

From Fig. 6, we find that as the number of messages continues to increase, the total time required for case1 and case2 to process data also increases, and the difference in time cost between the two cases gets larger. When the number of messages reaches 180, the time cost in case2 is

TABLE IV
THE TOTAL TIME COST IN FIVE CASES (MS)

number of messages	1	20	40	60	80	100	120	140	160	180
case1	8.304	112.897	221.409	330.351	438.011	546.897	656.874	767.616	876.722	982.283
case2	8.062	72.756	140.478	208.828	276.620	344.881	413.654	482.219	549.989	618.517
case3	7.627	42.410	78.921	115.429	152.307	188.771	225.691	260.120	299.074	334.573
Ours	4.916	39.752	76.540	113.150	149.591	185.894	222.525	259.898	296.096	330.005
Cui *et al.* [19]	111.315	1321.598	2596.497	3870.815	5145.959	6429.417	7697.581	8973.244	10248.620	11528.180
Xiong *et al.* [30]	6.218	97.081	192.365	288.545	381.752	479.956	575.872	671.434	766.753	862.276

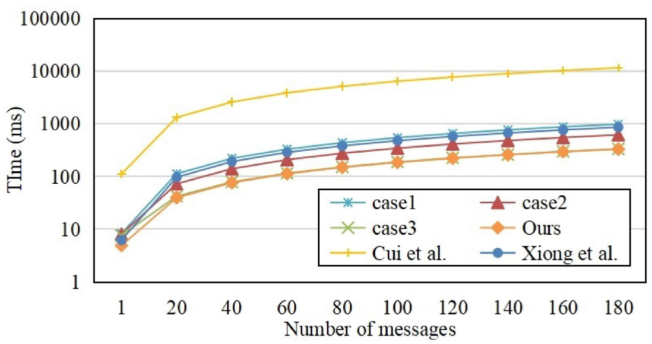


Fig. 6. Comparison of the total time overhead.

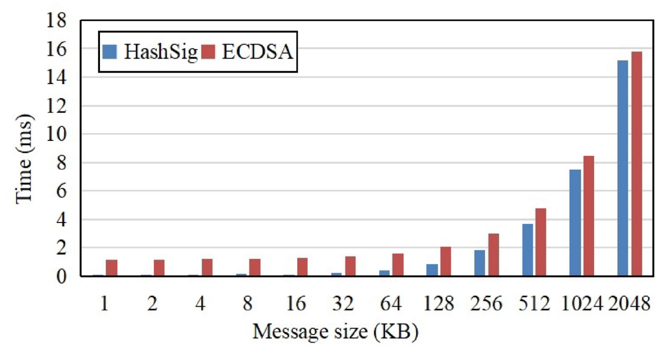


Fig. 7. Comparison of signing time between HashSig and ECDSA.

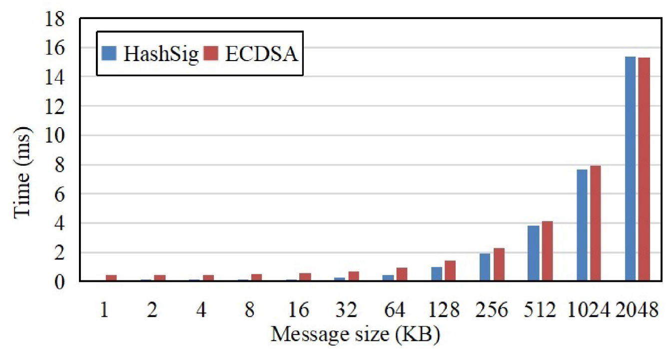


Fig. 8. Comparison of verification time between HashSig and ECDSA.

about 363.766 ms less than that in case1, reflecting batch authentication's superiority.

As shown in Table IV, we find that when the number of messages is 20, the time cost in case3 is 30.346 ms less than that in case2. And from Fig. 6, we find that the difference in total time overhead between case2 and case3 is getting more significant as the number of messages increases. Therefore, using ES to assist SD_j with message authentication can effectively reduce the total time overhead. In addition, by combining Table IV and Fig. 6, we can see that the total time overhead in case3 and Ours is always about the same because the difference between these two cases is that ES uses different signatures, and ECDSA and HashSig are both lightweight signatures.

In Fig. 5, We find that the total computational overhead of Ours is the lowest when the number of messages is 1, which is about 4.42% of [19] and 79.06% of [30]. The reason is that in [19], smart devices with limited computational power need to perform many time-consuming bilinear pairing operations. And in [30], although the authentication algorithm is lightweight, using edge servers to assist smart devices in authentication is not considered. From Fig. 6, we can see that the computational overheads of [19] and [30] are consistently higher than those of Ours. Therefore, compared with schemes [19] and [30], our proposed lightweight edge-assisted batch authentication scheme is more suitable for IIoT systems.

2) Advantages of HashSig: In the proposed scheme, to prove that the HashSig algorithm is lightweight, we compare the HashSig algorithm with the lightweight ECDSA. HashSig signature and verification time will be tested on the ES side and compared with ECDSA signature and verification time. The results are shown in Fig. 7 and Fig. 8. From Fig. 7 and Fig. 8, we find that the time consumed by HashSig is very close to the time consumed by ECDSA. Although the HashSig verification time exceeds the ECDSA verification time as the length of the message increases to 2048 KB, its verification time is still short. Combining Table IV, Fig. 5 and Fig. 6, we can see that the total time consumed by Ours is very close to the total time consumed by case3, and the time cost of HashSig is negligible as a percentage of the total time. Therefore, our proposed HashSig is a lightweight signature algorithm that can be applied to IIoT environments with high data real-time requirements.

C. Communication Overhead

In the cryptographic library, the elements in G are 97 bytes, and the theoretical size of large integers is 48 bytes. We set the size of the plaintext message to be 56 bytes and the timestamp to be 16 bytes.

In case1, case2, case3, and Ours, the signatures used by SD_i are the same, so the data sent by SD_i are $\{\sigma_{i,j}, M_i, T_i, PID_{i,j}\}$, and the length of the data they send is also the same, about $97 + 97 + 48 + 64 + 16 + 56 = 378$

bytes. For case3 and Ours, the length of the data sent by ES depends on the signature algorithm they use and the number of batch authentication messages. For example, the number of messages is n. If the ECDSA signature algorithm is used, the length of the final data sent is 96 + 49*n bytes, where 96 bytes is the size of the parameters needed to verify ECDSA and 49 bytes is the size of each message after processing. If the signature algorithm used is HashSig, the final data sent is $(VK_{k-1} \oplus VK_k)||(h_5(FinList||VK_{k-1}||VK_k||T_{NM}))||FinList||T_{NM}$, length is 56 + 48 + 16 + 49*n = 120 + 49*n bytes. Although the data finally transmitted using HashSig signature is longer than that using ECDSA signature, the difference is only 24 bytes with the same number of messages, which does not produce significant transmission delay and is still suitable for IIoT environments.

We use the same method to calculate the communication overhead for [19] and [30]. In [19], the length of the data sent by SD_i is about 395 bytes, which is 17 bytes longer than Ours. In [30], the length of the data sent by SD_i is about 752 bytes, which is 374 bytes longer than Ours. In addition, the length of the data sent by ES in [30] is about 732*n bytes, which is 683*n − 120 bytes longer than Ours. Note that in [19], the data sender and receiver communicate directly with each other, so the length of the data sent by ES is 0 bytes. However, the total computational overhead in our proposed scheme is less than that in [19].

VII. Conclusion

This paper proposes an efficient edge computing-based batch authentication scheme to protect privacy-sensitive data in IIoT environments. First, we design an ECC-based batch authentication algorithm to improve the efficiency of verifying messages sent from SD. Second, we use edge servers to reduce the authentication overhead of smart devices. Third, we design a lightweight signature based on a hash chain to improve the efficiency of ES in signing notification messages and the efficiency of SD in verifying notification messages. The security proof and analysis demonstrate that the scheme provides high security and can meet the security requirements of the IIoT system. Experimental results and performance analysis show that the scheme has a lower computing cost, further proving the scheme's feasibility in the IIoT environment. However, the proposed scheme is more suitable for a single administrative domain and does not consider authentication between devices in cross-domain IIoT. Therefore, in our future work, we will introduce blockchain technology to design a practical and lightweight authentication scheme for cross-domain IIoT environments.

Acknowledgment

The authors are very grateful to the anonymous referees for their detailed comments and suggestions regarding this paper.

References

[1] M. M. Dhanvijay and S. C. Patil, "Internet of Things: A survey of enabling technologies in healthcare and its applications," *Comput. Netw.*, vol. 153, pp. 113–131, Apr. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128619302695

[2] R. Ande, B. Adebisi, M. Hammoudeh, and J. Saleem, "Internet of Things: Evolution and technologies from a security perspective," *Sustain. Cities Soc.*, vol. 54, Mar. 2020, Art. no. 101728. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210670719303725

[3] S. Nižetić, P. Šolić, D. L.-D.-I. González-de-Artaza, and L. Patrono, "Internet of Things (IoT): Opportunities, issues and challenges towards a smart and sustainable future," *J. Clean. Prod.*, vol. 274, Nov. 2020, Art. no. 122877. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095965262032922X

[4] D. S. Gupta, S. H. Islam, M. S. Obaidat, P. Vijayakumar, N. Kumar, and Y. Park, "A provably secure and lightweight identity-based two-party authenticated key agreement protocol for IIoT environments," *IEEE Syst. J.*, vol. 15, no. 2, pp. 1732–1741, Jun. 2021.

[5] A. K. Sahu, A. K. Sahu, and N. K. Sahu, "A review on the research growth of industry 4.0: IIoT business architectures benchmarking," *Int. J. Bus. Anal.*, vol. 7, no. 1, pp. 77–97, 2020.

[6] J. Singh, A. Gimekar, and S. Venkatesan, "An efficient lightweight authentication scheme for human-centered industrial Internet of Things," *Int. J. Commun. Syst.*, to be published.

[7] M. Ghobakhloo, "Industry 4.0, digitization, and opportunities for sustainability," *J. Clean. Prod.*, vol. 252, Apr. 2020, Art. no. 119869. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0959652619347390

[8] 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 Commun. Surveys Tuts.*, vol. 22, no. 4, pp. 2462–2488, 4th Quart., 2020.

[9] A. Sari, A. Lekidis, and I. Butun, "Industrial networks and IIoT: Now and future trends," in *Industrial IoT*. Cham, Switzerland: Springer, 2020, pp. 3–55.

[10] P. K. Illa and N. Padhi, "Practical guide to smart factory transition using IoT, big data and edge analytics," *IEEE Access*, vol. 6, pp. 55162–55170, 2018.

[11] M. Humayun, N. Jhanjhi, B. Hamid, and G. Ahmed, "Emerging smart logistics and transportation using IoT and blockchain," *IEEE Internet Things Mag.*, vol. 3, no. 2, pp. 58–62, Jun. 2020.

[12] C. Perera, M. Barhamgi, A. K. Bandara, M. Ajmal, B. Price, and B. Nuseibeh, "Designing privacy-aware Internet of Things applications," *Inf. Sci.*, vol. 512, pp. 238–257, Feb. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025519309120

[13] M. A. Azad, S. Bag, F. Hao, and A. Shalaginov, "Decentralized self-enforcing trust management system for social Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2690–2703, Apr. 2020.

[14] J. W. Guck, A. Van Bemten, and W. Kellerer, "DetServ: Network models for real-time QoS provisioning in SDN-based industrial environments," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 4, pp. 1003–1017, Dec. 2017.

[15] T. Hussain, K. Muhammad, J. D. Ser, S. W. Baik, and V. H. C. de Albuquerque, "Intelligent embedded vision for summarization of multiview videos in IIoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2592–2602, Apr. 2020.

[16] M. Shen *et al.*, "Blockchain-assisted secure device authentication for cross-domain industrial IoT," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 942–954, May 2020.

[17] B. Varghese and R. Buyya, "Next generation cloud computing: New trends and research directions," *Future Gener. Comput. Syst.*, vol. 79, pp. 849–861, Feb. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X17302224

[18] G. Aceto, V. Persico, and A. Pescapé, "Industry 4.0 and health: Internet of things, big data, and cloud computing for *Healthcare* 4.0," *J. Ind. Inf. Integr.*, vol. 18, Jun. 2020, Art. no. 100129. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2452414X19300135

[19] J. Cui, F. Wang, Q. Zhang, Y. Xu, and H. Zhong, "Anonymous message authentication scheme for semitrusted edge-enabled IIoT," *IEEE Trans. Ind. Electron.*, vol. 68, no. 12, pp. 12921–12929, Dec. 2021.

[20] Y. Wu, H.-N. Dai, and H. Wang, "Convergence of blockchain and edge computing for secure and scalable IIoT critical infrastructures in industry 4.0," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2300–2317, Feb. 2021.

[21] S. Chen, Z. Pang, H. Wen, K. Yu, T. Zhang, and Y. Lu, "Automated labeling and learning for physical layer authentication against clone node and Sybil attacks in industrial wireless edge networks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 3, pp. 2041–2051, Mar. 2021.

[22] W. Ali, I. Ud Din, A. Almogren, M. Guizani, and M. Zuair, "A lightweight privacy-aware IoT-based metering scheme for smart industrial ecosystems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 9, pp. 6134–6143, Sep. 2021.

[23] J. Cui, L. Wei, J. Zhang, Y. Xu, and H. Zhong, "An efficient message-authentication scheme based on edge computing for vehicular ad hoc networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 5, pp. 1621–1632, May 2019.

[24] S. Jiang, X. Zhu, and L. Wang, "An efficient anonymous batch authentication scheme based on HMAC for VANETs," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 8, pp. 2193–2204, Aug. 2016.

[25] Z. Yang, C. Jin, Y. Tian, J. Lai, and J. Zhou, "LiS: Lightweight signature schemes for continuous message authentication in cyber-physical systems," in *Proc. 15th ACM Asia Conf. Comput. Commun. Security*, New York, NY, USA, 2020, pp. 719–731. [Online]. Available: https://doi.org/10.1145/3320269.3372195

[26] S. Yu, J. YoungLee, M. Kim, and Y. Park, "A secure biometric based user authentication protocol in wireless sensor networks," in *Proc. 10th Annu. Comput. Commun. Workshop Conf. (CCWC)*, 2020, pp. 830–834.

[27] C. Esposito, A. Castiglione, F. Palmieri, and A. De Santis, "Integrity for an event notification within the industrial Internet of Things by using group signatures," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3669–3678, Aug. 2018.

[28] S. Hussain, S. A. Chaudhry, O. A. Alomari, M. H. Alsharif, M. K. Khan, and N. Kumar, "Amassing the security: An ECC-based authentication scheme for Internet of drones," *IEEE Syst. J.*, vol. 15, no. 3, pp. 4431–4438, Sep. 2021.

[29] X. Li, J. Niu, M. Z. A. Bhuiyan, F. Wu, M. Karuppiah, and S. Kumari, "A robust ECC-based provable secure authentication protocol with privacy preserving for industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3599–3609, Aug. 2018.

[30] H. Xiong, Y. Wu, C. Su, and K.-H. Yeh, "A secure and efficient certificateless batch verification scheme with invalid signature identification for the Internet of Things," *J. Inf. Security Appl.*, vol. 53, Aug. 2020, Art. no. 102507. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214212619307999

[31] J. Liu, H. Cao, Q. Li, F. Cai, X. Du, and M. Guizani, "A large-scale concurrent data anonymous batch verification scheme for mobile healthcare crowd sensing," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1321–1330, Apr. 2019.

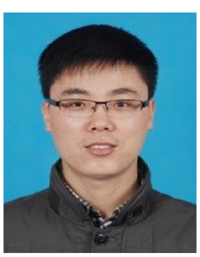
[32] C. Guo, X. Jiang, K.-K. R. Choo, X. Tang, and J. Zhang, "Lightweight privacy preserving data aggregation with batch verification for smart grid," *Future Gener. Comput. Syst.*, vol. 112, pp. 512–523, Nov. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X1933225X

[33] J. Zhang, H. Zhong, J. Cui, Y. Xu, and L. Liu, "An extensible and effective anonymous batch authentication scheme for smart vehicular networks," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3462–3473, Apr. 2020.

[34] H. Yang *et al.*, "Blockchain-enabled tripartite anonymous identification trusted service provisioning in industrial IoT," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 2419–2431, Feb. 2022.

[35] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *J. Cryptol.*, vol. 13, no. 3, pp. 361–396, 2000.

[36] "Miracl core." Accessed: Oct. 12, 2020. [Online]. Available: https://github.com/miracl/core



Jie Cui (Senior Member, IEEE) was born in Henan Province, China, in 1980. He received the Ph.D. degree from the University of Science and Technology of China in 2012. He is currently a Professor and a Ph.D. Supervisor with the School of Computer Science and Technology, Anhui University. 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. His current research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security, and software-defined networking.

Fengqun Wang is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Anhui University. His research focuses on the security of Industrial Internet of Things.

Qingyang Zhang was born in Anhui Province, China, in 1992. He received the B.Eng. and Ph.D. degrees in computer science from Anhui University in 2021, where he is currently a Lecturer with the School of Computer Science and Technology. His research interest includes edge computing, computer systems, and security.

Chengjie Gu received the Ph.D. degree from the Nanjing University of Posts and Telecommunications in 2012. From 2012 to 2017, he was an Innovation Team Leader with the 38th Research Institute of CETC and conducted research and development in the communication and networking sector. He is currently a President of Security Research Institute with New H3C Group. He is also supported by a Postdoctoral Fellowship with 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.

Hong Zhong was born in Anhui Province, China, in 1965. She received the Ph.D. degree in computer science from the University of Science and Technology of China in 2005. She is currently a Professor and a Ph.D. Supervisor with the School of Computer Science and Technology, Anhui University. 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. Her research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security, and software-defined networking.