*mathematics*

MDPI

*Article*

# Efficient Conditional Privacy-Preserving Authentication Scheme for Safety Warning System in Edge-Assisted Internet of Things

**Jianfeng Li [1], Na Hou [2], Guangwei Zhang [1], Jihao Zhang [1], Yu Liu [1] and Xiang Gao [3,*]**

[1] School of Mechatronical Engineering, Beijing Institute of Technology, Beijing 100081, China; 3420205008@bit.edu.cn (J.L.); 7520220037@bit.edu.cn (G.Z.); zhangjihao@bit.edu.cn (J.Z.); 3420195007@bit.edu.cn (Y.L.)
[2] System Engineering Research Institute, Academy of Military Sciences, Beijing 100141, China
[3] School of Integrated Circuits and Electronics, Beijing Institute of Technology, Beijing 100081, China
* Correspondence: bitxianggao@bit.edu.cn

**Abstract:** With the advent of smart cities, the significance of the Internet of Things (IoT) is gaining greater prominence. At the same time, the safety early warning system in the IoT has a significant impact on real-time monitoring and the response to potential risks. Despite the advancements made in edge-assisted IoT deployments, several challenges and constraints persist. Given the potential threat to life posed by safety-related messages, ensuring the authenticity of messages in the edge-assisted IoT safety warning system is crucial. However, considering the identity privacy of devices participating in the edge-assisted Internet of Things system, directly verifying the identity of the sending device is undesirable. To address this issue, in this work, we design a linkable group signature scheme that allows devices to anonymously send safety-related messages to edge nodes, defending against Sybil attacks while ensuring the traceability of malicious device identities. Then, we present a high-efficiency conditional privacy-preserving authentication (CPPA) scheme based on the designed group signatures for the safety warning system in edge-assisted IoT. This scheme effectively protects device identity privacy while providing a reliable authentication mechanism to ensure the credibility and traceability of alert messages. The proposed scheme contributes to the field of safety warning systems in the context of edge-assisted IoT, providing a robust solution for privacy preservation and authentication.

**Keywords:** conditional privacy-preserving authentication; group signature; safety warning system; Internet of Things

**MSC:** 94A60

check for updates

**Citation:** Li, J.; Hou, N.; Zhang, G.; Zhang, J.; Liu, Y.; Gao, X. Efficient Conditional Privacy-Preserving Authentication Scheme for Safety Warning System in Edge-Assisted Internet of Things. *Mathematics* **2023**, *11*, 3869. https://doi.org/10.3390/math11183869

Academic Editor: Daniel-Ioan Curiac

Received: 11 July 2023
Revised: 1 September 2023
Accepted: 4 September 2023
Published: 11 September 2023

**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

## 1. Introduction

The concept of the Internet of Things (IoT) [1] has led to a vast array of physical objects that are equipped with sensors and electronics. This development has revolutionized numerous domains including smart homes [2], intelligent transportation [3], industrial automation [4], and so on. This innovation opens up new possibilities for societal transformation and enhances our quality of life. However, with the explosive growth of the number of mobile devices worldwide, IoT devices have severe limitations in computing ability, storage, communication, and security, making it difficult for resource-constrained IoT applications to provide satisfactory computing and storage services. Recently, the concept of edge computing has emerged as an expansion of the traditional cloud computing model by deploying computing servers densely throughout the network. The goal is to sink computing, storage, and communication from the cloud to the network edge, allowing users to access computing services in close proximity. There are numerous case studies on

edge-assisted IoT, encompassing cloud offloading, smart homes/city applications, challenges, and future research goals [5]. Edge-assisted IoT solutions can effectively achieve low-latency, high-bandwidth, and localized service features.

The safety warning system constitutes a vital safety-focused application within IoT environments, involving the collection and analysis of safety-related messages. For instance, this includes traffic information in the VANET network and sensor data in smart homes. Figure 1 presents the standard three-layer edge-assisted IoT paradigm architecture [6]. The edge nodes, located in the edge layer, serve as a mediator to enable the localization of IoT services and data storage, bridging the upper cloud layer and the button device layer. The design and implementation of edge-assisted IoT, including edge-assisted IoT-enabled safety warning systems, raise various security and privacy concerns. Firstly, the transmission of safety-related messages through open networks in IoT can expose them to potential attacks (e.g., modification attacks), leading to real-world consequences, such as accidents.

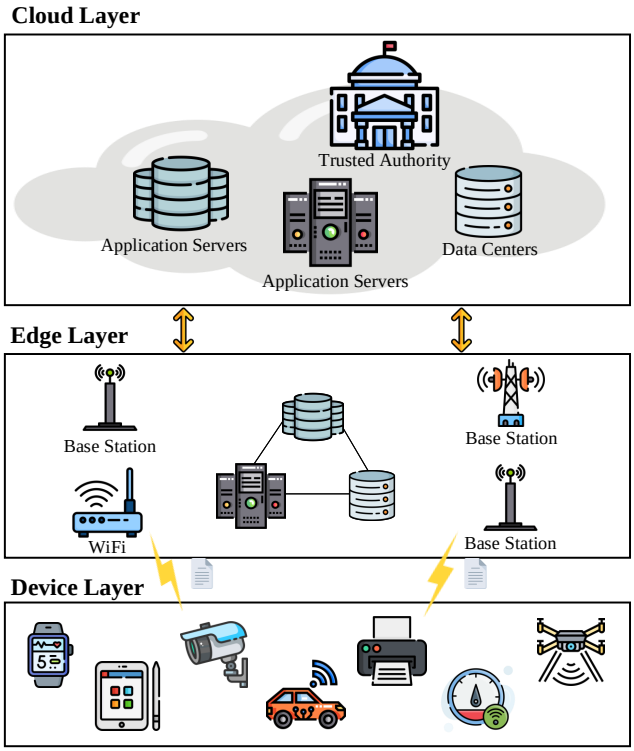


**Figure 1.** The typical structure of the edge-assisted IoT paradigm.

In the realm of IoT applications, the data generated encompass different facets of physical surroundings. While certain data may sometimes be confidential and sensitive, for example, health status, preferences, personal activities, and industrial designs, other data can be made publicly available, such as temperature information, air quality, gas concentration (such as carbon dioxide, carbon monoxide, etc.), social events, and so on. The widely held belief is that the ownership of all generated data lies with the respective data owners. However, in many cases, these data are often shared among multiple entities without the explicit permission of the owner in order to explore the potential utility [6]. This raises serious privacy concerns for users [7]. However, absolute anonymity can give rise to certain issues, such as the potential for malicious users to exploit strong anonymity and disseminate false information without being identified. Hence, it is imperative to establish a mechanism for identifying and penalizing malicious users who violate the system. That is, the privacy of users should be conditional.

A group signature [8] is a conditional anonymous digital signature scheme that is proposed to hide the identity of the signer in a group. If users honestly sign messages on behalf of the group they belong to, then any public verifier can verify the validity of

a generated signature without knowing which group member the signature belongs to. Since a group manager can de-anonymize any user, exposing the ownership information of a group signature. Due to the strong anonymity, i.e., non-linkability, provided by conventional group signatures, attackers can send false messages without fear of being caught, thereby preventing punitive actions against them. This can lead to Sybil attacks, where a malicious device generates a false message and then endorses this message by computing a number of signatures. As the signatures are unlinkable, no one can ascertain that all of these signatures originate from the same device. To defend against Sybil attacks, the concept of a message-linkable group signature (MLGS) scheme is designed [9]. MLGS provides a verification mechanism to determine whether two given MLGS signatures on the same message are provided by the identical user.

### *1.1. Contributions*

Unlike traditional IoT systems, our architecture relies on edge nodes for real-time safety warnings, necessitating rapid and reliable authentication. These edge nodes also introduce the need for accountability and resilience against Sybil attacks. To overcome shortcomings, such as high computing and communication costs, privacy breaches, and data misuse, this work presents a linkable group signature scheme. We design a high-efficiency conditional privacy-preserving authentication (CPPA) scheme for a safety warning system in edge-assisted IoT by using the proposed linkable group signature scheme. This work makes the following contributions.

- **Linkable group signature scheme with enhanced anonymity and accountability**: We present a new linkable group signature scheme with a variant of the Boneh–Boyen–Shacham (BBS) signature [10]. Our scheme not only allows edge devices to anonymously send safety-related messages to sensors, but also supports message linkability for linking group signatures from the same device to counteract Sybil attacks.
- **Efficient conditional privacy-preserving authentication (CPPA) protocol**: We present an efficient CPPA protocol for a safety warning system in edge-assisted IoT. By using the designed linkable group signature scheme, our work can achieve a balance between the anonymity and accountability of edge nodes. This is a key challenge in ensuring the integrity of safety warnings while maintaining privacy. Moreover, our protocol is designed to effectively counteract Sybil attacks, which further strengthens the security measures within the system.
- **Demonstrated security and performance advantages**: We demonstrate the effectiveness of our scheme in meeting the security requirements, and the evaluation of computation and communication overhead reveals that our scheme outperforms existing schemes in terms of performance. Thereby, our designed CPPA scheme is well-suited for the safety warning system in edge-assisted IoT.

### *1.2. Organization*

The study is arranged as follows. Section 2 provides a comprehensive review of the related work in the field. In Section 3, this work recalls several building blocks of cryptographic primitives and number-theoretic assumptions. Section 4 introduces the system framework, threat model, and design goals of the study. We depict an in-depth explanation of the construction process and the CPPA scheme for a safety warning system in edge-assisted IoT in Sections 5 and 5.2, respectively. Section 6 analyzes the security properties and evaluates the performance of the proposed scheme. Eventually, Section 7 provides the conclusion of the study.

## 2. Related Work

There are several promising edge-assisted IoT applications, including industrial IoT, autonomous driving, and smart homes. Currently, several research studies have been conducted on privacy-preserving schemes in edge-assisted IoT. Wu et al. [9] developed a method that protects vehicle privacy and ensures message reliability in vehicle-to-vehicle

(V2V) communications. Huang et al. [11] introduced a distributed reputation management system for security protection and efficiency optimization with the assistance of vehicular edge computing servers, e.g., base stations, roadside units (RSUs), and Wi-Fi hotspots. Ni et al. [6] explored the security, privacy, and efficiency concerns in edge-assisted IoT, and provided research opportunities to address these issues. Kang et al. [12] established a robust mechanism for secure data storage and sharing in vehicular edge networks by leveraging blockchain and smart contract technologies. Moreover, the authors presented a reputation-based data-sharing protocol that fosters high-quality data exchange among vehicles. Wang et al. [13] designed a privacy-preserving scheme named BalancePIC, which works towards achieving a balance among user privacy, data integrity, and computation overhead in edge-assisted IoT devices. Jan et al. [14] presented an end-to-end encryption system called SmartEdge, which uses a lightweight symmetric encryption method for a smart city application, ensuring dependable data transmission for facilitating communication between smart devices, edge nodes, and cloud data centers. Gai et al. [15] suggested a permissioned blockchain edge paradigm for smart grid edge-assisted IoT networks that solve privacy and energy security by merging edge computing with blockchain technology. Liu et al. [16] used secret sharing and blockchain to design a cooperative group authentication scheme providing a data-tracking function in vehicular edge computing. Lu et al. [17] designed a novel group signature scheme to realize anonymous authentication. Using the proposed group signature scheme, they presented a blockchain-based cloud storage protocol for sensors in industrial IoT. In the scheme by Yang et al. [18], an efficient anonymous certificateless aggregation signcryption scheme was designed to achieve a privacy-preserving aggregation authentication scheme for a safety warning system in fog-cloud based vehicular ad hoc networks. Bernard et al. [19] proposed a robust mutual authentication protocol utilizing the visual cryptography technique. Aiming to protect users' identity privacy while authenticating their identity in IoT applications, Yang et al. [20] introduced Zero-Cerd by designing a self-blindable anonymous authentication system based on blockchain and incorporating a dynamic accumulator scheme. Existing works either cannot support message linkability to resist Sybil attacks or have heavy communication and computing overhead. As shown in Table 1, we present a comparison of properties between the proposed MLGS scheme and relevant existing works.

**Table 1.** Comparison of security and privacy properties.

| | **Properties** | | | | |
|---|---|---|---|---|---|
| **Scheme** | **Anonymity** | **Traceability** | **Message-Linkability** | **Dynamics** | **Open Efficiency** |
| [9] | ✓ | ✓ | ✓ | ✓ | $\mathcal{O}(n)$ |
| [17] | ✓ | ✓ | × | ✓ | $\mathcal{O}(1)$ |
| [21] | ✓ | × | × | ✓ | - |
| [22] | ✓ | ✓ | × | × | $\mathcal{O}(n)$ |
| Ours | ✓ | ✓ | ✓ | ✓ | $\mathcal{O}(1)$ |

## 3. Preliminaries

This part recalls the cryptographic building blocks of our work, namely bilinear pairing, BBS signatures, ElGamal encryption, and group signatures. Descriptions of notations used in this paper are presented in Table 2.

**Table 2.** Notations and descriptions.

| Notations | Descriptions |
|---|---|
| $\lambda$ | A security parameter |
| $p$ | A large prime |
| $\mathbb{G}_1, \mathbb{G}_2$ | Two additive cyclic groups of prime order $p$ |
| $\mathbb{G}_T$ | A multiplicative cyclic group of prime order $q$ |
| $g, \hat{g}$ | Generators of $\mathbb{G}_1, \mathbb{G}_2$, respectively |
| $\hat{e}$ | A bilinear map |
| $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3$ | Secure hash functions: $\{0,1\}^* \to \mathbb{Z}_q$, $\{0,1\}^* \to \mathbb{G}_1$, $\{0,1\}^* \to \{0,1\}^l$ |
| $\vec{a}$ | A vector of $\{a[0], \ldots, a[n]\}$ |
| $mpk/msk$ | The master public/secret key |
| $gpk/gsk$ | The user's public/secret key |
| $\sigma_c$ | The membership certificate |
| $\sigma$ | The group signature |
| $Q$ | The message-link identifier |

### *3.1. Notions*

**Definition 1** (Bilinear pairing)**.** *An efficiently computable function $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, known as a bilinear map, is established for prime-order groups $\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_T$. The bilinear map satisfies both requirements:*

1. Bilinearity*: for all $\phi \in \mathbb{G}_1, \psi \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$, there is $e(\phi^a, \psi^b) = e(\phi, \psi)^{ab}$.*
2. Non-triviality*: for all generators $g \in \mathbb{G}_1 \backslash \{1_{\mathbb{G}_1}\}$ and $\hat{g} \in \mathbb{G}_2 \backslash \{1_{\mathbb{G}_2}\}$, there is $e(g, \hat{g}) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}}$ is the identity element of $\mathbb{G}$.*

*We employ a type-3 bilinear pairing, where $\mathbb{G}_1 \neq \mathbb{G}_2$, and there exists no efficient computable homomorphism between them.*

**Definition 2** (DDH assumption)**.** *Consider a cyclic group $\mathbb{G}$ with a prime order $p$ and a generator $g$. The decisional Diffie–Hellman (DDH) assumption means that it is computationally infeasible for any polynomial-time algorithm to differentiate between $ab$ and $c$ when given $g^a$, $g^b$, and $g^c$, where $a, b, c$ are unknown values in $\mathbb{Z}_p$. The probability of distinguishing $ab \stackrel{?}{=} c$ is negligible.*

**Definition 3** (q-SDH assumption)**.** *Consider a cyclic group $\mathbb{G}$ with a prime order $p$ and a generator $g$. The q-strong Diffie–Hellman (q-SDH) assumption means that the probability for any polynomial-time algorithm to compute $(s, g^{\frac{1}{a+s}})$ for any $s \in \mathbb{Z}_p$ when given $g, g^a, \ldots, g^{a^q}$ is negligible, where $a$ is an unknown value in $\mathbb{Z}_p$.*

**Definition 4** (Weakened CPA-full-anonymity [23])**.** *The weakened CPA-full-anonymity allows an adversary to access users' secret keys and certificates, except for those associated with the challenge users. Formally, the weakened CPA-full-anonymity game between the challenger $\mathcal{B}$ and an adversary $\mathcal{A}$ is shown as follows:*

***Setup****: $\mathcal{B}$ generates public parameters and the master public–secret key pair. The honest user list $\mathbb{L}_{honest}$ and corrupt user list $\mathbb{L}_{corrupt}$ are prepared from $\mathcal{A}$. $\mathcal{B}$ sends the public parameters and master public key to $\mathcal{A}$.*

***Queries****: $\mathcal{A}$ makes adaptive queries to $\mathcal{B}$: (1) Issue—$\mathcal{A}$ acts as a compromised user, querying $\mathcal{B}$ for certificate $\sigma_c$. (2) Corrupt—$\mathcal{A}$ queries the private key and certificate of an honest user $uid_i$. $\mathcal{B}$ returns $(gsk_i, \sigma_c)$ and includes $uid_i$ in the $\mathbb{L}_{corrupt}$ list. (3) Sign—$\mathcal{A}$ queries a signature $\sigma_i$ for honest user $uid_i$ with the message $m$. $\mathcal{B}$ computes $\sigma_i$ and responds. (4) Hash—$\mathcal{A}$ queries a hash. $\mathcal{B}$ responds if in the hash list $\mathbb{L}_{hash}$, otherwise, it generates $c_i$ randomly and updates $\mathbb{L}_{hash}$. The hash function is modeled as a random oracle.*

***Challenge**: During this stage, $\mathcal{A}$ picks a message $m^*$ and two honest users, $uid_0^*$ and $uid_1^*$, with $uid_0^*, uid_1^* \in \mathbb{L}_{honest}$ and $uid_0^*, uid_1^* \notin \mathbb{L}_{honest}$. $\mathcal{B}$ randomly picks $b \in \{0,1\}$, and constructs a challenge signature $\sigma_b^*$ with $(gsk_b^*, \sigma_{c_b}^*)$. Subsequently, $\mathcal{B}$ furnishes $\mathcal{A}$ with $\sigma_b^*$.*

***Guess**: $\mathcal{A}$ guesses $b' \in \{0,1\}$ of $uid_b^*$, and wins if $b' = b$.*

### 3.2. BBS Signatures

We utilize a shorter version [10] of BBS+ signatures in our group signature. The initial proposal of BBS signatures was put forth by Boneh, Boyen, and Shacham [24]. The devised version of BBS+ signatures [25] is well-suited for use in many privacy-preserving application scenarios, thanks to the efficiency of their algebraic structures in facilitating proof of knowledge for message–signature pairs that allow for partial disclosure. This includes four probabilistic polynomial time (PPT) algorithms, which are listed as follows:

- Setup$(1^\lambda) \rightarrow par$: Given a security parameter $1^\lambda$ as input, it it outputs a set of public parameters $par = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g, \vec{h}, \hat{g}, e)$, where $\vec{h} = \{h[0], \ldots, h[\ell]\} \in \mathbb{G}_1^\ell$ is a vector of generators in $\mathbb{G}_1$.
- KeyGen$(par) \rightarrow (pk, sk)$: Given $par$ as input, it randomly picks $sk = x \in \mathbb{Z}_p$ and it returns a key pair $\left(sk, pk = \hat{g}^{sk}\right)$.
- Sign$(sk, \vec{m}) \rightarrow \sigma$: Given a secret key sk and a message vector $m$ as input, it computes $C = g\prod_{i=0}^{\ell} h[i]^{m[i]}$, randomly picks $\gamma \in \mathbb{Z}_p$ and then returns $\sigma = (A, \gamma)$, where $A = C^{\frac{1}{x+\gamma}}$.
- Verify$(pk, \sigma = (A, \gamma)) \rightarrow 0/1$: Given a public key $pk$ and a signature $\sigma$, it computes $C = g\prod_{i}^{\ell} h[i]^{m[i]}$ and outputs $e(A, \hat{g}^\gamma pk) = e(C, \hat{g})$.

### 3.3. ElGamal Encryption

Subsequently, we recall a definition of the ElGamal encryption scheme [26]. In our scheme, the actual identity of a user is concealed in the ciphertext. When necessary, the identity can be exposed. It is made up of the following PPT algorithms:

- Setup$(1^\lambda) \rightarrow par$: Given a security parameter $1^\lambda$ as the input, it outputs a set of public parameters $par = \{p, \mathbb{G}_2, \hat{g}\}$.
- KeyGen$(par) \rightarrow (pk, sk)$: Given $par$ as the input, it outputs a secret–public key pair $(sk \in \mathbb{Z}_p, pk = \hat{g}^{sk})$.
- Enc$(pk, msg) \rightarrow (ct_1, ct_2)$: Given a public key pk and a message $msg$ as input, it randomly chooses a scalar $r \in \mathbb{Z}_p^*$ and returns $(ct_1, ct_2)$ as the ciphertext, where $ct_1 = \hat{g}^r, ct_2 = pk^r msg$.
- Dec$(sk, ct_1, ct_2) \rightarrow msg$: Given a secret key $sk$ and ciphertexts $ct_1, ct_2$, it returns the message $msg = ct_2 ct_1^{-sk}$.

### 3.4. Group Signatures

The group signature, as proposed by Chaum and van Heyst [27], allows for anonymous authentication while maintaining accountability to a service. In this system, a designated group manager oversees a group of users who have the ability to generate anonymous signatures representing the group. Essentially, anyone can verify that a signature originates from one of the group members. Except for the group manager, it is impossible to ascertain the actual originator of the signature. A typical group signature scheme typically comprises six algorithms that are executed with polynomial time complexity.

- GSetup$(1^\lambda) \rightarrow par$: Given a security parameter $1^\lambda$ as input, it outputs public parameters $par$.
- IKeyGen$(par) \rightarrow (mpk, msk)$: Given the public parameters $par$ as input, it outputs $(mpk, msk)$, where $(mpk, msk)$ is the master public–secret key pair.
- UKeyGen$(par, mpk) \rightarrow (gpk, gsk)$: Given the public parameter $par$ and master public key $mpk$ as input, it outputs the user's public–secret key pair $(gpk, gsk)$.

- Issue$(gpk, mpk, msk) \rightarrow \sigma_c$: Given a public key $gpk$ of a user and the master public–secret key pair $(mpk, msk)$ as input, it outputs a membership certificate $\sigma_c$ for the user.
- GSign$(msg, mpk, gpk, gsk, \sigma_c) \rightarrow \sigma$: Given a message $msg$, a master public key $mpk$, the public–secret key pair $(gpk, gsk)$, and membership certificate $\sigma_c$ of the user as input, it returns a group signature $\sigma$.
- GVerify$(msg, mpk, \sigma) \rightarrow b$: Given a message $msg$, the master public key $mpk$, and a group signature $\sigma$ as input, the $b$ is set as 1 if $\sigma$ is valid, and the $b$ is set as 0 otherwise. Finally, it returns $b$.
- GOpen$(\sigma, msk) \rightarrow gpk$: Given a group signature $\sigma$ and the master secret key $msk$, a user's identity $gpk$ is returned.

## 4. Problem Overview

### *4.1. System Architecture*

As indicated in Figure 2, the system architecture employed in our work encompasses a trusted authority (TA), edge nodes (ENs), and devices. Edge nodes use the network resources at the edge of the network to serve as intermediaries, to realize the localization of IoT services and data storage. The roles and functions of each component are delineated as follows.

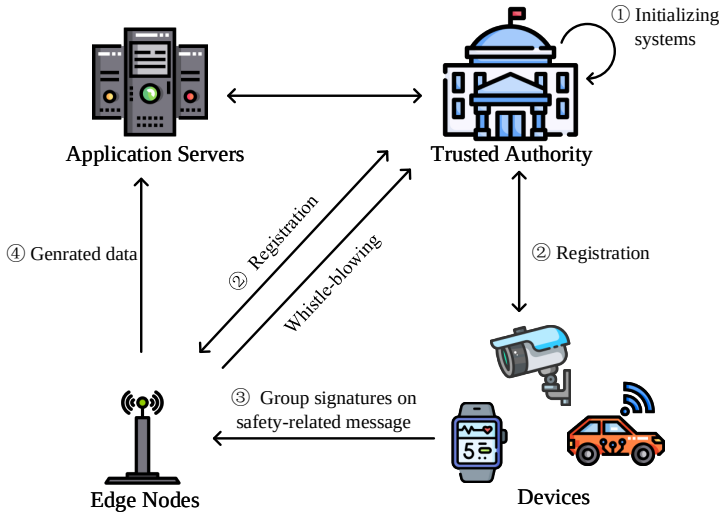


**Figure 2.** System architecture.

- **TA**: TA is located in the cloud layer, which is far away from data centers with no mobility. It serves as a reliable entity, offering system-wide monitoring and centralized control services. It stores safety warning data obtained from sensors and performs data processing tasks that surpass the capabilities of edge nodes. TA is accountable for generating and periodically updating public parameters, as well as issuing certificates to devices. Moreover, TA can trace malicious devices.
- **Edge nodes**: Edge nodes in the edge layer are highly distributed with mobility support. They can be macro/micro base stations or Wi-Fi hotspots. Edge nodes facilitate uplink and downlink data transmission by performing preprocessing tasks, thereby reducing communication overhead and caching functionalities to support IoT applications. Edge nodes gather safety-related message–signature pairs from the devices. After authenticating and analyzing the received data, edge nodes transmit the genuine data to the TA and application servers. They communicate with the TA via wired secure connections.
- **Devices**: Two categories of IoT devices [6] exist: fixed devices, including environment sensors that are pre-installed in specific areas, and mobile devices, which are personally carried by their owners (such as smartphones, smartwatches, and smart vehicles). Smart devices are embedded with a range of sensors that collect desired safety-related

message data from the environment and transmit the generated data to the TA and application servers through relayed edge nodes. In conclusion, devices anonymously sign and endorse the safety-related messages to be submitted, and then send them to edge nodes.

### *4.2. Threat Model*

Firstly, the TA is considered to be completely trustworthy in our assumptions. Security threats in safety warning systems can arise from two aspects, internal and external adversaries. Overall, internal threats are typically posed by edge nodes and devices. Edge nodes are generally considered semi-trusted, implying that they will faithfully execute the entire process but may have an interest in obtaining privacy-related information from devices. Devices are assumed to be malicious, exhibiting curiosity regarding the content of messages transmitted by neighboring edge devices and/or the identities of these devices. Moreover, they may also impersonate other devices to propagate false messages that can lead to severe accidents. The threats posed by external attackers resemble those posed by malicious devices.

### *4.3. Design Goals*

This work achieves the following security properties:

- **Authentication**. It guarantees the authenticity of a received message, confirming its origin from a valid edge device and remains unaltered during transmission.
- **Anonymity**. Anonymity implies that both internal and external adversaries are unable to deduce the actual identity of an edge device based on transmitted data.
- **Traceability**. Traceability refers to the ability of a TA to trace the identity of malicious or misbehaving users. In cases where a malicious device disseminates a fraudulent message, its identity can be efficiently tracked and identified by the TA. Other entities lack the authority to identify participants.
- **Message linkability**. Message linkability implies that, when presented with two signatures on an identical message, we deduce that these signatures originate from the same group member, although it remains unclear exactly which one.
- **Devices dynamics**. Following the system initialization, an edge device has the flexibility to enroll in the system at any given time. That is to say, the devices are not stationary in the system initialization phase and can vary over time throughout the whole system.

## 5. The Detail of Construction

### *5.1. The Proposed Linkability Group Signatures Scheme*

This section introduces a new message-linkable group signature scheme, which is formed by eight algorithms, namely ***Setup***, ***IKeyGen***, ***UKeyGen***, ***Issue***, ***GSign***, ***GVerify***, ***GOpen***, and ***Link***. The details are described as follows.

- $\mathrm{GSetup}(1^\lambda) \rightarrow par$: Given the security parameter $1^\lambda$ as input, it outputs public parameters $par = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g, h, \hat{g}, e)$.
- $\mathrm{IKeyGen}(par) \rightarrow (mpk, msk)$: Given the public parameter $par$ as input, it randomly picks $x \in \mathbb{Z}_p$, sets $msk := x, mpk := (mpk_1, mpk_2) = (g^x, \hat{g}^x)$, and generates a master public–secret key pair $(mpk, msk)$.
- $\mathrm{UKeyGen}(par, mpk) \rightarrow (gpk, gsk)$: Given the public parameter $par$ and master public key $mpk$ as input, it picks $gsk := y \in \mathbb{Z}_p$ (in random), computes $gpk = h^{gsk}$, and generates a public–secret key pair $(gpk, gsk)$.
- $\mathrm{Issue}(gpk, mpk, msk) \rightarrow \sigma_c$: Given the public key $gpk = h^y$ of a user and the master public–secret key pair $(mpk, msk)$ as input, it selects $\gamma \in \mathbb{Z}_p$ randomly, and computes $A = (gh^y)^{\frac{1}{x+\gamma}}$.

- GSign$(msg, mpk, gpk, gsk, \sigma_c) \rightarrow \sigma$: Given a message $msg$, the master public key $mpk$, the public–secret key pair $(gpk, gsk)$, and membership certificate $\sigma_c$ of the user as input, it executes the following:
  (a) Sets $D = gh^y$.
  (b) Randomly chooses $\alpha, \beta \in \mathbb{Z}_p$ and computes

$$\bar{A} = A^{\alpha}, \quad \bar{B} = D^{\alpha}\bar{A}^{-\gamma}, \quad E = D^{\alpha}h^{-\alpha}$$
$$C_1 = g^{\beta}, \quad C_2 = mpk_1^{\beta}gpk, \quad Q = \mathcal{H}_2(msg)^y$$

  where $\bar{B} = \bar{A}^{-\gamma}h^{\alpha}E, g = E^{-\alpha}h^{\mu}, \mu = -y - \alpha^2$.
  (c) Randomly picks $r_\alpha, r_\beta, r_\gamma, r_\mu, r_y \in \mathbb{Z}_p$, and computes

$$T_1 = \bar{A}^{r_\gamma}h^{r_\alpha}, \quad T_2 = E^{-r_\alpha}h^{r_\mu}, \quad T_3 = g^{r_\beta},$$
$$T_4 = mpk_1^{r_\beta}h^{r_y}, \quad T_5 = \mathcal{H}_2(msg)^{r_y}.$$

  (d) Computes the challenge $c = \mathcal{H}_1(msg \parallel \bar{A} \parallel \bar{B} \parallel C_1 \parallel C_2 \parallel E \parallel Q \parallel T_1 \parallel T_2 \parallel T_3 \parallel T_4 \parallel T_5)$.
  (e) Computes $s_\alpha = r_\alpha - c \cdot \alpha, s_\beta = r_\beta - c \cdot \beta, s_\gamma = r_\gamma + c \cdot \gamma, s_\mu = r_\mu - c \cdot \mu, s_y = r_y - c \cdot y \bmod p$.
  (f) Returns the signature $\sigma = (\bar{A}, \bar{B}, E, C_1, C_2, Q, c, r_\alpha, r_\beta, r_\gamma, r_y)$.
- GVerify$(msg, mpk, \sigma) \rightarrow b$: Given a message $msg$, the master public key $mpk$, and a group signature $\sigma$ as input, it returns a bit $b \in \{0, 1\}$, and works as follows:
  (a) Parses $\sigma = (\bar{A}, \bar{B}, E, C_1, C_2, Q, c, s_\alpha, s_\beta, s_\gamma, s_\mu, s_y)$.
  (b) Checks if the equation $e(\bar{A}, mpk_2) = e(\bar{B}, \hat{g})$ holds. If so, it continues. Else, it returns 0.
  (c) Computes

$$T_1' = (\bar{B}/E)^c\bar{A}^{s_\gamma}h^{s_\alpha}, \quad T_2' = g^cE^{-s_\alpha}h^{s_\mu},$$
$$T_3' = C_1^c g^{s_\beta}, \quad T_4' = C_2^c mpk^{s_\beta}h^{s_y},$$
$$T_5' = Q^c\mathcal{H}_2(msg)^{s_y}.$$

  (d) Verifies if $\mathcal{H}_1(msg \parallel \bar{A} \parallel \bar{B} \parallel C_1 \parallel C_2 \parallel E \parallel Q \parallel T_1' \parallel T_2' \parallel T_3' \parallel T_4' \parallel T_5') = c$. If the aforementioned equation is true, it returns 1. Conversely, if the equation is false, it returns 0, indicating that the signature fails the verification.
- GOpen$(\sigma, msk) \rightarrow gpk$: Given a group signature $\sigma$ and the master secret key $msk$, it generates a real identity $gpk$ of the signature generator by computing the following equation:

$$gpk = C_2/C_1^x.$$

- Link$(\sigma, \sigma', msg) \rightarrow 0/1/\perp$: Given two group signature $\sigma = (\bar{A}, \bar{B}, E, C_1, C_2, Q, c, r_\alpha, r_\beta, r_\gamma, r_y), \sigma' = (\bar{A}', \bar{B}', E', C_1', C_2', Q', c', r_\alpha', r_\beta', r_\gamma', r_y')$, and a message $msg$, if GVerify $(msg, mpk, \sigma) = 0$, or GVerify$(msg, mpk, \sigma') = 0$, it returns $\perp$, which means an error occurred, otherwise, it further checks if $Q = Q'$. If $Q = Q'$, it returns 1, otherwise, it returns 0.

**Correctness**. The correctness of the designed group signature scheme is demonstrated by substantiating the following facts:

$$
\begin{aligned}
T_1' &= (\bar{B}/E)^c \bar{A}^{s_\gamma} h^{s_\alpha} = (\bar{B}/E)^c \bar{A}^{r_\gamma + c\cdot\gamma} h^{r_\alpha - c\cdot\alpha} \\
&= (\bar{B}/E)^c \bar{A}^{r_\gamma} h^{r_\alpha} (\bar{A}^{-\gamma} h^{\alpha})^{-c} = \bar{A}^{r_\gamma} h^{r_\alpha} = T_1, \\
T_2' &= g^c E^{-s_\alpha} h^{s_\mu} = g^c E^{c\cdot\alpha - r_\alpha} h^{r_\mu - c\cdot\mu} \\
&= g^c E^{-r_\alpha} h^{r_\mu} (E^{-\alpha} h^{\mu})^{-c} = E^{-r_\alpha} h^{r_\mu} = T_2, \\
T_3' &= C_1^c g^{s_\beta} = C_1^c g^{r_\beta - c\cdot\beta} = C_1^c g^{r_\beta} C_1^{-c} = g^{r_\beta} = T_3, \\
T_4' &= C_2^c mpk^{s_\beta} h^{s_y} = C_2^c mpk^{r_\beta - c\cdot\beta} h^{r_y - c\cdot y} \\
&= C_2^c mpk^{r_\beta} h^{r_y} (mpk^{\beta} h^{y})^{-c} = mpk^{r_\beta} h^{r_y} = T_4, \\
T_5' &= Q^c \mathcal{H}_2(msg)^{s_y} = Q^c \mathcal{H}_2(msg)^{r_y - c\cdot y} \\
&= Q^c \mathcal{H}_2(msg)^{r_y} (H_2(msg)^{y})^{-c} = \mathcal{H}_2(msg)^{r_y} = T_5.
\end{aligned}
$$

Therefore, the proof of the correctness is completed.

### *5.2. Proposed Authentication*

Utilizing the aforementioned group signatures scheme as a foundation, we develop our CPPA scheme for a safety warning system in the edge-assisted Internet of Things. In this scheme, each registered device possessing a valid certificate $\sigma_c$ is granted membership in the authorized group. The membership certificate $\sigma_c$ allows valid group of members to sign and submit safety-related warning messages. Our scheme comprises five distinct phases, i.e., system initialization, registration, message delivery, verify and decrypt, and trace.

#### 5.2.1. System Initialization

Specifically, the TA initiates the whole scheme in this phase by

1. Choosing a security parameter $\lambda$, and then running the algorithm GSetup$(1^\lambda)$ to generate the parameters $par = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g, h, \hat{g}, e)$.
2. Running the IKeyGen$(par)$ algorithm to produce the TA's master key pair $(mpk, msk)$.
3. Picking two secure cryptographic hash functions $\mathcal{H}_1 : \{0,1\}^* \to \mathbb{Z}_q, \mathcal{H}_2 : \{0,1\}^* \to \mathbb{G}_1, \mathcal{H}_3 : \{0,1\}^* \to \{0,1\}^l$.
4. Publishing the public parameters $par = (par, mpk, \mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3)$.

#### 5.2.2. Registration

In this phase, each device within the system must undergo registration with the TA in order to obtain its respective public–secret key pair. Let the device be $D_i$. It includes the following steps:

1. $D_i$ first runs the UKeyGen$(par, mpk)$ algorithm to produce its public/secret key pair $(gpk, gsk)$.
2. TA runs the Issue$(gpk, mpk, msk)$ algorithm to generate the membership certificate $\sigma_c = (A, \gamma)$ for $D_i$, then secretly sends the $\sigma_c$ to $D_i$ through a secure channel.
3. Once $D_i$ receives its membership certificate $\sigma_c = (A, \gamma)$ from the TA, it checks if $A \neq 1_{\mathbb{G}_1}$ and $e(A, \hat{g}^\gamma mpk_2) = e(gh^y, \hat{g})$. $D_i$ sets $\sigma_c := (A, \gamma)$ as its membership certificate if all the above equations hold; otherwise, it discards it.

Each edge node also needs to register in the system. We denote an edge node as $EN$. It executes the following steps to register itself. First, it picks a randomizer $ssk \in \mathbb{Z}_p$, and then computes $spk = g^{ssk}$, where $(spk, ssk)$ is set as the public–secret key of the edge node. Next, it proves to the TA that it knows the knowledge of the public key $spk$ using Schnorr's protocol [28]. After successfully convincing the TA, the TA will generate a PKI-based certificate for the edge node $EN$.

#### 5.2.3. Message Delivery

During this phase, the authorized device $D_i$ can anonymously transmit safety-related message data to the edge nodes nearby; we require each edge node to periodically broadcast its public key $spk_j$. This phase encompasses the following three steps:

1. $D_i$ picks random numbers $\beta_i \in \mathbb{Z}_q$ and then encrypts the safety-related *data* as $CT = data \oplus \mathcal{H}_3(spk_j^{\beta_i})$.
2. $D_i$ will send a safety-related message $m_i = CT||ts_i$ to the edge nodes, where $ts_i$ represents a timestamp. It executes the algorithm $\mathrm{GSign}(m_i, mpk, gpk_i, gsk_i, \sigma_{c_i})$ to generate a message-linkable group signature $\sigma_i = (\bar{A}_i, \bar{B}_i, E_i, C_{1i}, C_{2i}, Q_i, c_i, r_{\alpha i}, r_{\beta_i}, r_{\gamma_i}, r_{y_i})$. It is worth noting that the random number $\beta_i$ used in the generation of $C_{1i}, C_{2i}$ is identical to the number $\beta_i$ in the ciphertext $CT$.
3. Finally, $D_i$ sends a tuple $(m_i, \sigma_i)$ to the nearby edge node $EN_j$ with the public key $spk_j$.

#### 5.2.4. Verify and Decrypt

When the edge node $EN_j$ receives multiple message tuples $(m_i, \sigma_i)_{i\in[n]}$, it first runs the algorithm $\mathrm{GVerify}(m_i, mpk, \sigma_i)$ to check the validity for all $i \in [n]$, where $\sigma_i = (\bar{A}_i, \bar{B}_i, E_i, C_{1i}, C_{2i}, Q_i, c_i, r_{\alpha i}, r_{\beta_i}, r_{\gamma_i}, r_{y_i})$; whichever one fails to pass the verification is discarded. For tuples with the same messages $m$ that pass verification, $S_j$ executes the algorithm $\mathrm{Link}(\sigma_i, \sigma_j, m)$ to check whether a device broadcasts a message more than once. If it returns 1, then one of them will be discarded as invalid. We suppose that $(m_u, \sigma_u)$ is valid. $S_j$ decrypts $CT$ by computing $data = CT \oplus \mathcal{H}_3(C_{1u}^{ssk_j})$ to the safety-related *data*.

#### 5.2.5. Trace

When a malicious message $(m_l, \sigma_l)$ generated by a device $D_l$ is found by edge node $S_j$, edge node $S_j$ will relay the malicious message $m_l$ to the TA to trace the real identity of the device $D_l$. During this phase, the TA undertakes the task of unveiling the true identity of a malevolent device. The TA executes the algorithm $\mathrm{GOpen}(\sigma_l, msk)$ to recover the public key $gpk_l$ of the device $D_l$.

## 6. Analysis and Experimental Findings

This section demonstrates security guarantees and the experimental performance.

### *6.1. Security Analysis of Our MLGS Scheme*

We demonstrate that the proposed MLGS scheme can provide weakened CPA-full-anonymity [23] and full traceability.

**Theorem 1.** *Our MLGS scheme is weakened CPA-full-anonymous if (1) the DDH assumption holds in $\mathbb{G}_1$, (2) the BBS+ signature is unlinkable, (3) the ElGamal scheme is CPA-secure, and (4) the SPK is simulation sound, zero-knowledge, and online-extractable.*

**Proof. Setup**: Given $(g, g^a, g^b, Z)$ as an instantiation of the DDH problem, we assume that $x$ is equal to $a$. $\mathcal{B}$ sets the master public key as $mpk = (g, \hat{g}, mpk_1 = g^a)$. The honest user list $\mathbb{L}_{honest}$ and corrupt user list $\mathbb{L}_{corrupt}$ are retrieved from $\mathcal{A}$. $\mathcal{B}$ generates key pairs $(gsk_i, gpk_i)$ for each honest user $uid_i$ by running $\mathsf{UKeyGen}(par, mpk)$ and creates certificates using $r_i \in \mathbb{Z}_p$ values.

**Queries**: $\mathcal{A}$ makes the following queries: (1) Issues query—$\mathcal{A}$ requests a certificate $\sigma_c$ for corrupt user $uid_i$ from $\mathcal{B}$. (2) Corrupt query—$\mathcal{A}$ queries the private key and certificate of an honest user $uid_i$. $\mathcal{B}$ returns $(gsk_i, \sigma_c)$ and updates $\mathbb{L}_{corrupt}$. (3) Sign query—$\mathcal{A}$ queries a signature $\sigma_i$ for honest user $uid_i$ with message $m$. $\mathcal{B}$ computes $\sigma_i$ and responds. (4) Hash query—$\mathcal{A}$ queries a hash. $\mathcal{B}$ responds if in the hash list $\mathbb{L}_{hash}$, otherwise, it generates $c_i$ randomly and updates $\mathbb{L}_{hash}$. The cryptographic hash function $\mathcal{H}_1$ is modeled as a random oracle.

**Challenge**: $\mathcal{A}$ presents a challenge $\{uid_0^*, uid_1^*, m^*\}$, where $uid_0^*, uid_1^* \in \mathbb{L}_{honest}$. $\mathcal{B}$ sets values $C_1 = g^b, C_2 = gpk_b^* \cdot Z$. Leveraging the zero-knowledge of SPK, we are capable of simulating the elements $(\bar{A}), \bar{B}, E, Q$. $\mathcal{B}$ randomly selects $c, r_\alpha, r_\beta, r_\gamma, r_y \in \mathbb{Z}_p$. Then, $\mathcal{B}$ updates $\mathbb{L}_{hash}$ and returns $\sigma_b^* = (\bar{A}, \bar{B}, E, C_1, C_2, Q, c, r_\alpha, r_\beta, r_\gamma, r_y)$.

**Guess**: $\mathcal{A}$ guesses $b' \in \{0,1\}$ of $uid_b^*$ . If $b' = b$, $\mathcal{B}$ determines $Z = g^{ab}$, otherwise, $Z \neq g^{ab} \in \mathbb{G}_1$.

To begin with, as $Z$ is selected uniformly at random from $\mathbb{G}_1$, the resulting element $C_2$ is also uniformly distributed within $\mathbb{G}_1$. Secondly, considering that the randomnesses are uniformly chosen from $\mathbb{Z}_p^*$, it follows that $\bar{A}$ and $\bar{B}$ are uniformly distributed over $\mathbb{G}_1$. Thirdly, the zero-knowledge attribute of SPK ensures the concealment of the witnesses. Thus, it can be deduced that $\sigma^*$ conceals the information tied to *uid*, thereby also hiding $b$. □

**Theorem 2.** *Our MLGS scheme is fully traceable if the q-SDH assumption holds.*

**Proof.** Suppose that an adversary $\mathcal{A}$ can win the traceability game with negligible probability, then we can build an algorithm $\mathcal{B}$ to break the q-SDH assumption. Our proof closely aligns with the modified proof of the BBS+ signature unforgeability outlined in [10]. Due to space limitations, we do not expand here in detail, and readers are advised to see ref. [10] for a more detailed explanation. □

### *6.2. Security Analysis of Our CPPA Scheme*

1. **Authentication**. As BBS signatures are unforgeable under q-SDH assumptions, no PPT adversary can forge a valid certificate without the secret. Also, from the soundness of knowledge signatures, we know that any PPT adversary cannot forge a valid group signature without a valid membership certificate. Thus, our scheme guarantees authentication property.
2. **Anonymity**. The devices employ an anonymous method to transmit safety-related messages to the edge nodes. Each signature will be randomized using random numbers to ensure that the identity information of the real signer remains undisclosed. Thereby, the anonymity property is satisfied.
3. **Traceability**. The TA can reveal the actual identity of malicious devices if needed. When the TA receives a group signature $\sigma_k$, which is generated by a misbehaved device from the edge nodes, the TA runs the algorithm $\mathrm{GOpen}(\sigma_k, msk)$ to obtain the true identity.
4. **Message linkability**. After receiving two valid signatures $\sigma_i, \sigma_j$ on message $m$, edge nodes can check whether $\mathrm{Link}(\sigma_i, \sigma_j, m) = 1$. If it holds, edge nodes can conclude that if Sybil attacks exist, then they will only retain one of the two signatures. This property ensures that malicious devices can always be identified. On the one hand, if a malicious device signs a wrong message, a trusted authority can track it. On the other hand, if a device tends to deceive by endorsing the same message multiple times, then other entities can easily link multiple signatures to the same device and, thus, discard or transfer them to the trusted authority for traceability. Therefore, our scheme can protect against Sybil attacks.
5. **Devices dynamics**. It is evident that devices have the flexibility to enroll the system at any point in time following system initialization. Additionally, the total number of devices is not predetermined. Moreover, during the system initialization process, the TA solely generates randomness and public parameters, eliminating the need for trust in this process.

### *6.3. Experiment and Performance*

We evaluate our work by examining its complexity in terms of theoretical comparison and practical implementation. In the theoretical analysis, we compare our linkable group signatures with the two most related schemes, e.g., by Wu et al. [9] and Li et al. [29], in terms of communication and computational complexity. Furthermore, we implement our scheme to measure the signature length and evaluate the execution times of the signing and verification algorithms.

*Experimental Environments.* In our proposed scheme, the message sender is a device, while the recipient is an edge node. Typically, the computational capabilities of devices are more resource-constrained. Therefore, we simulate the sender's computational environment using the Raspberry Pi platform. Correspondingly, the computational environment of edge nodes and the trusted authority (TA) is on a personal computer (PC) platform. We conduct tests on these two platforms to measure the computation times of the main operations involved. The PC is a Dell laptop running the Ubuntu 18.04 operating system, equipped with an i7-10700 Processor and 16 GB RAM. Raspberry Pi runs the Linux Raspberry Pi 5.10.17 operating system, equipped with a Cortex-A72(ARM 8) 1.5 GHZ processor and 4 GB RAM. For the implementation of cryptographic primitives, we utilize the Relic Library [30]. We choose a 381-bit Barreto–Lynn–Scott (BLS) curve of embedding degree 12.

*Theoretical analysis.* We evaluate the time costs of the main cryptographic operations and the sizes of the used group elements (see Table 3). The comparisons between our linkable group signature scheme and the most relevant schemes [9,29] are shown in Table 4. We only consider the time-consuming operations, i.e., the point multiplication on the group, bilinear pairing, and hash point. Among the three schemes, our scheme is slightly worse than the one in ref. [9] in signing time cost, verifying time cost, and signature length, but the tracing time of [9] has a linear relationship with the number of group members $n$, while our tracing only requires a constant amount of time.

**Table 3.** Experimental evaluation based on the Relic Library.

| Notions | Description | Value (ms/Bytes) | |
|---|---|---|---|
| | | PC | Raspberry Pi |
| $T_{\mathbb{G}_1}$ | Time of a point multiplication on $\mathbb{G}_1$ | 0.088 | 1.878 |
| $T_{\mathbb{G}_2}$ | Time of a point multiplication on $\mathbb{G}_2$ | 0.171 | 5.659 |
| $T_{\mathbb{G}_T}$ | Time of a point multiplication on $\mathbb{G}_T$ | 0.264 | 11.789 |
| $T_{par}$ | Time of a bilinear pairing | 0.700 | 14.460 |
| $T_{htp}$ | Time of the hash function to point | 0.129 | 4.812 |
| $\lvert\mathbb{G}_1\rvert$ | Length of an element in group $\mathbb{G}_1$ | 49 | 49 |
| $\lvert\mathbb{G}_2\rvert$ | Length of an element in group $\mathbb{G}_2$ | 97 | 97 |
| $\lvert\mathbb{G}_T\rvert$ | Length of an element in group $\mathbb{G}_T$ | 576 | 576 |
| $\lvert\mathbb{Z}_p\rvert$ | Length of an element in group $\mathbb{Z}_p$ | 32 | 32 |

**Table 4.** Theoretical comparison.

| Schemes | Sign a Signature | Verify a Signature | Open a Signature | Signature Length |
|---|---|---|---|---|
| Wu et al. [9] | $6T_{\mathbb{G}_1} + T_{htp}$ | $4T_{\mathbb{G}_1} + 3T_{par} + T_{htp}$ | $\mathcal{O}(n)$ | $4\lvert\mathbb{G}_1\rvert + 2\lvert\mathbb{Z}_p\rvert$ |
| Li et al. [29] | $19T_{\mathbb{G}_1} + T_{htp}$ | $14T_{\mathbb{G}_1} + 2T_{par} + T_{htp}$ | $\mathcal{O}(1)$ | $6\lvert\mathbb{G}_1\rvert + 7\lvert\mathbb{Z}_p\rvert$ |
| This work | $15T_{\mathbb{G}_1} + T_{htp}$ | $13T_{\mathbb{G}_1} + 2T_{par} + T_{htp}$ | $\mathcal{O}(1)$ | $6\lvert\mathbb{G}_1\rvert + 5\lvert\mathbb{Z}_p\rvert$ |

*Practical analysis.* The computational overhead analysis of our scheme is shown in Figure 3 by comparing with [9,29] in the running time of GSign, GVerify, and GOpen algorithms. We set the group member numbers to 20 and 40, respectively. Figure 3 shows the computation time in our scheme and in the other schemes; see Wu et al. [9] and Li et al. [29]. From Figure 3, we can see that the computation overheads of GSign and GVerify are 1.45 ms and 1.80 ms in our scheme, which are smaller than that of Li et al. [29] and larger than in ref. [9]. Moreover, our MLGS scheme saves about 14% more bandwidth than in [29] (454 bytes versus 518 bytes). However, the algorithm GOpen by Li et al. [29] grows linearly with the number of group members. Obviously, in real applications, the total number of devices is very large, and it will take a lot of overhead to trace malicious users in [29], which is not desirable in practice.

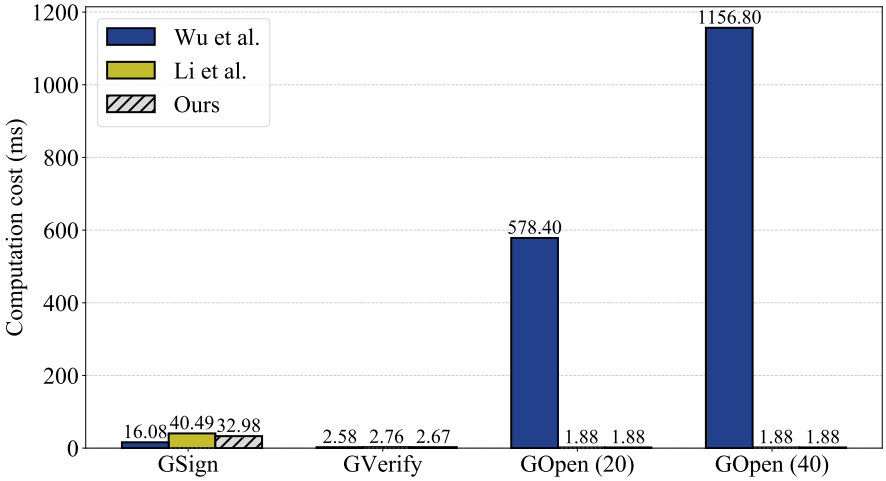


**Figure 3.** The comparative results of computational overhead [9,29].

## 7. Conclusions

This work presents an efficient conditional privacy-preserving authentication scheme for a safety warning system in the edge-assisted IoT paradigm. We design a linkable group signature scheme to resist Sybil attacks, facilitating the capability of a TA to track the group signature and disclose the authentic identity of the signature producer. Moreover, we informally discuss the security guarantees of our work. Eventually, we conduct the experimental evaluations to show the advantages of our scheme in real scenes. Consequently, the CPPA scheme we designed is highly appropriate for the safety warning system in edge-assisted IoT applications. Further studies will focus on the optimization of message-linkable group signatures to further improve the efficiency of conditional privacy-preserving authentication schemes.

**Author Contributions:** Conceptualization, J.L. and X.G.; methodology, J.L. and X.G.; software, N.H. and Y.L.; validation, Y.L., J.Z. and X.G.; formal analysis, N.H. and G.Z.; investigation, J.L., J.Z. and G.Z.; writing—original draft preparation, N.H. and J.L.; writing—review and editing, G.Z. and J.Z.; visualization, J.Z.; supervision, X.G.; funding acquisition, X.G. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the National Natural Science Foundation of China Youth Fund (grant no. 62203048) and the National Natural Science Foundation of China (grant no. 62073039).

**Data Availability Statement:** Not applicable.

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

## References

1. Gubbi, J.; Buyya, R.; Marusic, S.; Palaniswami, M. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Gener. Comput. Syst.* **2013**, *29*, 1645–1660. [CrossRef]
2. Brush, A.; Hazas, M.; Albrecht, J. Smart homes: Undeniable reality or always just around the corner? *IEEE Pervasive Comput.* **2018**, *17*, 82–86. [CrossRef]
3. Ali, I.; Hassan, A.; Li, F. Authentication and privacy schemes for vehicular ad hoc networks (VANETs): A survey. *Veh. Commun.* **2019**, *16*, 45–61. [CrossRef]
4. Gilchrist, A. *Industry 4.0: The Industrial Internet of Things*; Springer: Berlin/Heidelberg, Germany, 2016.
5. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [CrossRef]
6. Ni, J.; Lin, X.; Shen, X.S. Toward edge-assisted Internet of Things: From security and efficiency perspectives. *IEEE Netw.* **2019**, *33*, 50–57. [CrossRef]
7. Chen, Q.; Ye, A.; Zhang, Q.; Huang, C. A new edge perturbation mechanism for privacy-preserving data collection in iot. *Chin. J. Electron.* **2023**, *32*, 1–10. [CrossRef]

8. Camenisch, J.; Stadler, M. Efficient group signature schemes for large groups. In Proceedings of the Advances in Cryptology—CRYPTO'97: 17th Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 1997; Springer: Berlin/Heidelberg, Germany, 1997; pp. 410–424.
9. Wu, Q.; Domingo-Ferrer, J.; González-Nicolás, U. Balanced trustworthiness, safety, and privacy in vehicle-to-vehicle communications. *IEEE Trans. Veh. Technol.* **2009**, *59*, 559–573.
10. Tessaro, S.; Zhu, C. Revisiting BBS Signatures. In Proceedings of the Advances in Cryptology–EUROCRYPT 2023: 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, 23–27 April 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 691–721.
11. Huang, X.; Yu, R.; Kang, J.; Zhang, Y. Distributed reputation management for secure and efficient vehicular edge computing and networks. *IEEE Access* **2017**, *5*, 25408–25420. [CrossRef]
12. Kang, J.; Yu, R.; Huang, X.; Wu, M.; Maharjan, S.; Xie, S.; Zhang, Y. Blockchain for secure and efficient data sharing in vehicular edge computing and networks. *IEEE Internet Things J.* **2018**, *6*, 4660–4670. [CrossRef]
13. Wang, T.; Bhuiyan, M.Z.A.; Wang, G.; Qi, L.; Wu, J.; Hayajneh, T. Preserving balance between privacy and data integrity in edge-assisted Internet of Things. *IEEE Internet Things J.* **2019**, *7*, 2679–2689. [CrossRef]
14. Jan, M.A.; Zhang, W.; Usman, M.; Tan, Z.; Khan, F.; Luo, E. SmartEdge: An end-to-end encryption framework for an edge-enabled smart city application. *J. Netw. Comput. Appl.* **2019**, *137*, 1–10. [CrossRef]
15. Gai, K.; Wu, Y.; Zhu, L.; Xu, L.; Zhang, Y. Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks. *IEEE Internet Things J.* **2019**, *6*, 7992–8004. [CrossRef]
16. Liu, H.; Zhang, P.; Pu, G.; Yang, T.; Maharjan, S.; Zhang, Y. Blockchain empowered cooperative authentication with data traceability in vehicular edge computing. *IEEE Trans. Veh. Technol.* **2020**, *69*, 4221–4232. [CrossRef]
17. Lu, J.; Shen, J.; Vijayakumar, P.; Gupta, B.B. Blockchain-based secure data storage protocol for sensors in the industrial internet of things. *IEEE Trans. Ind. Inform.* **2021**, *18*, 5422–5431. [CrossRef]
18. Yang, Y.; Zhang, L.; Zhao, Y.; Choo, K.K.R.; Zhang, Y. Privacy-preserving aggregation-authentication scheme for safety warning system in Fog-Cloud based VANET. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 317–331. [CrossRef]
19. Ehui, B.B.; Chen, C.; Wang, S.; Guo, H.; Liu, J. A Secure Mutual Authentication Protocol Based on Visual Cryptography Technique for IoT-Cloud. *Chin. J. Electron.* **2022**, *33*, 1–16.
20. Yang, K.; Yang, B.; Wang, T.; Zhou, Y. Zero-Cerd: A Self-Blindable Anonymous Authentication System Based on Blockchain. *Chin. J. Electron.* **2023**, *32*, 587–596. [CrossRef]
21. Li, M.; Zhu, L.; Zhang, Z.; Lal, C.; Conti, M.; Alazab, M. User-defined privacy-preserving traffic monitoring against n-by-1 jamming attack. *IEEE/ACM Trans. Netw.* **2022**, *30*, 2060–2073. [CrossRef]
22. Lin, X.; Sun, X.; Ho, P.H.; Shen, X. GSIS: A secure and privacy-preserving protocol for vehicular communications. *IEEE Trans. Veh. Technol.* **2007**, *56*, 3442–3456.
23. Wang, Y.; Wang, X.; Dai, H.N.; Zhang, X.; Imran, M. A Data Reporting Protocol with Revocable Anonymous Authentication for Edge-assisted Intelligent Transport Systems. *IEEE Trans. Ind. Inform.* **2022**, *19*, 7835–7847. [CrossRef]
24. Boneh, D.; Boyen, X.; Shacham, H. Short group signatures. In *Annual International Cryptology Conference*; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3152, pp. 41–55.
25. Au, M.H.; Susilo, W.; Mu, Y. Constant-size dynamic k-TAA. In Proceedings of the Security and Cryptography for Networks: 5th International Conference, SCN 2006, Maiori, Italy, 6–8 September 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 111–125.
26. Tsiounis, Y.; Yung, M. On the security of ElGamal based encryption. In Proceedings of the Public Key Cryptography: First International Workshop on Practice and Theory in Public Key Cryptography, PKC'98 Pacifico, Yokohama, Japan, 5–6 February 1998; Springer: Berlin/Heidelberg, Germany, 2006; pp. 117–134.
27. Chaum, D.; Van Heyst, E. Group signatures. In Proceedings of the Advances in Cryptology—EUROCRYPT'91: Workshop on the Theory and Application of Cryptographic Techniques, Brighton, UK, 8–11 April 1991; Springer: Berlin/Heidelberg, Germany, 1991; pp. 257–265.
28. Schnorr, C.P. Efficient identification and signatures for smart cards. In *Advances in Cryptology—CRYPTO'89 Proceedings 9*; Springer: New York, NY, USA, 1990; pp. 239–252.
29. Li, J.; Li, Y.; Cao, C.; Lam, K.Y. Conditional anonymous authentication with abuse-resistant tracing and distributed trust for internet of vehicles. *IEEE Internet Things J.* **2021**, *9*, 8749–8762. [CrossRef]
30. Aranha, D.F.; Gouvêa, C.P.L.; Markmann, T.; Wahby, R.S.; Liao, K. RELIC Is an Efficient LIbrary for Cryptography. Available online: https://github.com/relic-toolkit/relic (accessed on 29 January 2020).


**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.