Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.073441

ARTICLE

An Efficient Certificateless Authentication Scheme with Enhanced Security for NDN-IoT Environments

Feihong Xu[1], **Jianbo Wu**[1,*], **Qing An**[1,*], **Fei Zhu**[1,2], **Zhaoyang Han**[3] **and Saru Kumari**[4]

[1]Hubei Engineering Research Center for BDS-Cloud High-Precision Deformation Monitoring, School of Artificial Intelligence, Wuchang University of Technology, Wuhan, 430223, China

[2]School of Computer Science and Artificial Intelligence, Wuhan Textile University, Wuhan, 430200, China

[3]College of Information Science & Technology, Nanjing Forestry University, Nanjing, 210037, China

[4]Department of Mathematics, Chaudhary Charan Singh University, Meerut, 250004, Uttar Pradesh, India

*Corresponding Authors: Jianbo Wu. Email: 120160287@wut.edu.cn or 120161684@wut.edu.cn; Qing An. Email: anqing@wut.edu.cn

Received: 18 September 2025; Accepted: 15 December 2025; Published: 10 February 2026

ABSTRACT: The large-scale deployment of Internet of Things (IoT) technology across various aspects of daily life has significantly propelled the intelligent development of society. Among them, the integration of IoT and named data networks (NDNs) reduces network complexity and provides practical directions for content-oriented network design. However, ensuring data integrity in NDN-IoT applications remains a challenging issue. Very recently, Wang et al. (Entropy, 27(5), 471(2025)) designed a certificateless aggregate signature (CLAS) scheme for NDN-IoT environments. Wang et al. stated that their construction was provably secure under various types of security attacks. Using theoretical analysis methods, in this work, we reveal that their CLAS design fails to meet unforgeability, a core security requirement for CLAS schemes. In particular, we demonstrate that their scheme is vulnerable to a malicious public-key replacement attack, enabling an adversary to produce authentic signatures for arbitrary fraudulent messages. Therefore, Wang et al.'s design cannot achieve its goal. To address the issue, we systematically examine the root causes behind the vulnerability and propose a security-enhanced CLAS construction for NDN-IoT environments. We prove the security of our improved design under the standard security assumption and also analyze its practical performance by comparing the computational and communication costs with several related works. The comparison results show the practicality of our design.

KEYWORDS: IoT; certificateless signature; public-key replacement attack; data integrity; aggregation

1 Introduction

The Internet of Things (IoT) has seamlessly integrated into our daily lives, transforming industries and urban infrastructure with its interconnected smart systems. However, the widespread interconnection of IoT devices and the rapid growth of data volume pose significant challenges to the security and efficiency of communication systems. To tackle these problems, named data networking (NDN) has gained recognition as an innovative content-centric communication framework, distinguished by its unique strengths [1,2]. Departing from conventional address-centric network models, NDN adopts a data-centric paradigm enabled by name-driven routing protocols, delivering superior flexibility, scalability, and native security features. In short, NDN shifts the model from host-to-host communication, like the current Internet Protocol (IP), to a data-centric model where users request content by name. However, integrating NDN and IoT contexts introduces multifaceted security complexities. Recall that data is the core resource of NDN-IoT applications, necessitating robust protective measures to safeguard its security. In real-world scenarios, however, data frequently traverses insecure public networks, and faces numerous security threats [3,4]. A key security requirement involves verification mechanisms where data receivers must validate the source's trustworthiness and confirm the data integrity throughout its transmission path [5]. In addition, an observer in NDN may be able to monitor which content names are being requested, potentially revealing sensitive information. Therefore, user's privacy should also not be ignored.

Digital signatures is an essential cryptographic mechanism for guaranteeing both data integrity and source authentication. Moreover, in high-throughput applications such as vehicular ad hoc networks and named data networking (NDN) networks, there are a large number of digital signatures that require efficient validation, which puts higher performance requirements on digital signatures. The aggregate signature scheme, initially put forward by Boneh et al. [6], presents an optimal solution by enabling the compression of n individual signatures into one consolidated form. This approach facilitates batch verification while significantly reducing bandwidth consumption.

Boneh et al.'s framework relies on public key infrastructure (PKI), and its actual deployment faces challenges due to the substantial overhead associated with key management. Alternative aggregate signature schemes using identity-based cryptography have emerged [7] to address PKI's limitations; however, identity-based setting suffers from the inherent key-escrow issue. The certificateless paradigm [8] elegantly resolves both concerns by employing a hybrid key generation model: the key generation center (KGC) supplies partial secret information while users independently select additional secret components, with public keys derived from the user's public information [9]. Due to its merits, recent years have witnessed significant academic interest in certificateless aggregate signature (CLAS) schemes for IoT applications [10,11].

1.1 Related Work & Motivation

To date, a number of CLAS schemes have been designed for IoT applications. Early schemes were designed based on bilinear pairing [12,13], requiring expensive computational costs. Cui et al. [14] designed a pairing-free CLAS scheme for vehicular ad hoc networks. However, their design cannot resist malicious-but-passive KGC attacks (i.e., called as Type 2 attacks) [15]. Xu et al. [16] put forward another CLAS scheme without pairings for VANETs. Zhu et al. [17] pointed out the security vulnerability of [16] in resisting the Type 2 attack and constructed a new scheme with enhanced security. However, their work was further pointed out by Yang et al. [18] to have a security vulnerability of the public-key replacement attack (i.e., called as Type 1 attacks). In [18], Yang et al. then proposed an improved CLAS scheme with new aggregate algorithm, which ensures the validity of all individual signatures participating in the aggregation. But the performance is a weakness of their design. In addition, Zhu and Guan [19] put forward an authentication scheme with conditional privacy protection for vehicular ad-hoc networks based on a CLAS scheme. However, their work cannot achieve Type 1 security [20]. A recent comprehensive survey of CLAS schemes can be found in [21].

More recently, Yue et al. [22] proposed a CLAS scheme for VANETs. However, their design is computationally inefficient and cannot ensure resistance to Type 1 attacks, where an adversary can systematically generate fraudulent signatures for arbitrary messages (refer to Appendix A). This vulnerability fundamentally compromises the unforgeability property, which is a core security requirement for any CLAS schemes. In addition, Wang et al. [23] designed a CLAS scheme for NDN-IoT environments. Wang et al. initially asserted the security of their CLAS construction. Our analysis reveals, however, that their implementation remains vulnerable to public-key replacement attacks. That is, their schemes can not ensure data integrity, thus cannot be deployed in real-world NDN-IoT applications.

Contribution. To solve data security and efficiency problems in NDN-IoT applications, we put forward a new CLAS scheme. The key contributions of this work are outlined below:

1. By presenting a concrete public-key replacement attack, we explored the security vulnerability of a very recent CLAS scheme in [23] proposed for NDN-IoT environments.
2. We systematically examine the root causes behind the vulnerability in [23] and propose an improved CLAS design.
3. We prove the security of our design based on the cryptographic assumption, and analyze its performance. The performance comparison results demonstrate that the improved CLAS scheme not only has better security but also has desirable computational and communication costs. Therefore, our design is suitable for NDN-IoT environments.
4. As an additional contribution, in Appendix A, we analyze the security flaw of a very recent CLAS construction in [22] and propose targeted countermeasures to enhance its security.

Organization. The subsequent sections of this paper are structured as the following: Section 2 introduces the foundational concepts and preliminaries. In Section 3, we review Wang et al.'s scheme in [23] and put forward our security analysis. In Section 4, we introduce our enhanced design with its rigorous security analysis. We evaluate the performance of our proposal in Section 5 and conclude the work in Section 6. In Appendix A, we provide a retrospective analysis of Yue et al.'s construction in [22], including identified security weaknesses and proposed response strategies.

2 Preliminaries

Here, we introduce some required preliminaries, such as notations and elliptic curve discrete logarithm problem (ECDLP).

2.1 Notations

Some notations are listed in Table 1.

Table 1: Notations and descriptions

Notations	Descriptions	Notations	Descriptions
IoT	Internet of Things	NDN	Named data networking
PKI	Public key infrastructure	KGC	Key generation center
CLAS	Certificateless aggregation signature	ECDLP	Elliptic curve discrete logarithm problem
λ	System security parameter	ppa	System public parameters
P_{kgc}	Public key of the KGC	s	Private key of the KGC
ID_i/PID_i	Identity/Pseudonym of entity i	D_i	Partial private key of entity i
(PK_i, SK_i)	Public/private key pair of sensor i	t_i	Timestamp
(m_i, σ_i)	Message-signature pair of i	σ	Aggregated signature for n entities

2.2 ECDLP

Let G be an q-order cyclic elliptic curve group and P be a generator of G. Given $(P, \alpha P) \in G$ for some unknown $\alpha \in Z_q^*$, the ECDLP is to find α.

3 Security Attack to Wang et al.'s CLAS Scheme in [23]

As shown in Fig. 1, there are several entities in [23]. The KGC is responsible for building the system. An end device (ED) can register as a producer or consumer in the network system by interacting with KGC. Acting as a vital element for secure data forwarding, the NDN router checks the integrity of data packets during transmission. It conducts signature verification on the embedded producer details within the data packets. Moreover, it supports batch processing of multiple signatures from multiple end devices. As a data requester, the consumer can send Interest packets to request needed data or services. In addition, the producer, which corresponds to the producer entity in NDN, is in charge of generating data in the NDN-IoT environment. It employs sensor devices to gather information like soil moisture levels, vehicle locations, and indoor temperatures.

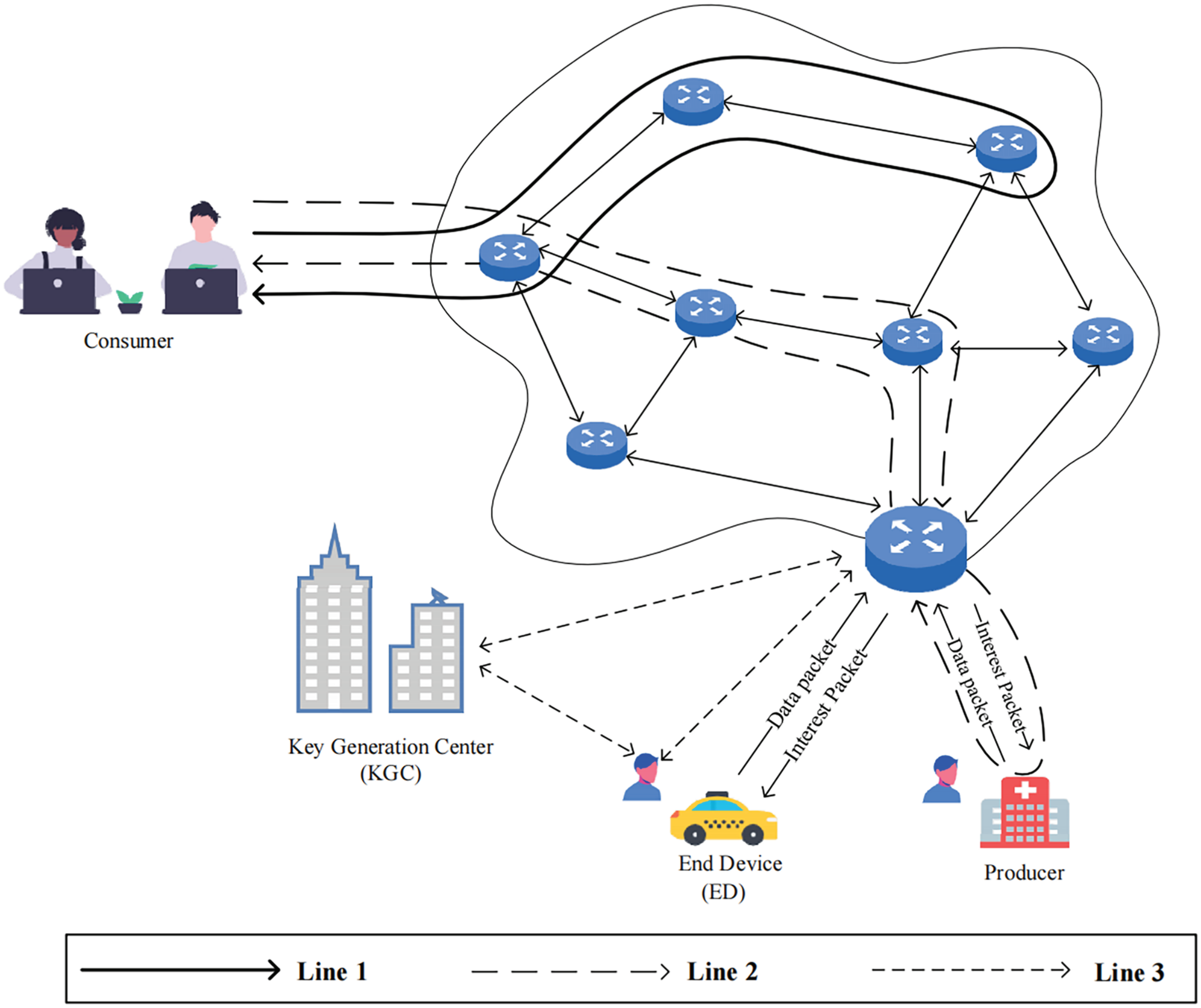

Figure 1: Wang et al's system structure. The figure is adopted from [23]. Line 1 depicts an instance of how a consumer seeks data forwarding from an NDN router. Line 2 showcases the procedure where a consumer requests data packets from multiple producers. Line 3 presents the interaction between terminal devices and the KGC for registration purposes, along with the process of creating an aggregate signature and sending data packets through the NDN router

The CLAS scheme proposed by Wang et al. [23] mainly formed by the following algorithms: System Setup, Device Pseudonym Generation, Device Keys Generation, Signing, Single Signature Verification, and Aggregated Signature Verification. We now briefly review their algorithms to support our analysis.

1. System Setup: Taking a security parameter ζ as input, the KGC sets up the system as below:
   (a) Define an q-order cyclic group $\mathbb{G} = \langle P \rangle$.

(b) Randomly select a master private key $\alpha \in \mathbb{Z}_q^*$ and calculate a public key $PK_{kgc} = \alpha P$.

(c) Choose three hash functions $H_i : \{0,1\}^* \to \mathbb{Z}_q^*$, $i = 1, 2, \ldots, 3$.

(d) Store α secretly and publish public parameters $ppa = \{\mathbb{G}, P, q, PK_{kgc}, H_i\}$.

2. Device Pseudonym Generation: In this algorithm, a terminal device ED_i with real identity ID_i interacts with KGC to generate a pseudonym $PID_i = \{AID_{i2}, T_i\}$ for a validity period T_i.

(a) ED_i randomly picks $e_i \in \mathbb{Z}_q^*$ and computes $E_i = e_i P$, $F_i = e_i P_{kgc}$, and $AID_{i1} = ID_i \oplus F_i$. Then, it sends $\{E_i, F_i, AID_{i1}\}$ to the KGC.

(b) KGC recovers $ID_i = AID_{i1} \oplus E_i$, computes $AID_{i2} = H_1(T_i, \alpha AID_{i1}) \oplus ID_i$, and sends $PID_i = \{AID_{i2}, T_i\}$ to ED_i.

3. Device Keys Generation:

(a) ED_i randomly picks $x_i \in \mathbb{Z}_q^*$ and computes $X_i = x_i P$.

(b) KGC picks $r_i \in \mathbb{Z}_q^*$ at random, computes $R_i = r_i P$, $h_i^{(2)} = H_2(PID_i, R_i, P_{kgc})$, $d_i = r_i + \alpha h_i^{(2)}$, and provides ED_i with the partial private key $D_i = (d_i, R_i)$.

(c) ED_i computes $h_i^{(2)} = H_2(PID_i, R_i, P_{kgc})$, $K_i = h_i^{(2)} X_i + R_i$, and sets its private key $SK_i = d_i + h_i^{(2)} x_i$ and public key $PK_i = (K_i, R_i)$.

4. Signing: To sign a message $m_i \in \{0,1\}^*$ at time t_i, ED_i performs the following:

(a) Pick $u_i \in \mathbb{Z}_q^*$ at random and calculate $U_i = u_i P$ and $h_i^{(3)} = H_3(m_i, PID_i, PK_i, U_i, t_i)$.

(b) Compute $V_i = u_i + h_i^{(3)} SK_i$ and set $\sigma_i = (U_i, V_i)$ as the signature.

5. Single Signature Verification: Given $\{PID_i, PK_i, m_i, \sigma_i, t_i\}$, the verifier verifies the freshness of t_i, recovers $h_i^{(2)} = H_2(PID_i, R_i, P_{kgc})$, and $h_i^{(3)} = H_3(m_i, PID_i, PK_i, U_i, t_i)$. It accepts the signature if $V_i P = U_i + h_i^{(3)}(K_i + h_i^{(2)} P_{kgc})$ and rejects otherwise.

6. Signature Aggregation: For n messages $\{PID_i, PK_i, m_i, \sigma_i, t_i\}$ from n ED_i, the verifier computes an aggregated signature $\sigma_{ag} = (U, V)$, where $U = \sum_{i=1}^{n} U_i$ and $V = \sum_{i=1}^{n} V_i$.

7. Aggregation Verification: To check the validity of $\sigma_{ag} = (U, V)$, verifier verifies first checks whether t_i is fresh. Then, it recovers $h_i^{(2)}$ and $h_i^{(3)}$ for $i = 1, 2, \ldots, n$. It accepts the signature if $VP = U + \sum_{k=1}^{n} h_i^{(3)}((K_i + h_i^{(2)} P_{kgc})$ and rejects otherwise.

3.1 Security Analysis to [23]

The first five algorithms in [23] naturally form a CLS scheme and the remaining algorithms are used to perform batch verification of multiple signatures. For ease presentation, our analysis focuses on their CLS scheme. For a CLS scheme, two distinct types of attackers should be considered, i.e., public-key replacement attacker (called as Type 1 attacker) and malicious-but-passive KGC (called as Type 2 attacker). In particular, a Type 1 attacker knows a target user's secret value. However, the attacker cannot access the user's partial private key. A Type 2 attacker knows the KGC's private key but does not allowed to access the target user's secret value. For more security definitions and security models, please refer to [23].

In [23], Wang et al. stated that their design can achieve both Type 1 and Type 2 security. In the following, we show that a Type 1 attacker $\mathscr{F}_1$ possesses the capability to produce a verifiable signature for any fraudulent message, thereby compromising the unforgeability property inherent in their cryptographic construction. Let the device ED_i with pseudonym PID_i be the target device attacked by $\mathscr{F}_1$. Given public parameters $ppa = \{\mathbb{G}, P, q, PK_{kgc}, H_i\}$ and ED_i's public key $PK_i = (K_i, R_i)$, $\mathscr{F}_1$ can also access EDi's secret value x_i. Suppose that $\mathscr{F}_1$ tries to generate a forgery σ_i^* on a message m_i^* at time t_i, as shown in Fig. 2, $\mathscr{F}_1$ operates as follows:

(1) Compute $h_i^{(2)} = H_2(PID_i, R_i, P_{kgc})$.

(2) Pick $\beta \in \mathbb{Z}_q^*$ at random and set $K_i^* = \beta P - h_i^{(2)} P_{kgc}$.
(3) Set $PK_i^* = (K_i^*, R_i)$ as the replaced public key.
(4) Select $u_i^* \in \mathbb{Z}_q^*$ at random and compute $U_i^* = u_i^* P$, $h_i^{(3)*} = H_3(m_i^*, PID_i, PK_i^*, U_i^*, t_i)$, and $V_i^* = u_i^* + h_i^{(3)*}\beta$.
(5) Set $\sigma_i^* = (U_i^*, V_i^*)$ as the forged signature.

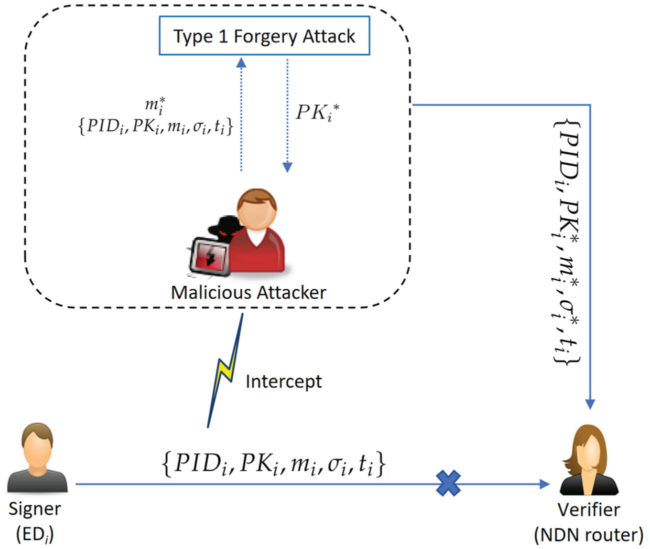
Type 1 Forgery Attack
m_i^*
$\{PID_i, PK_i, m_i, \sigma_i, t_i\}$
PK_i^*
Malicious Attacker
$\{PID_i, PK_i^*, m_i^*, \sigma_i^*, t_i\}$
Intercept
$\{PID_i, PK_i, m_i, \sigma_i, t_i\}$
Signer
(ED$_i$)
Verifier
(NDN router)

Figure 2: An example of the Type 1 attack

Now, the correctness of σ_i^* is checked by:

$$\begin{aligned} V_i^* P &= (u_i^* + h_i^{(3)*}\beta)P = u_i^* P + h_i^{(3)*}\beta P \\ &= U_i^* + h_i^{(3)*}(K_i^* + h_i^{(2)} P_{kgc}). \end{aligned}$$

Since the underlying CLS scheme is insecure, the CLAS construction is therefore cannot achieve unforgeability.

4 Our Improvement

In [23], a verifier checks a received signature through the equation $V_i P = U_i + h_i^{(3)}(K_i + h_i^{(2)} P_{kgc})$. However, due to the lack of binding between K_i and $h_i^{(2)}$, a $\mathscr{F}_1$ attacker can use the algebraic relationship in the equation to bypass the KGC's private key α (corresponding to P_{kgc}).

To patch this vulnerability, our improvement is as follows:

1. The algorithms System Setup and Device Pseudonym Generation are the same as the original scheme.
2. Device Keys Generation:
   (a) ED_i randomly picks $x_i \in \mathbb{Z}_q^*$ and computes $X_i = x_i P$.
   (b) KGC picks $r_i \in \mathbb{Z}_q^*$ at random, computes $R_i = r_i P$, $h_i^{(2)} = H_2(PID_i, X_i, R_i, P_{kgc})$, $d_i = r_i + \alpha h_i^{(2)}$, and securely provides ED_i with the partial private key $D_i = (d_i, R_i)$.
   (c) ED_i sets its private key $SK_i = (x_i, d_i)$ and public key $PK_i = (X_i, R_i)$.

3. Signing: To generate a signature on message $m_i \in \{0,1\}^*$ at time t_i, ED_i performs the following:
   (a) Randomly pick $u_i \in \mathbb{Z}_q^*$ and calculate $U_i = u_i P$ and $h_i^{(3)} = H_3(m_i, PID_i, PK_i, U_i, t_i)$.
   (b) Compute $V_i = u_i + h_i^{(3)}(x_i + d_i)$ and set the signature $\sigma_i = (U_i, V_i)$.
4. Single Signature Verification: Given $\{PID_i, PK_i, m_i, \sigma_i, t_i\}$, the verifier checks whether t_i is fresh. It then recovers $h_i^{(2)} = H_2(PID_i, X_i, R_i, P_{kgc})$ and $h_i^{(3)} = H_3(m_i, PID_i, PK_i, U_i, t_i)$. It accepts the signature if $V_i P = U_i + h_i^{(3)}(X_i + R_i + h_i^{(2)} P_{kgc})$ and rejects otherwise. The correctness:

$$\begin{aligned} V_i P &= (u_i + h_i^{(3)}(x_i + d_i))P = u_i P + h_i^{(3)}(x_i + d_i))P \\ &= U_i + h_i^{(3)}(x_i P + d_i P) = U_i + h_i^{(3)}(X_i + R_i + h_i^{(2)} P_{kgc}). \end{aligned}$$

5. Signature Aggregation: The algorithm is the same as the original scheme.
6. Aggregation Verification: To check the validity of $\sigma_{ag} = (U, V)$, the verifier checks whether t_i is fresh. Then, it recovers $h_i^{(2)}$ and $h_i^{(3)}$ for $i = 1, 2, \ldots, n$. It accepts the signature if $VP = U + \sum_{i=1}^{n} h_i^{(3)}(X_i + R_i) + (\sum_{i=1}^{n} h_i^{(3)} h_i^{(2)}) P_{kgc}$ and rejects otherwise. The correctness:

$$\begin{aligned} VP &= \left(\sum_{i=1}^{n} V_i\right) P = \sum_{i=1}^{n} \left(u_i + h_i^{(3)}\,(x_i + d_i)\right) P = \sum_{i=1}^{n} \left(U_i + h_i^{(3)}\left(X_i + R_i + h_i^{(2)} P_{kgc}\right)\right) \\ &= \sum_{i=1}^{n} U_i + \sum_{i=1}^{n} h_i^{(3)}\left(X_i + R_i + h_i^{(2)} P_{kgc}\right) = U + \sum_{i=1}^{n} h_i^{(3)}(X_i + R_i) + \left(\sum_{i=1}^{n} h_i^{(3)} h_i^{(2)}\right) P_{kgc}. \end{aligned}$$

Following Wang et al.'s proof approach in [23], the modified scheme can be easily proven to be secure. To avoid a lot of repetitive proof work, we omit the proof process here. Compared to the original scheme, the improvement adds one point multiplication, one point addition, and a general hash operation. This is acceptable since the modification achieves greater security.

4.1 Security Proof

Here, we proof the security of our improved design. Note that for ease presentation, our proof directly focuses on our underlying CLS scheme. Following the proof idea in [23,24], the improved CLS design is resistant to forgery attacks against Type 1 and Type 2 adversaries.

Theorem 1: *The improved CLS scheme is secure against any Type 1 adversary if ECDLP is hard.*

Proof: This theorem demonstrates that if a Type 1 adversary $\mathbb{A}_1$ compromises the underlying CLS scheme, there must exist an adversary $\mathbb{B}$ capable of resolving the ECDLP. Now, $\mathbb{A}_1$ and $\mathbb{B}$ performs the following:

- Stage-1: $\mathcal{B}$ operates as System Setup to obtain system parameters $ppa = \{\mathbb{G}, P, q, PK_{kgc}, H_i\}$, where $P_{kgc} = \alpha P$ for some unknown $\alpha \in \mathbb{Z}_q^*$. It sends ppa to $\mathcal{A}_1$. For simplicity, let PID_{i^*} be $\mathcal{A}_1$'s target identity. During the forgery game, $\mathcal{A}_1$ keeps a series of lists to store the query results. In the initial stage, all lists are empty.
- Stage-2: In this stage, $\mathcal{B}$ responds to $\mathcal{A}_1$'s adaptive queries as below.
  H_2-Query: For a H_2 query on (PID_i, R_i, P_{kgc}), if the item $(PID_i, X_i, R_i, P_{kgc}, h_i^{(2)})$ can be found in the list L_{H_2}, $\mathcal{B}$ returns $h_i^{(2)}$ to $\mathcal{A}_1$. Otherwise, $\mathcal{B}$ picks $h_i^{(2)} \in_R \mathbb{Z}_q^*$, inserts $(PID_i, X_i, R_i, P_{kgc}, h_i^{(2)})$ to L_{H_2}, and responds $h_i^{(2)}$ to $\mathcal{A}_1$.
  H_3-Query: For a H_3 query on $(m_i, PID_i, PK_i, U_i, t_i)$, if the item $(m_i, PID_i, PK_i, U_i, t_i, h_i^{(3)})$ exists in the list L_{H_3}, $\mathcal{B}$ returns $h_i^{(3)}$ to $\mathcal{A}_1$. Otherwise, $\mathcal{B}$ picks $h_i^{(3)} \in_R \mathbb{Z}_q^*$, inserts $(m_i, PID_i, PK_i, U_i, t_i, h_i^{(3)})$ to L_{H_3}, and responds $h_i^{(3)}$ to $\mathcal{A}_1$.

Secret value-Query: $\mathcal{A}_1$ can issue such query on PID_i. $\mathcal{B}$ searches the tuple (PID_i, x_i, X_i) from the list L_{sv} and sends it to $\mathcal{A}_1$. Otherwise, $\mathcal{B}$ selects $x_i \in_R \mathbb{Z}_q^*$, stores (PID_i, x_i, X_i) to L_{sv}, and responds x_i to $\mathcal{A}_1$.

Partial private key-Query: $\mathcal{A}_1$ can issue any partial secret key query regarding PID_i. If $PID_i = PID_{i^*}$, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ searches the list L_{psk} to find (PID_i, d_i, R_i) and send it to $\mathcal{A}_1$. If the tuple (PID_i, d_i, R_i) does not exist in L_{psk} and the tuple $(PID_i, X_i, R_i, P_{kgc}, h_i^{(2)})$ does not exist in L_{H_2}, $\mathcal{B}$ selects $d_i, h_i^{(2)} \in_R \mathbb{Z}_q^*$, computes $R_i = d_i P - h_i^{(2)} P_{kgc}$, and sets $h_i^{(2)} = H_2(PID_i, X_i, R_i, P_{kgc})$. $\mathcal{A}$ updates lists L_{H_2} and L_{psk} and provides $\mathcal{A}_1$ with (PID_i, d_i, R_i). Public key-Query: Once $\mathcal{B}$ receives $\mathcal{A}_1$'s query on PID_i $(PID_i = PID_{i^*})$, $\mathcal{B}$ checks if $(PID_i, x_i, X_i, d_i, R_i)$ exists in the list L_{key}. If it exists, $\mathcal{B}$ returns (X_i, R_i). Otherwise, $\mathcal{B}$ runs as Secret value-Query and Partial private key-Query to generate and update $(PID_i, x_i, X_i, d_i, R_i)$, and then returns (X_i, R_i).

Public key replacement-Query: Once $\mathcal{B}$ receives a query for some (ID_i, PK_i, PK_i') from $\mathcal{A}_1$, $\mathcal{B}$ searches the tuple (ID_i, PK_i) from L_{key} and replaces it with $(PID_i, \perp, X_i', d_i, R_i)$.

Signing-Query: Upon receiving $\mathcal{A}_1$'s query on (m_i, PID_i), $\mathcal{B}$ performs as below. If $PID_i \neq PID_{i^*}$, $\mathcal{B}$ scans the lists to obtain the required parameters and runs as Signing to produce a signature $\sigma_i = (U_i, V_i)$ as the response. Otherwise, $\mathcal{B}$ picks $h_i^{(2)}$, $h_i^{(3)}$, $V_i \in_R \mathbb{Z}_q^*$, sets $U_i = V_i P - h_i^{(3)}(X_i + R_i + h_i^{(2)} P_{kgc})$, and returns $\sigma_i = (U_i, V_i)$.

- Stage-3: Eventually, $\mathcal{F}_1$ either admits failure or returns its forgery $\sigma_i^* = (U_i^*, V_i^*)$ on m_i^*.

  If σ_i^* is a valid forgery under (PID_i^*, m_i^*), then $V_i^* P = U_i^* + h_i^{(3^*)}(X_i^* + R_i + h_i^{(2^*)} P_{kgc})$ holds. By applying the forking lemma in [25], $\mathcal{B}$ replays $\mathcal{A}_1$ with the same random tape, but provides two distinct values of H3. $\mathcal{A}_1$ can output another valid signature $\sigma_i^* = (U_i^*, V_i^{*'})$. Hence, we have $V_i^{*'} P = U_i^* + h_i^{(3^*)}(X_i^* + R_i + h_i^{(2^{*'})} P_{kgc})$. Therefore, $\mathcal{B}$ calculates $\alpha = (V_i^* - V_i^{*}{}')(h_i^{(3^*)}(h_i^{(2^*)} - h_i^{(2^{*'})}))^{-1}$ as a solution to ECDLP. □

Theorem 2: *The improved CLS scheme is secure against any Type 2 adversary if ECDLP is hard.*

Proof: The proof follows a similar approach to that of Theorem 1 and is thus omitted for brevity. □

Theorem 3: *The improved CLS scheme achieves conditional privacy-preserving.*

Proof: In our design, the anonymity of the end device is assured by the pseudonym PID_i. Recall that $PID_i = \{AID_{i2}, T_i\}$, where $AID_{i2} = H_1(T_i, \alpha AID_{i1}) \oplus ID_i$, $AID_{i1} = ID_i \oplus F_i$, $F_i = e_i P_{kgc}$, $E_i = e_i P$, $e_i \in \mathbb{Z}_q^*$, and T_i is the valid period. To extract the real identity ID_i, the attacker must compute $H_1(T_i, \alpha AID_{i1})$. However, computing $H_1(T_i, \alpha AID_{i1})$ means that the attacker must know α and AID_{i1}. Note that α is the private key of the KGC. Meanwhile, according to the above equation, to compute AID_{i1}, the attacker needs to recover e_i from $E_i = e_i P$, which is solving the ECDLP. Due to the hardness of the ECDLP, it is evident that no such attacker can reveal ID_i. However, in scenarios where an end device fails to operate correctly or triggers an operational issue, the KGC can trace ID_i to take appropriate action in a timely manner.

In Signing, to generate a signature on a message, three distinct random numbers e_i, x_i, and u_i are generated by the end device. The inherent randomness of these random numbers ensures that the attacker cannot correlate anonymous identities or associate disparate signatures produced by the same end device, thereby achieving unlinkability in our improvement. The combination of the above properties implies the proof. □

5 Performance Analysis

This section analyzes the performance of our design by comparing its computational and communication costs with recent schemes in [17,22,23]. We adopt the experiment parameters provided in [23] for our analysis, which was tested on a Raspberry Pi 3B+ device under the Curve25519 elliptic curve, achieving 128-bit security level. Specifically, the running time for different operations is as follows: general hash $T_h = 0.0729$ ms, point addition operation $T_{pa} = 0.1652$ ms, and point multiplication operation $T_{pm} = 23.4405$ ms.

Computational costs. Taking the algorithm Signing in our improved scheme as an example, it executes one point multiplication operation and one general hash operation to generate the signature. Hence, the total computational cost is $T_{pm} + T_h = 23.5134$ ms. Similarly, we count the cost for the remaining schemes and record the computational costs in Table 2.

Table 2: Comparison of computation cost with related works

Scheme	Signing	Single signature verification	Aggregation verification	Communication cost
[17]	$T_{pm} + 3T_h$ $\approx 23.6592ms$	$4T_{pm} + 3T_{pa} + 3T_h$ $\approx 94.4763ms$	$(4n+4)T_{pm} + 3nT_{pa} + 3nT_h$ $\approx 94.4763n + 93.7620ms$	$4\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q^*\rvert + 8$ $= 200$ bytes
[23]	$T_{pm} + T_h$ $\approx 23.5134ms$	$3T_{pm} + 2T_{pa} + 2T_h$ $\approx 70.7977ms$	$(2n+1)T_{pm} + 2nT_{pa} + 2nT_h$ $\approx 47.3572n + 23.4405ms$	$3\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q^*\rvert + 8$ $= 168$ bytes
[22]	$T_{pm} + 2T_h$ $\approx 23.5863ms$	$4T_{pm} + 3T_{pa} + 3T_h$ $\approx 94.4763ms$	$(3n+5)T_{pm} + (4n+4)T_{pa} + (3n+3)T_h$ $\approx 71.2010n + 118.0820ms$	$3\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q^*\rvert + 8$ $= 168$ bytes
Ours	$T_{pm} + T_h$ $\approx 23.5134ms$	$3T_{pm} + 3T_{pa} + 2T_h$ $\approx 70.9639ms$	$(n+2)T_{pm} + 2nT_{pa} + 2nT_h$ $\approx 23.9167n + 46.8810ms$	$3\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q^*\rvert + 8$ $= 168$ bytes

In Signing, the cost of our design is the same as that of [23] and lower than that of [17] (i.e., 23.6592 ms) and [22] (i.e., 23.5863 ms). In Verification, the schemes in [17,22] require a relatively high computational cost. Though the cost of our scheme is slightly higher than that of [23], the gap between them is quite small (i.e., 0.1652 ms). In addition, as can be seen from the table and Fig. 3, our scheme achieves the smallest computational cost in Aggregate Verification. Therefore, our scheme has better security and desirable computational cost.

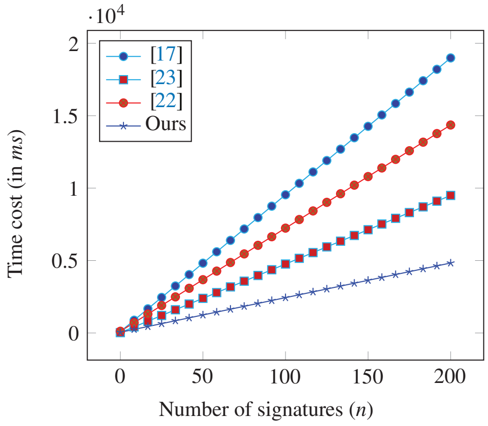

Figure 3: Computational costs comparison between the improved CLAS scheme and [17,22,23] in aggregation verification phase

Communication costs. Based on the above curve parameters, the length of $\mathbb{G}$ and $\mathbb{Z}_q^*$ can be represented by 32 bytes and 32 bytes, respectively [26,27]. We assume that the size of both the identity and the timestamp is 4 bytes. In our design, the signer needs to send $\{PID_i, PK_i, m_i, \sigma_i, t_i\}$ to the verifier, where $PID_i = \{AID_{i2}, T_i\}$, $PK_i = (X_i, R_i)$, and $\sigma_i = (U_i, V_i)$. Since $X_i, R_i, U_i \in G$ and $AID_{i2}, V_i \in Z_q^*$, the cost is $3|\mathbb{G}| + 2|\mathbb{Z}_q^*| + 8 = 32 \times 5 + 8 = 168$ bytes. Similarly, Table 2 counts the communication costs of these schemes. The above results indicate that the communication cost required for the scheme in [17] is 200 bytes, while other schemes, including ours, only require 168 bytes.

In summary, our improved scheme not only has better security but also has desirable computational and communication costs.

Discussion

In Wang et al.'s design in [23], the main reason why their proposal has the security vulnerability under Type 1 attack is that the verification equation $V_iP = U_i + h_i^{(3)}(K_i + h_i^{(2)}P_{kgc})$ in the verification algorithm has some special algebraic relationship (i.e., K_i and $h_i^{(2)}$ are independent and do not affect each other). As we analyzed in Section 3.1, the Type 1 attacker uses such an algebraic relationship to replace K_i by setting $K_i^* = \beta P - h_i^{(2)}P_{kgc}$, where the random $\beta \in \mathbb{Z}_q^*$. Hence, the attacker can bypass the KGC's private key α (corresponding to $P_{kgc} = \alpha P$).

In our improvement, we have made corresponding adjustments to the device private-public key pair generation method, signature generation process, and verification equation, avoiding the problems found in Wang et al.'s design. The above performance analysis shows that compared with existing work, our improvement has reached the optimal state in signature generation, signature batch verification processing, and communication cost. However, our work cannot not achieve optimal performance in terms of single signature verification. This is a cost for our solution in achieving high security. To address this limitation, a feasible approach is to combine certificateless cryptosystems with lightweight hash-based message authentication code [28] to construct new privacy preserving authentication schemes. However, However, this may require a new security model.

6 Conclusion

In this effort, we explored the security vulnerability of a very recent CLAS scheme in [23] proposed for NDN-IoT environments. By presenting a specific Type 1 attack, our analysis demonstrates how attackers can use their scheme to forge legitimate signatures for fraudulent environmental data. This manipulation allows malicious actors to deceive consumers, thereby guiding them to make wrong decisions. In view of this, we have systematically examined the root causes behind the vulnerability in [23] and proposed an improved CLAS design to secure NDN-IoT applications. We proved its security based on the cryptographic assumption, and analyzed its performance. The performance comparison results showed that our improved scheme not only has better security but also has desirable computational and communication costs. Finally, as an additional contribution, we analysed the security vulnerability of a very recent CLAS scheme in [22] and proposed targeted countermeasures to enhance its security.

Acknowledgement: None.

Funding Statement: This work was supported in part by the Hubei Engineering Research Center for BDS-Cloud High-Precision Deformation Monitoring Open Funding (No. HBBDGJ202507Y), in part by the National Natural Science Foundation of China (No. 62377037).

Author Contributions: Conceptualization, Feihong Xu, Fei Zhu, Saru Kumari; Methodology, Jianbo Wu, Qing An; Writing—original draft, Feihong Xu and Fei Zhu; Writing—review & editing, Feihong Xu, Jianbo Wu, Qing An, Zhaoyang Han, and Saru Kumari. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

Appendix A Cryptanalysis and Improvement of Yue et al.'s CLAS scheme in [22]

Appendix A.1 Review of the Original Scheme

The CLAS scheme introduced by Yue et al. [22] consists of the following algorithms: System Setup, Pseudonym Identity Generation, Partial Private Key Generation, Vehicle Key Generation, Individual Signature Generation, Single Signature Verification, Aggregated Signature Generation, and Aggregated Signature Verification. For ease of presentation, we only briefly review their first six algorithms to support our analysis, which naturally form a CLS scheme.

1. System Setup: Taking a security parameter ζ as input, the KGC and TA generate below system parameters:
   (a) Define an q-order cyclic group $\mathbb{G} = \langle P \rangle$.
   (b) Choose hash functions $H_i : \{0,1\}^* \to \mathbb{Z}_q^*$, $i = 0, 1, \ldots, 4$.
   (c) (KGC:) Randomly select a private key $a \in \mathbb{Z}_q^*$ and calculate a public key $PK_{kgc} = aP$.
   (d) (TA:) Randomly select a private key $b \in \mathbb{Z}_q^*$ and calculate a public key $PK_{ta} = bP$.
(e) KGC and TA store a and b, respectively, and publish public parameters $ppa = \{\mathbb{G}, P, q, PK_{kgc}, PK_{ta}, H_i\}$.
2. Pseudonym Identity Generation: A vehicle V_i with real identity RID_i interacts with TA to generate a pseudonym PID_i for a validity period T_i.
   (a) V_i randomly picks $x_i \in \mathbb{Z}_q^*$ and calculates $X_i = x_i P$, $TID_i = RID_i \oplus H_0(x_i T_{pub})$, and submits $\{X_i, TID_i\}$ to TA.
   (b) TA extracts $RID_i = TID_i \oplus H_0(bX_i)$, computes pseudonym $PID_i = RID_i \oplus H_1(bX_i, T_i, t_i)$, and sends $\{PID_i, X_i, T_i, t_i\}$ to KGC, where T_i is the validity period for PID_i and t_i is current timestamp.
3. Partial Private Key Generation: After checking the validity of T_i and t_i, the KGC randomly picks $r_i \in Z_q^*$ and computes $R_i = r_i P$, $h_i^{(2)} = H_2(PID_i, R_i, PK_{kgc}, X_i, T_i)$, $d_i = r_i + a h_i^{(2)}$, and $D_i = d_i \oplus H_0(aX_i)$. Then, it sends the tuple $\{PID_i, D_i, R_i, T_i, t_i\}$ to V_i.
4. Vehicle Key Generation: V_i recovers $d_i = D_i \oplus H_0(x_i PK_{kgc})$ and $h_i^{(2)} = H_2(PID_i, R_i, PK_{kgc}, X_i, T_i)$. Note that d_i can be checked by $d_i P = R_i + h_i^{(2)} PK_{kgc}$. If and only if d_i is valid, V_i accepts its private key $SK_i = (x_i, d_i)$ and public key $PK_i = (X_i, R_i)$.
5. Signing: To sign a message $m_i \in \{0,1\}^*$ at time t_i, V_i performs the following:
   (a) Select $u_i \in \mathbb{Z}_q^*$ at random and calculate $U_i = u_i P$.
   (b) Calculate $h_i^{(3)} = H_3(m_i, PID_i, PK_i, PK_{kgc}, t_i)$ and $h_i^{(4)} = H_4(m_i, PID_i, PK_i, U_i, t_i)$.
   (c) Compute $s_i = u_i + d_i h_i^{(3)} + x_i h_i^{(4)}$ and set the signature $\sigma_i = (U_i, s_i)$.
6. Single Signature Verification: Given $\{PID_i, PK_i, m_i, \sigma_i, T_i, t_i\}$, the verifier checks the freshness of t_i. Then it recovers $h_i^{(2)} = H_2(PID_i, R_i, PK_{kgc}, X_i, T_i)$, $h_i^{(3)} = H_3(m_i, PID_i, PK_i, PK_{kgc}, t_i)$, and

$h_i^{(4)} = H_4(m_i, PID_i, PK_i, U_i, t_i)$. It accepts the signature if $s_iP = U_i + h_i^{(3)}(R_i + h_i^{(2)}P_{kgc}) + h_i^{(4)}X_i$ and rejects otherwise.

Appendix A.2 Security Analysis to [22]

In [22], Yue et al. stated that their design is secure against both Type 1 and Type 2 attackers. Here, we show that a Type 1 attacker $\mathscr{F}_1$ possesses the capability to produce a verifiable signature for any fraudulent message, thereby compromising the unforgeability property inherent in their cryptographic construction. Let the vehicle V_i with pseudonym PID_i be the target device attacked by $\mathscr{F}_1$. Given public parameters $ppa = \{\mathbb{G}, P, q, PK_{kgc}, PK_{ta}, H_i\}$ and V_i's public key $PK_i = (X_i, R_i)$, $\mathscr{F}_1$ can also access V_i's secret value x_i. Suppose that $\mathscr{F}_1$ wants to generate a forgery σ_i^* on a message m_i^* at time t_i, as shown in Fig. A1, $\mathscr{F}_1$ operates as follows:

(1) Compute $h_i^{(2)} = H_2(PID_i, R_i, PK_{kgc}, X_i, T_i)$ and $h_i^{(3)} = H_3(m_i, PID_i, PK_i, PK_{kgc}, t_i)$.

(2) Pick $\beta \in \mathbb{Z}_q^*$ at random and set $U_i^* = \beta P - h_i^{(3)}(R_i + h_i^{(2)}P_{kgc})$.

(3) Compute $h_i^{(4)*} = H_4(m_i^*, PID_i, PK_i, U_i^*, t_i)$ and $s_i^* = \beta + h_i^{(4)}x_i$.

(4) Set its forgery $\sigma_i^* = (U_i^*, s_i^*)$.

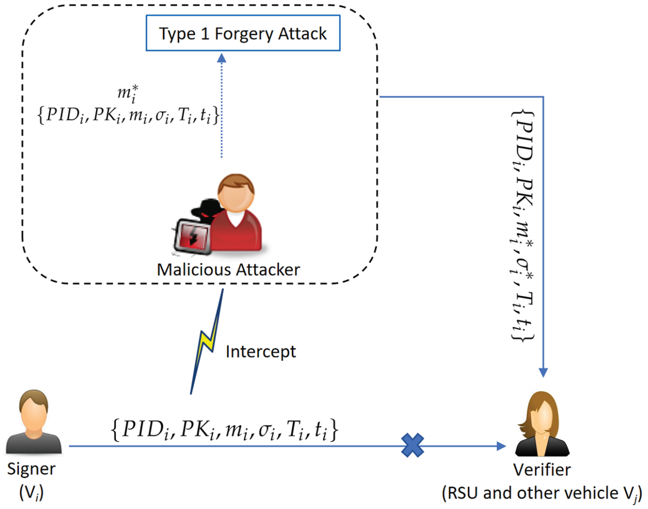
Type 1 Forgery Attack
m_i^*
$\{PID_i, PK_i, m_i, \sigma_i, T_i, t_i\}$
Malicious Attacker
$\{PID_i, PK_i, m_i^*, \sigma_i^*, T_i, t_i\}$
Intercept
$\{PID_i, PK_i, m_i, \sigma_i, T_i, t_i\}$
Signer
(V_i)
Verifier
(RSU and other vehicle V_j)

Figure A1: An example of the Type 1 attack

Now, the correctness of σ_i^* is checked by:

$$\begin{aligned} s_i^*P &= (\beta + h_i^{(4)}x_i)P = \beta P + h_i^{(4)}x_iP \\ &= U_i^* + h_i^{(3)}(R_i + h_i^{(2)}P_{kgc}) + h_i^{(4)}X_i. \end{aligned}$$

Due to the insecurity of the underlying CLS scheme, their CLAS construction cannot achieve unforgeability.

Appendix A.3 Improvement

In [22], a verifier checks a received signature through the equation $s_iP = U_i + h_i^{(3)}(R_i + h_i^{(2)}P_{kgc}) + h_i^{(4)}X_i$. However, due to the lack of binding between U_i and $h_i^{(3)}$, the attacker $\mathscr{F}_1$ can use the algebraic relationship in the equation to bypass the KGC's private key a (corresponding to $P_{kgc} = aP$).

To patch this vulnerability, a simple suggestion is to include U_i in computing $h_i^{(3)}$. That is, $h_i^{(3)} = H_3(m_i, PID_i, PK_i, PK_{kgc}, U_i, t_i)$. Following Yue et al.'s proof approach in [22], the modified scheme can be easily proven to be secure. The modification does not add any additional computational cost.

References

1. Daniel E, Tschorsch F. IPFS and friends: a qualitative comparison of next generation peer-to-peer data networks. IEEE Commun Surv Tutorials. 2022;24(1):31–52. doi:10.1109/comst.2022.3143147.
2. Benmoussa A, Kerrache CA, Lagraa N, Mastorakis S, Lakas A, Tahari AEK. Interest flooding attacks in named data networking: survey of existing solutions, open issues, requirements, and future directions. ACM Comput Surv. 2023;55(7):139:1–37. doi:10.1145/3539730.
3. Mazhar T, Irfan HM, Haq I, Ullah I, Ashraf M, Shloul TA, et al. Analysis of challenges and solutions of IoT in smart grids using AI and machine learning techniques: a review. Electronics. 2023;12(1):242. doi:10.3390/electronics12010242.
4. Mazhar T, Talpur DB, Shloul TA, Ghadi YY, Haq I, Ullah I, et al. Analysis of IoT security challenges and its solutions using artificial intelligence. Brain Sci. 2023;13(4):683. doi:10.3390/brainsci13040683.
5. Zhu F, Yi X, Abuadbba A, Luo J, Nepal S, Huang X. Efficient hash-based redactable signature for smart grid applications. In: ESORICS 2022. Copenhagen, Denmark; 2022 Sep 26–30. Vol. 13556. Cham, Switzerland: Springer; 2022. p. 554–73.
6. Boneh D, Gentry C, Lynn B, Shacham H. Aggregate and verifiably encrypted signatures from bilinear maps. In: EUROCRYPT 2003. Warsaw, Poland; 2003 May 4–8. Vol. 2656. Cham, Switzerland: Springer; 2003. p. 416–32.
7. Shen L, Ma J, Liu X, Wei F, Miao M. A secure and efficient ID-based aggregate signature scheme for wireless sensor networks. IEEE Internet Things J. 2017;4(2):546–54. doi:10.1109/jiot.2016.2557487.
8. Al-Riyami SS, Paterson KG. Certificateless public key cryptography. In: ASIACRYPT 2003. Taipei, Taiwan; 2003 Nov 30–Dec 4. Vol. 2894. Cham, Switzerland: Springer; 2003. p. 452–73.
9. Shim K. A secure certificateless signature scheme for cloud-assisted industrial IoT. IEEE Trans Ind Informatics. 2024;20(4):6834–43. doi:10.1109/tii.2023.3343437.
10. Yang W, Wang S, Mu Y. An enhanced certificateless aggregate signature without pairings for E-healthcare system. IEEE Inter Things J. 2021;8(6):5000–8. doi:10.1109/jiot.2020.3034307.
11. Aljarwan AZA, Ngadi MA. Review of certificateless authentication scheme for vehicular ad hoc networks. IEEE Access. 2025;13:100074–94. doi:10.1109/access.2025.3576926.
12. Mei Q, Xiong H, Chen J, Yang M, Kumari S, Khan MK. Efficient certificateless aggregate signature with conditional privacy preservation in IoV. IEEE Syst J. 2021;15(1):245–56. doi:10.1109/jsyst.2020.2966526.
13. Cahyadi EF, Su T, Yang CC, Hwang M. A certificateless aggregate signature scheme for security and privacy protection in VANET. Int J Distributed Sens Networks. 2022;18(5). doi:10.1177/15501329221080658.
14. Cui J, Zhang J, Zhong H, Shi R, Xu Y. An efficient certificateless aggregate signature without pairings for vehicular ad hoc networks. Inf Sci. 2018;451–452:1–15. doi:10.1016/j.ins.2018.03.060.
15. Kamil IA, Ogundoyin SO. An improved certificateless aggregate signature scheme without bilinear pairings for vehicular ad hoc networks. J Inf Secur Appl. 2019;44(1):184–200. doi:10.1016/j.jisa.2018.12.004.
16. Xu G, Zhou W, Sangaiah AK, Zhang Y, Zheng X, Tang Q, et al. A security-enhanced certificateless aggregate signature authentication protocol for InVANETs. IEEE Netw. 2020;34(2):22–9. doi:10.1109/mnet.001.1900035.
17. Zhu F, Yi X, Abuadbba A, Khalil I, Huang X, Xu F. A security-enhanced certificateless conditional privacy-preserving authentication scheme for vehicular ad hoc networks. IEEE Trans Intell Transp Syst. 2023;24(10):10456–66. doi:10.1109/tits.2023.3275077.
18. Yang W, Fan J, Song K, Zheng Y, Zhang F. An efficient and practical conditional privacy-preserving aggregate authentication for vehicular ad-hoc networks. IEEE Trans Intell Transp Syst. 2024;25(12):20256–67. doi:10.1109/tits.2024.3474210.
19. Zhu D, Guan Y. Secure and lightweight conditional privacy-preserving identity authentication scheme for VANET. IEEE Sensors J. 2024;24(21):35743–56. doi:10.1109/jsen.2024.3431557.

20. Zhu F, Hu Y, Ren Y, Han B, Yang X. Public-Key replacement attacks on lightweight authentication schemes for resource-constrained scenarios. Cyber Secur Applicat. 2025;3:100102. doi:10.1016/j.csa.2025.100102.
21. Verma RK, Khan AJ, Kashyap SK, Chande MK. Certificateless aggregate signatures: a comprehensive survey and comparative analysis. J Univers Comput Sci. 2024;30(12):1662–90. doi:10.3897/jucs.116249.
22. Yue Q, Jiang W, Lei H. A lightweight certificateless aggregate signature scheme without pairing for VANETs. Sci Rep. 2025;15(1):23663. doi:10.1038/s41598-025-08656-1.
23. Wang C, Wu H, Gan Y, Zhang R, Ma M. ECAE: an efficient certificateless aggregate signature scheme based on elliptic curves for NDN-IoT environments. Entropy. 2025;27(5):471. doi:10.3390/e27050471.
24. Xu F, Liu S, Yang X. An efficient privacy-preserving authentication scheme with enhanced security for IoMT applications. Comput Commun. 2023;208:171–8. doi:10.1016/j.comcom.2023.06.012.
25. Pointcheval D, Stern J. Security arguments for digital signatures and blind signatures. J Cryptol. 2000;13(3):361–96. doi:10.1007/s001450010003.
26. Sasdrich P, Güneysu T. Implementing Curve25519 for side-channel-protected elliptic curve cryptography. ACM Trans Reconfigurable Technol Syst. 2015;9(1):3:1–15. doi:10.1145/2700834.
27. Tanksale V. Efficient elliptic curve diffie-hellman key exchange for resource-constrained IoT devices. Electronics. 2024;13(18):3631. doi:10.3390/electronics13183631.
28. Katz J, Lindell Y. Introduction to modern cryptography. 2nd ed. Philadelphia, PA, USA: CRC Press; 2014.