PAPER Special Section on Next-generation Security Applications and Practice

An Efficient Public Verifiable Certificateless Multi-Receiver Signcryption Scheme for IoT Environments*

Dae-Hwi LEE[†], ***Nonmember***, **Won-Bin KIM**[†], ***Student Member***, **Deahee SEO**[††], ***and*** **Im-Yeong LEE**[†a)], ***Nonmembers***

SUMMARY Lightweight cryptographic systems for services delivered by the recently developed Internet of Things (IoT) are being continuously researched. However, existing Public Key Infrastructure (PKI)-based cryptographic algorithms are difficult to apply to IoT services delivered using lightweight devices. Therefore, encryption, authentication, and signature systems based on Certificateless Public Key Cryptography (CL-PKC), which are lightweight because they do not use the certificates of existing PKI-based cryptographic algorithms, are being studied. Of the various public key cryptosystems, signcryption is efficient, and ensures integrity and confidentiality. Recently, CL-based signcryption (CL-SC) schemes have been intensively studied, and a multi-receiver signcryption (MRSC) protocol for environments with multiple receivers, i.e., not involving end-to-end communication, has been proposed. However, when using signcryption, confidentiality and integrity may be violated by public key replacement attacks. In this paper, we develop an efficient CL-based MRSC (CL-MRSC) scheme using CL-PKC for IoT environments. Existing signcryption schemes do not offer public verifiability, which is required if digital signatures are used, because only the receiver can verify the validity of the message; sender authenticity is not guaranteed by a third party. Therefore, we propose a CL-MRSC scheme in which communication participants (such as the gateways through which messages are transmitted) can efficiently and publicly verify the validity of encrypted messages.

key words: *IoT, lightweight cryptographic system, certificateless signcryption, multi-receiver signcryption, public verifiability*

1. Introduction

The number of IoT devices connected to the Internet is increasing rapidly; many more services are emerging. Recent IoT services include smart cities and smart meters, first exemplified by smart homes [1]–[4]. Smart metering is a key element of smart grids, as a representative IoT service that monitors home energy consumption in real time. The "duplex communication structure" can also issue power supply or cut-off commands [5]–[8].

Given the proliferation of various types of IoT services, the importance of security is increasing. IoT services affect our daily lives and security breaches can create serious problems. Taking the smart metering environment as an example, leakage of sensitive information, such as personal power usage, may make it possible to cut off power to the home by executing a command that stops the power supply [7], [9]. Therefore, in IoT environments such as smart metering ones, security should never be neglected when small devices are connected. However, it is difficult to apply existing security technology such as public key infrastructure (PKI) to IoT environments in which both small and large devices participate. Existing PKI-based cryptographic algorithms are not suitable for IoT environments because they have large overheads associated with the management of certificates or keys. Lightweight public key cryptosystems for IoT environments, such as certificateless public key cryptography (CL-PKC), are constantly developing [10]–[16]. As encryption is needed to protect the data of individuals or organizations that are transmitted in the IoT environment, and as various devices are interconnected, authentication of both the object and the data transmitted by that object is required. Therefore, it is essential to consider encryption and signatures in the IoT environment.

The signcryption technique, which combines encryption and signing into one logical process, was proposed by Zheng in 1997 [17]. Conventionally, to provide the required confidentiality and signature functions, a data signature-then-encryption scheme is used, in which encryption is performed after signing. In this scheme, only the receiver can verify the validity of the signature because it must first decrypt the message. Therefore, public verifiability is not available, which is a basic requirement of a digital signature. However, in some environments, it is necessary to ensure public verifiability because if signcryption messages are forwarded at some stage, it is essential to be able to check the validity of messages transmitted by the intermediate object to improve service reliability [18]. Also, if a sender wishes to transmit a message to several receivers, each must receive a unique signcryption message. Current multi-receiver signcryption (MRSC) technologies seek to improve this situation. The MRSC scheme provides security suitable for the communication structure of the IoT environment. In the smart metering environment of Fig. 1, if MDMS needs to send a message requesting power consumption data to each household's smart meter, it will have to individually

Manuscript received February 10, 2021.
Manuscript revised May 19, 2021.
Manuscript publicized July 14, 2021.

[†]The authors are with Soonchunhyang University, South Korea.
[††]The author is with Sangmyung University, South Korea.

*This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-019R1A2C1085718) and the Soonchunhyang University Research Fund and the Republic of Korea's MSIT (Ministry of Science and ICT), under the High-Potential Individuals Global Training Program (2021-0-01516) supervised by the IITP (Institute of Information and Communications Technology Planning & Evaluation).

a) E-mail: imylee@sch.ac.kr (Corresponding author)

DOI: 10.1587/transinf.2021NGP0012

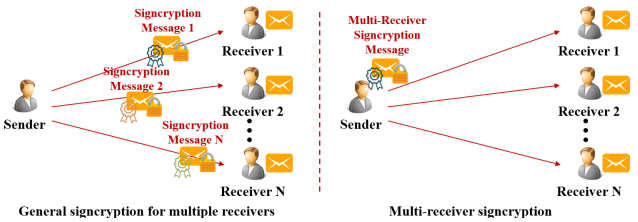

Fig. 1 Comparison of (a) general signcryption for multiple receivers and (b) multi-receiver signcryption.

encrypt the request message using each household's public key. If the MRSC scheme is applied to a smart metering environment, MDMS generates and transmits a multi-receiver signcryption message to be transmitted to a group of smart meters, rather than individually encrypting and transmitting messages for end-to-end communication with smart meters. Through this, the MDMS server does not have to perform end-to-end communication as much as the number of receivers, and the receiver (smart meter) can easily decrypt the message with own private key. In this study, we developed an MRSC technology that provides CL-PKC-based public verifiability. The scheme also ensures third-party sender authenticity. It is possible to verify the validity of the signcryption message transmitted by the object that delivers the message but, to preserve confidentiality, the object cannot read the original message. In addition, we design a protocol appropriate for a multi-receiver environment; the sender can multicast and transmit only one signcryption message, rather than creating individual messages to be sent to multiple receivers. In particular, our scheme is immune to security threats, such as the public key substitution attacks that threaten CL-based encryption technology. Finally, we do not use pairing operations that require relatively large amounts of computation.

2. Related Work

2.1 CL-PKC

In 2003, Al-Riyami developed a CL-PKC to solve the key escrow problem of the ID-based cryptography (IBC) [10]. The CL-PKC does not generate all of the public and private keys (unlike the KGC); rather, only partial keys are returned to users. These are termed partial secret keys; each user generates a complete key pair using the partial secret key. As the complete key pair is generated by the user, the key escrow problem is solved. Also, the keys are smaller than those of existing PKI, and the overhead associated with public key verification is no longer present; the CL-PKC is thus appropriate for lightweight, low-power environments, such as IoT environments. CL-PKCs find applications in various fields of cryptography, including authentication, key agreement, signing, and encryption. The major difference between the CL-PKC and PKI is the absence in the former of a certificate that verifies the public key for a user; this explains the use of the word "certificateless".

2.2 Signcryption

Signcryption is a cryptographic technology that ensures message integrity, non-repudiation, and confidentiality. In a typical PKI, the sender performs encryption using the receiver's public key and creates a signature employing a private key. The receiver verifies the signature using the sender's public key, and then employs confidentiality and signature functions (in the form of signature-then-encryption) that decrypt the original message using the receiver's private key. In signcryption, the 'signcrypt' operation performs signing and encryption using the sender's private key and the receiver's public key simultaneously; the receiver performs the 'unsigncrypt' operation, which decrypts the message and verifies its validity using the receiver's private key and the sender's public key.

Zheng was the first to propose signcryption, in 1997; signing and encryption were combined into a single logical process [17]. Zheng showed that the operation cost of signcryption was lower than that of a signature-then-encryption scheme that first applied signatures and only then encrypted. Since that time, signcryption has been applied to various aspects of public key cryptography. However, the signature provided by general signcryption technology is not public verifiable; this is an essential feature of a digital signature because, when seeking to verify a signcryption signature, the message must first be verified. This can be performed only by the receiver identified by the sender when encrypting. Therefore, in general signcryption, only the receiver chosen by the sender can verify signature validity. A recent study suggested that intermediate objects or third parties should be able to check the validity of transmitted messages, to improve the reliability of services in environments where data are received and then forwarded [18]–[21]. For example, in an environment where data from a smart meter must pass through a gateway during transmission to an meter data management server (MDMS), the content of the message transmitted by the gateway would remain unknown, but sender authenticity and validity would be verified. This meets the public verifiability requirement of a signature. In this paper, a scheme ensuring public verifiability is proposed in Sect. "Proposed CL-MRSC Scheme".

2.3 Existing CL-MRSC Schemes

In 2002, Malone et al. [22] proposed a scheme using ID-based encryption to implement signcryption, and CL-based signcryption (CL-SC) using CL-PKC was proposed in 2008 by Barbosa et al. [23] Since then, various CL-SC techniques have been developed [20], [21], [23]–[25]. However, the general CL-SC technique considers an end-to-end environment. When there are multiple receivers, the senders must perform as many signcrypt operations as there are receivers. Thus, CL-MRSC schemes allowing for multiple receivers of a single message have been proposed by Selvi et al. [26] and others, who focused on various security requirements [27]–

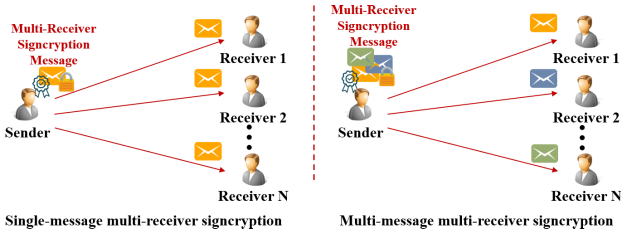

Fig. 2 Comparison of (a) single-message multi-receiver signcryption and (b) multi-message multi-receiver signcryption.

[32]. Figure 1 compares general signcryption and MRSC. As sensor devices are grouped in an IoT environment, CL-MRSC can be applied by sending commands to all devices. In MRSC, the methods of transmission of single and multiple messages differ. When the sender wants to send the same message to a set of receivers, it creates a signcryption message (as shown on the (a) of Fig. 2) and then sends it. When the sender wants to send a different message to each receiver, a multi-message MRSC (as shown on the (b) of Fig. 2) is used. Single-message MRSC includes only one message and one tag for signature verification by all receivers. MRSC allows messages to be sent to multiple receivers as appropriate, although the message sizes are relatively large because both the numbers of messages and verification tags increase in proportion to the number of receivers.

In existing CL-MRSC schemes, only the KGC that issues the partial secret key, and the sets of senders and receivers, participate as objects. However, recent schemes have included public verifiability by a third party; we compare and analyze these schemes here.

Wang et al. [27] proposed a CL-MRSC scheme for downlink multicast transmission between an MDMS server and a smart meter in an ambient intelligence (AMI) environment. A control message generated by the MDMS server is signed and transmitted to each smart meter, allowing for public signature verification. Encryption/decryption are achieved using the XOR key. However, there is a disadvantage that only single-message can be transmitted to multiple smart meters, and the equation of the encrypted message is configured so that only single-receiver rather than multiple receivers can be decrypted due to a protocol error.

Pang et al. proposed three types of schemes [28]–[30]. The scheme in Pang et al. [28], published in 2018, is a Pang's first CL-MRSC scheme based on elliptic curve cryptography (ECC) ensuring receiver anonymity without bilinear pairing (which requires a lot of computation). The Pang's second scheme [29] is uses MRSC for multi-messaging, and Pang's third scheme [30] does not use a secure channel during key extraction for single messages. All three schemes are susceptible to public key replacement attacks. As the Pang's first and third schemes support only single-messaging, the same message is transmitted to all receivers. In Pang's second scheme, the absence of a secure channel facilitates simple encryption and key distribution, which facilitates multi-messaging, but does not protect against public key replacement attacks (as mentioned above).

Qiu et al. [31] developed a multi-message CL-MRSC for heterogeneous, smart mobile IoT environments. Not only senders and receivers, but also a gateway, participate in communication; the receiver partly decrypts the message. However, gateway decryption is insignificant; any participant can engage in it. An IBC and CL-PKC are used simultaneously, in addition to a reliable KGC and a private key generator (PKG). This burdens the infrastructure. In particular, as the key generated by the PKG is sent to each sender via a secure channel, any such channel must still be configured, which confers no advantage compared to existing schemes.

Ming et al. [32] developed multi-message MRSC for healthcare IoT environments. In Pang et al. [30], an anonymous identifier was used to ensure sender anonymity. Unlike existing signcryption, which encrypts a message using an XOR operation or a symmetrical key, the key decryption function $f(x)$ allow receivers to create their own messages rather than a key. The receiver directly extracts the message using a private key without any need for decryption. However, non-repudiation is weak given that an anonymous identifier is used; the system is susceptible to public key replacement attack. As sender authenticity must be ensured when messages are transmitted within IoT service environments, the use of anonymous identifiers may become problematic if there are any problems. Also, public key replacement attacks are possible.

2.4 Security Model of CL-MRSC

Several security vulnerabilities are evident in a CL-MRSC scheme. The public key used in the CL-PKC does not include a certificate so it is impossible to authenticate a user. Thus, non-repudiation functions are not available, and a public key replacement attack is possible; this is the typical security attack experienced by a CL-PKC. A CL-MRSC public key replacement attack seeks to decrypt a message or forge a signature, by replacing the public key of the sender with a public key generated by the attacker, to decrypt or forge the signature of a message sent by a sender to receivers. This is done using the public key that the attacker creates, bypassing verification of the signature generated by the sender. Such an attack is possible because no certificate that verifies the sender's public key is available.

In addition, in a CL-MRSC, a sender who creates a signcryption message has been issued a partial key by the KGC. When the KGC receives the sender's identifier, it is common to securely transmit the signer's partial key pair to the user using the *Extract-Partial-Key* process. The *Extract-Partial-Key* algorithm is one of the important algorithms of the CL-MRSC scheme, and is a process in which system participants such as senders and receivers receive partial key from KGC. The KGC generates and stores partial keys for multiple receivers; attacks that forge signatures or undermine the confidentiality of messages using the partial keys

can occur. There are two types of CL-PKC attacks: public key replacement and KGC partial key generation attacks. These two attack types are distinguished by describing the attackers as A_I and A_{II}. A_I can arbitrarily replace the public key of a legitimate user without using the system master key. A_{II} cannot replace user's public keys, but knows the KGC master secret key and can calculate partial user keys. In this paper, security using elliptic curve discrete logarithm problem (ECDLP) and computational Diffie-Hellman (CDH) problem based on elliptic curve will be provided, and the definition of elliptic curve cryptography and mathematical problems based on it is as follows.

Definition 1. Elliptic Curve Cryptography (ECC). Let F_q denote a finite field with a large prime order q, and let E_q denote an elliptic curve on F_q which is specified by the equation: $y^2 = x^3 + ax + b \ (mod\ p)$ where $a, b \in F_q$ and $(4a^3 + 27b^2) \ mod \ p \neq 0$. Let O denote a point of infinity on the elliptic curve, form the additive cyclic group G of the elliptic curve under the computation of point addition $T = U + V$ for $U, V \in G$ defined on the basis of a chord-and-tangent rule. P is a generator of circulating group G, and $x \cdot P$ in $x \in Z_q^*$ is equal to $x \cdot P = (P + P + \ldots + P)$ (x times) defined as scalar multiplication.

Definition 2. Elliptic Curve Discrete Logarithm Problem (ECDLP). The elliptic curve cryptography was designed based on the elliptic curve discrete logarithm problem (ECDLP). ECDLP is a problem of finding the integer $x \in Z_q^*$ at $Q = x \cdot P$ given P, Q.

Definition 3. Computational Diffie-Hellman (CDH) Problem. Given the generator P above group G and two arbitrary points $a \cdot P, b \cdot P \in G$, the problem of calculating $a \cdot b \cdot P$ is the computational Diffie-Hellman problem.

2.5 Security Requirements

Integrity. The most important feature of the digital signature function provided by signcryption is integrity. In an IoT environment, all participating objects transmit and receive data using a wireless communication network. It is very important to ensure that messages have not been forged by attackers.

Confidentiality. In signcryption, the encryption function ensures confidentiality. In MRSC, only receivers identified by a sender can decrypt messages; an unrecognized receiver cannot decrypt. Public validation by a third party ensures integrity, but message decryption is not permitted.

Unforgeability. It should be impossible to create a pair of forged signatures or messages that might be verified when a third party (including the KGC) uses a sender's public key. The KGC must not be able to identify the full key pair of the sender's signature when generating a partial key, and must also be unable to generate a forged signature that can be verified via public messages.

Sender Authenticity. Sender authenticity is the opposite of sender anonymity; it must be possible to confirm that a signcryption message was sent by a specific sender. This ensures non-repudiation, authenticating a sender if a dispute arises in an IoT service environment.

Public Verifiability. In an IoT service environment, such as smart metering, it is important to ensure end-to-end integrity between the sender and receiver. This requires a signature verification function in the transmission path of the service; a gateway or server may be involved. Although the message must remain confidential, its validity should be verifiable by anyone to ensure sender authenticity. If a dispute arises, it is necessary to confirm that the message was sent by a certain sender.

3. Proposed CL-MRSC Scheme

Existing schemes introduced in Sect. 2.2 have disadvantages such as not providing public verifiability, or being able to transmit only to single-receiver even if provided. This paper proposes an improved public verifiable multi-receiver signcryption scheme. We develop a public verifiable CL-MRSC scheme for IoT services, such as smart metering. Figure 3 shows a diagram of the scheme. Signcryption is used to ensure confidentiality, integrity, and the signing of data transmitted in the IoT environment. To minimize computational cost, the multi-receiver component does not use a pairing operation. The scheme is a multi-message MRSC scheme for transmission of different messages to each receiver, which incorporates sender authentication and public verifiability of encrypted messages by third parties.

And, in the proposed scheme, a partial key is generated to prevent a public key replacement attack. In general, the form of Schnorr signature of partial secret key generated in KGC of CL-PKC is $r_{ID}+H(ID_{ID}, r_{ID} \cdot P) \cdot msk$, and in this paper, user's public key is added to this signature bind with the public key (msk is the master key of KGC, $r_I D$ is a randomly selected value for user). The form of the partial secret key generated by KGC becomes $r_{ID}+H(ID_{ID}, r_{ID} \cdot P, pu_{ID}) \cdot msk$, and it can be verified by binding the identifier and the public key. Therefore, the user first creates his/her own key pair and transmits the identifier and public key to KGC. Afterwards, KGC generates a partial key using the received identifier and public key and delivers it to the user. The CL-MRSC

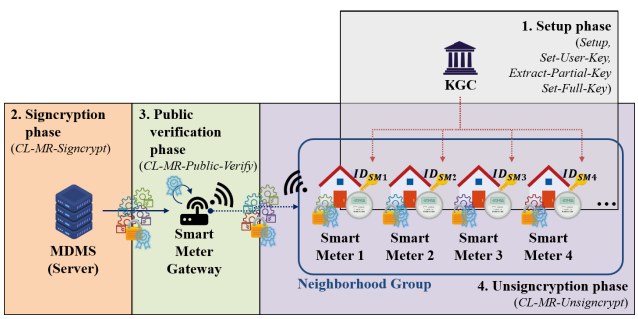

Fig. 3 The proposed CL-MRSC scheme.

scheme proposed in this paper consists of 7 algorithms, and each algorithm is as follows.

- *Setup*: The KGC creates a master secret key *msk* and a master public key P_{Pub} by inputting the security parameter k, and creates and discloses the public parameter *params*.
- *Set-User-Key*: The participant (user) generates a private verification key sv_{ID} and public key pu_{ID} using the public parameter *params* and its identifier *ID*. This is not yet a full key pair, instead serving as a verification key pair after the generation of a partial secret key by the KGC.
- *Extract-Partial-Key*: The KGC generates a user's partial private key z_{ID} and a partial public key R_{ID} by inputting the public parameter *params*, the master secret key *msk*, the user's personal identifier *ID*, and the public key pu_{ID} for verification; and creates a partial key pair $D_{ID} = (R_{ID}, z_{ID})$ that is then delivered to the user.
- *Set-Full-Key*: The user sets the full key pair (Pr_{ID}, Pu_{ID}) by inputting the public parameter *params*, the partial key pair D_{ID} received from the KGC, and the public key sv_{ID} for verification.
- *CL-MR-Signcrypt*: Among the users who generated keys, the user who wants to send the message becomes sender S and creates a signcryption message using its private key. The input is the receiver set $(ID_{r1}, \ldots, ID_{rn}, Pu_{r1}, \ldots, Pu_{rn})$; the message set to be sent to receivers $(m_{r1}, \ldots, m_{rn})$; and the sender's full private key Pr_S. These generate the signcryption message δ_S and send it to the receivers.
- *CL-MR-Public-Verify*: Among the objects participating in the network, a third-party validator other than the receiver, such as a gateway, cannot read encrypted messages but must be able to verify their validity. The verifier confirms validity by inputting the sender's full public key Pu_S and the public parameter *params*.
- *CL-MR-Unsigncrypt*: Receiver *ri* decrypts and verifies the signcryption message using the public parameter *params*, its own private key Pr_{ri}, and the sender's public key Pu_S, to finally obtain the message m_{ri} of the sender.

The setup phase includes the *Setup*, *Set-User-Key*, *Extract-Partial-Key*, and *Set-Full-Key* algorithms, while the signcryption phase includes the *CL-MR-Signcrypt* algorithm. In the public verification and unsigncryption phases, the *CL-MR-Public-Verify* and *CL-MR-Unsigncrypt* algorithms are run, respectively.

The system parameters used in this paper are as follows.

- k: A security parameter used in the setup phase and used to create a master private key and public key pair, and public parameters
- ID_*: The identifier of the participating entity
- pu_*, sv_*: The verification key pair of participating entities
- Pu_*, Pr_*: The full key pair of participating entities
- E: The elliptic curve with prime order q on cyclic group G
- P: The generator point on the elliptic curve E
- $D_* = (R_*, z_*)$: The partial key of the entity (partial public key and partial private key pair)
- *msk*: The KGC's master secret key
- P_{Pub}: The KGC's master public key $(P_{Pub} = msk \cdot P)$
- $H_1(\cdot)$: Cryptographic one-way hash function $(\{0,1\}^* \times G \times G \rightarrow Z_q^*)$
- $H_2(\cdot)$: Cryptographic one-way hash function $(G \rightarrow Z_q^*)$
- $H_3(\cdot)$: Cryptographic one-way hash function $(G \times G \rightarrow Z_q^*)$
- $H_4(\cdot)$: Cryptographic one-way hash function $(\{0,1\}^* \times G \times G \times G \rightarrow Z_q^*)$
- $H_5(\cdot)$: Cryptographic one-way hash function $(\{0,1\}^* \times G \times Z_q^* \times Z_q^* \times \ldots \times Z_q^* \times G \rightarrow Z_q^*)$

3.1 Setup Phase

The setup phase includes four of the seven algorithms of the proposed scheme. First, the KGC creates an initial parameter by executing the *Setup* algorithm using security parameter k. The KGC generates a master secret key *msk* and a master public key P_{Pub}. After creating public parameters, the KGC generates partial keys for all participants. Those who wish to engage with the service generate individual verification key pairs using the *Set-User-Key* algorithm and send the identifier and public key to the KGC for verification. The KGC runs the *Extract-Partial-Key* algorithm to create the private/public key pair. The KGC transmits the partial key to each participant, who then creates their own full key pair using the *Set-Full-Key* algorithm.

Step 1. The KGC selects security parameter k and executes the *Setup* algorithm as follows. First, a master secret key *msk* is created and the master public key $P_{Pub} = msk \cdot P$ is calculated. Next, the public parameter *params* is calculated as follows.

$$params = \{G, q, E, P, P_{Pub}, H_1, H_2, H_3, H_4, H_5\} \quad (1)$$

Step 2. Participants who want to generate partial keys using the KGC generate individual public and private key pairs pu_{ID}, sv_{ID} using the following *Set-User-Key* algorithm. Participants select $x_{ID} \in Z_q^*$ and calculate $pu_{ID} = x_{ID} \cdot P$. The sv_{ID} is $sv_{ID} = x_{ID}$.

Step 3. Participants transmit their identifiers *(IDs)* and the public key pu_{ID} to the KGC for verification; this triggers execution of *Extract-Partial-Key*, which generates partial keys for all participants. The KGC runs the *Extract-Partial-Key* algorithm to generate partial keys. The KGC selects $r_{ID} \in Z_q^*$ and generates $R_{ID} = r_{ID} \cdot P$. Then, the signature for the public key (a portion of each participant's private key) $z_{ID} = r_{ID} + msk \cdot H_1(ID, pu_{ID}, R_{ID})$ is generated. The KGC

transmits the partial key pair $D_{ID} = (R_{ID}, z_{ID})$ to each participant via a secure channel.

Step 4. Participants who received partial keys D_{ID} execute *Set-Full-Key* and create personal full private keys Pr_{ID} and full public keys Pu_{ID}, as follows.

$$Pr_{ID} = sv_{ID} + z_{ID}, \tag{2}$$

$$Pu_{ID} = (pu_{ID}, R_{ID}, Z_{ID} = z_{ID} \cdot P) \tag{3}$$

3.2 Signcryption Phase

The signcryption phase involves the *CL-MR-Signcrypt* algorithm of the seven algorithms of the proposed scheme. Of the various participants, the one who wants to send a message becomes sender S, signs a message using its key, and creates a cryptogram using the public key and identifier of the receiver set. In particular of the proposed scheme, only designated receivers can decrypt the encrypted value, while generating a signcryption message with a signature that allows public verification of the message.

Step 1. Sender S selects an ephemeral secret key $t_S \in Z_q^*$, and calculates an ephemeral public key $T_S = t_S \cdot P$.

Step 2. Sender S calculates h_{ri} using the public keys of all receivers, as follows, and transmits the signcryption message to the receiver set $RL_{ID} = (ID_{r1}, ID_{r2}, \ldots, ID_{rn})$.

$$h_{ri} = H_1(ID_{ri}, pu_{ri}, R_{ri}) \qquad (i = 1, 2, \ldots, n) \tag{4}$$

Step 3. Sender S creates E_{ri}. This is used to generate the decryption keys of the receivers. An E_{ri} can only be generated by a sender S who knows t_S via the elliptic curve discrete logarithm problem (ECDLP) and the receiver ri who has the private key Pr_{ri}.

$$E_{ri} = t_S \cdot (pu_{ri} + Z_{ri}) \tag{5}$$

Step 4. Sender S calculates e_{ri}, e_S and a public verifiable signature σ_S, as follows.

$$e_{ri} = H_2(E_{ri}) \qquad (i = 1, 2, \ldots, n) \tag{6}$$

$$e_S = H_3(Pu_S, T_S) \tag{7}$$

$$\sigma_S = t_S + h_S \cdot Pr_S \ (h_S = H_4(ID_S, pu_S, Z_S, T_S)) \tag{8}$$

Step 5. Sender S sends the e_S that it has created to the e_{rn+1} decoding the values of receivers $(e_{r1}, e_{r2}, \ldots, e_{rn})$, converts these to sets $(e_{r1}, e_{r2}, \ldots, e_{rn}, e_{rn+1})$, computes the polynomial $f(x)$, and generates a coefficient set C_S.

$$\begin{aligned} f(x) = &\prod_{i=1, i\neq 1}^{n+1} \frac{(x - e_{ri})}{(e_1 - e_{ri})} \cdot m_{r1} + \ldots \\ &+ \prod_{i=1, i\neq n}^{n+1} \frac{(x - e_{ri})}{(e_{rn} - e_{ri})} \cdot m_{rn} \\ &+ \prod_{i=1, i\neq n+1}^{n+1} \frac{(x - e_{ri})}{(e_{rn+1} - e_{ri})} \cdot \sigma_S \end{aligned}$$

$$= g_n x^n + g_{n-1} x^{n-1} + \ldots + g_1 x + g_0 \ (mod\ p) \tag{9}$$

$$C_S = (g_n, g_{n-1}, \ldots, g_1, g_0) \tag{10}$$

Step 6. Sender S calculates the signature V_G that can link to T_S to verify the integrity of the C_S that it has created, as follows.

$$V_G = H_5(ID_S, Pu_S, C_S, T_S) \cdot Pr_S \tag{11}$$

Step 7. The sender's signcryption message δ_S reads as follows.

$$\delta_S = (T_S, C_S, V_G) \tag{12}$$

3.3 Public Verification Phase

In the public verification phase, the validity of an encrypted message is verified at an intermediate point (an IoT service such as a gateway). Throughout this phase, the proposed scheme allows for public verifiability (this is a requirement of a digital signature) and filters forged signatures or ciphertexts, thus improving reliability. Below, the intermediate communication point is assumed to be a gateway and we describe an appropriate protocol. The *CL-MR-Public-Verify* algorithm is included among the seven algorithms of the proposed scheme.

Step 1. Sender S first transmits signcryption message δ_S and RL_{ID} to the gateway with the intention that the gateway should forward δ_S to the receivers.

Step 2. The gateway constructs the polynomial $f(x)$ via C_S, generates $e_S^{'}$ using the sender's public key Pu_S, and calculates $\sigma_S^{'}$.

$$e_S^{'} = H_3(Pu_S, T_S) \tag{13}$$

$$f(e_S^{'}) = g_n e_S^n + \ldots + g_1 e_S + g_0 = \sigma_S^{'} \tag{14}$$

Step 3. The gateway verifies the validity of $\delta_S^{'}$ and V_G as follows. If verification is successful, δ_S is multicast to the receiver set; otherwise, the message is discarded.

$$\begin{aligned} &(\sigma_S^{'} + V_G) \cdot P \\ &? = T_S + (h_S + H_5(ID_S, Pu_S, C_S, T_S)) \cdot (pu_S + Z_S) \end{aligned} \tag{15}$$

3.4 Unsigncryption Phase

In the unsigncryption phase, a receiver ri can calculate σ_S and V_G using δ_S received from the gateway, to check the validity of the entire message, and can calculate E_{ri} to obtain message m_{rn}. This involves the last *CL-MR-Unsigncrypt* algorithm (there are seven algorithms in all).

Step 1. Receiver ri constructs the polynomial $f(x)$ via C_S, and calculates $\sigma_S^{''}$ by generating $e_S^{''}$ using the sender's public key Pu_S.

$$e_S^{''} = H_3(Pu_S, T_S) \tag{16}$$

$$f(e_S^{''}) = g_n e_S^n + \ldots + g_1 e_S + g_0 = \sigma_S^{''} \quad (17)$$

Step 2. Receiver ri verifies the validity of $\sigma_S^{''}$ and V_G as follows. If verification is successful, E_{ri} is calculated as follows; otherwise, the message is discarded.

$$(\sigma_S^{''} + V_G) \cdot P$$
$$? = T_S + (h_S + H_5(ID_S, Pu_S, C_S, T_S)) \cdot (pu_S + Z_S) \quad (18)$$
$$E_{ri} = T_S \cdot Pr_{ri} \quad (19)$$

Step 3. Receiver ri calculates e_{ri} using E_{ri} and finally obtains message m_{ri} by using it as an input to the polynomial $f(x)$.

$$e_{ri} = H_2(E_{ri}) \quad (20)$$
$$f(e_{ri}) = g_n e_{ri}^n + \ldots + g_1 e_{ri} + g_0 = m_{ri} \quad (21)$$

4. Security Analysis of Proposed Scheme

The CL-MRSC proposed in this paper aims to solve various problems of existing CL-MRSC schemes used in IoT environments. In this chapter, we compare and analyze how the proposed scheme and existing schemes satisfy the security requirements. Table 1 shows the comparison of the proposed CL-MRSC scheme with the existing schemes corresponding to each security requirement.

4.1 Correctness of the Proposed Scheme

In the protocol introduced in Sect. 3, signcryption is performed using the full key pair of the sender, the identifier of the receiver, and the full key pair set. The validity of signing and encryption performed in the signcryption phase can be verified in both the public verification and unsigncryption phases. Prior to those phases, the validity of the sender's partial secret key z_S can be verified as follows.

$$z_S \cdot P = (r_S + msk \cdot H_1(ID, pu_{ID}, R_{ID})) \cdot P$$
$$= R_S + P_{Pub} \cdot H_1(ID, pu_{ID}, R_{ID}) \quad (22)$$

Using this equation, sender S can verify that the partial secret key was generated appropriately by the KGC using the public key P_{Pub}. The KGC can generate the value only by using the ECDLP. In addition, the signature values σ_S, V_G generated by Eqs. (8) and (12) can be verified using Eqs. (15) and (18), by a third party such as a gateway or a receiver; the mathematics are based on the ECDLP and CDH problems. Validity can be proved as follows:

$$\begin{aligned}
&(\sigma_S + V_G) \cdot P \\
&= (t_S + h_S \cdot Pr_S) \cdot P \\
&\quad + H_5(ID_S, Pu_S, C_S, T_S) \cdot Pr_S \cdot P \\
&= (T_S + h_S \cdot (sv_S + z_S) \cdot P) \\
&\quad + H_5(ID_S, Pu_S, C_S, T_S) \cdot (sv_S + z_S) \cdot P \\
&= T_S + h_S \cdot (pu_S + Z_S) \\
&\quad + H_5(ID_S, Pu_S, C_S, T_S) \cdot (pu_S + Z_S) \\
&= T_S + (h_S + H_5(ID_S, Pu_S, C_S, T_S)) \cdot (pu_S + Z_S)
\end{aligned} \quad (23)$$

Therefore, the value calculated by a third party or receiver using Eqs. (15) and (18) can be generated only by sender S who knows the full private key. Finally, for the receiver to acquire the message, E_{ri} must be calculated using the T_S obtained from the sender and the full private key Pr_{ri}. The equivalence of Eqs. (5) and (19) used to calculate E_{ri} can be proven as follows. Only the sender and receiver can calculate E_{ri} on the basis of the CDH problem.

$$\begin{aligned}
E_{ri} &= t_S \cdot (pu_{ri} + Z_{ri}) \\
&= t_S \cdot (sv_{ri} + z_{ri}) \cdot P \\
&= T_S \cdot Pr_{ri}
\end{aligned} \quad (24)$$

4.2 Integrity

The integrity of the CL-MRSC scheme that we propose can be verified using σ_S and V_G; these signatures publicly verify the validity of a session created using the ephemeral secret key t_S. A third party, such as a gateway, can engage in verification using Eqs. (15) and (18). The receiver can also cal-

Table 1 Security analysis of proposed scheme with existing schemes.

No.	Wang et al. [27]	Pang et al. [28]	Pang et al. [29]	Pang et al. [30]	Qiu et al. [31]	Ming et al. [32]	Proposed Scheme
(1)	Single-message for single receiver	Single-message	Multi-message	Single-message	Multi-message	Multi-message	Multi-message
(2)	O	X	X	O	O	O	O
(3)	O	X	X	O	O	O	O
(4)	O	△ Only receiver can authenticate sender	△ Only receiver can authenticate sender	△ Only receiver can authenticate sender	△ Only receiver can authenticate sender	X	O
(5)	O	X	X	X	X	△ Cannot sender validation	O

O: Offer, △: Partially offer, X: Not offer
(1): Structure of message, (2): Confidentiality, (3): Unforgeability, (4): Sender authenticity, (5): Public verifiability

culate σ_S and V_G to check the validity of the signature, and can extract the message using the T_S corresponding to the public key t_S and private key Pr_S of the receiver. The message set $(m_{r1}, m_{r2}, \ldots, m_{rn})$ can be calculated using the coefficient set C_S that constitutes the polynomial $f(x)$, and the integrity of the message set can be verified while confirming the integrity of C_S. Ultimately, in the proposed scheme, the integrity of the session depends on t_S, and verification of σ_S confirms that the sender generated t_S. Also, verification of V_G confirms that C_S was created by the user who created t_S. If the message is genuine, its integrity (in the sense that it was transmitted from S) can be checked using σ_S and V_G. If a signcryption message δ_S is forged by an attacker, σ_S cannot be obtained, and σ_S and V_G cannot be verified.

4.3 Confidentiality

CL-MRSC confidentiality refers to the requirement that it must be impossible for an attacker to decrypt a specific signcryption message sent by a sender. Confidentiality is divided into attacks by attacker A_I (who can replace the public key) and attacker A_{II} (a malicious KGC that is honest-but-curious). Even if the receiver's public key is replaced by A_I or the partial key is determined by A_{II}, the message must not be decryptable from the signcrypted message.

Confidentiality by adversary I. In a CL-MRSC, an attack on confidentiality succeeds if adversary A_I acquires message m_{ri} sent to a specific receiver ri. A_I can replace a participant's public key with a public key that it creates itself. If A_I acquires message m_{ri} via public key replacement, the attack is successful. In such an attack, the attacker replaces the public key of the receiver, and the sender creates a cryptogram using the new public key (which is generated by the attacker). A_I aims to attack the decryption key; if A_I can calculate E_{ri} using Eqs. (5) and (19), message m_{ri} can be obtained. The attacker can replace the receiver's full public key with $Pu'_{ri} = (pu'_{ri}, R'_{ri}, Z'_{ri})$, and the sender will then try to create a legitimate E_{ri} using pu'_{ri} and Z'_{ri}. However, in our proposed scheme, Z_{ri} is generated in the form of $Z_{ri} = z_{ri} \cdot P = R_{ri} + P_{Pub} \cdot H_1(ID_{ri}, pu_{ri}, R_{ri})$ for the full public key. In other words, as Z_{ri} is bound to the public key pu_{ri} and the identifier ID_{ri} used to verify ri, and can itself be verified with P_{Pub}, the public key will be invalid even if the attacker creates Pu'_{ri}. In practical terms, public key replacement is impossible, and E_{ri} can be calculated only by the sender who generated t_S using the CDH and the receiver who uses Pr_{ri} to obtain message m_{ri}. Our proposed scheme stymies A_I.

Confidentiality by adversary II. As attacker A_{II} knows msk, it knows all of the partial keys of the participants. As for A_I, E_{ri} must be computed if the goal is to attack confidentiality. If message m_{ri} can be decrypted using only a partial key, A_{II} succeeds. In particular, as the A_{II} attacker knows the partial key, the message m_{ri} can be decrypted if the value E_{ri} required for decryption is calculated using only the partial key. However, in the proposed scheme, E_{ri} is calculated using the CDH problem only by a sender who knows t_S of Eq. (5), or a receiver who knows Pr_{ri} of Eq. (19). Pr_{ri} is calculated as $Pr_{ri} = (sv_{ri}+z_{ri})$, and as only z_{ri} can be found using partial keys, it is impossible to infer sv_{ri}. Therefore, the proposed scheme prohibits decryption of message m_{ri} by attacker A_{II}, thus ensuring confidentiality.

4.4 Unforgeability

Signature unforgeability must be considered; an arbitrary message and signature pair that can be verified by an attacker must not be created. As for confidentiality, unforgeability is divided into attacks by attacker A_I (who seeks to engage in public key replacement) and attacks by attacker A_{II}, (a malicious KGC that is honest-but-curious). In our proposed scheme, the publicly verifiable signatures σ_S, V_G are used; neither value should be forgeable. If a pair of forged signatures σ'_S, V'_G corresponding to Eqs. (8) and (11), and a forged message m'_{ri}, can be generated and delivered to a specific receiver, the attacker succeeds in forging the signature.

Unforgeability by adversary I. If attacker A_I is to succeed in a forgery attack, the forged signatures σ'_S, V'_G must be appropriately verified by the receiver by replacing the sender's public key Pu_S with Pu'_S. To solve the public key replacement attack problem, CL-PKC explicitly authenticates the user. It is thus necessary to strengthen the binding of the public key to the identifier in the partial key. When verifying a public key or a signature signed with a private key, it is only necessary to confirm that the user has a public key or signature created with a key that was in turn created using a partial key received from the KGC. In other words, in the partial secret key z_S and partial public key R_S received by the sender from the KGC, R_S is the tag that verifies z_S; if it can be proved that the sender's public key Z_S created using z_S is a real sender's key, the public key is secure from replacement attacks. In the proposed scheme, if signature and message forgery by A_I is to succeed, a message m'_{ri} corresponding to σ'_S and V'_G must be generated. The formula used to generate the signature σ_S for message m_{ri} is $\sigma_S = t_S + h_S \cdot Pr_S$ and, as attacker A_I cannot know t_S or Pr_S, a valid σ_S cannot be generated. In addition, the equation used to generate signature V_G is $V_G = H_5(ID_S, Pu_S, C_S, T_S) \cdot Pr_S$ and the attacker A_I does not know Pr_S; therefore, a valid V_G cannot be created so the proposed scheme prohibits forgery by A_I.

Unforgeability by adversary II. If attacker A_{II} is to successfully mount a forgery attack, it must be able to obtain a partial key D_S through msk and create a forged signature σ'_S, V'_G that can be verified by Pu_S. If A_{II} creates σ'_S, V'_G and the receiver verifies it using the sender's public key Pu_S, the attack succeeds. Consider $\sigma_S = t_S + h_S \cdot Pr_S$. An attacker can generate only the partial secret key z_S of Pr_S, and thus cannot forge or induce σ'_S by forging the entire σ_S. The included sender's identifier and public key, and the ephemeral public key T_S of the session, cannot

be modified. In addition, $H_5(ID_S, Pu_S, C_S, T_S)$ in $V_G = H_5(ID_S, Pu_S, C_S, T_S) \cdot Pr_S$ includes internal parameters of the sender's public key and identifier, and the ephemeral public key. In addition, C_S (required for message decryption) cannot be forged, ensuring integrity. Similarly, as z_S (a part of Pr_S) cannot generate V_G' (a forged V_G), our proposed scheme ensures unforgeability by A_{II}.

4.5 Sender Authenticity and Public Verifiability

Given the security requirements described in Sect. 2, when a message is received by an object in the IoT service environment, it must be possible to publicly verify who sent the message. A digital signature meets this requirement. However, in existing signcryption systems, sender authenticity is not public because only the receiver who decrypts the message can verify its validity. In an IoT service environment, only the receiver (thus not a third object that facilitates message transmission) should be able to decrypt the message, but it is necessary to verify the validity of the message to determine who sent it; this improves service reliability. Schemes that verify the validity of an encrypted message public may or may not check sender authenticity. The Ming et al. [32] allows for public verification but not sender authenticity in a manner ensuring sender anonymity. In this paper, we show that it is possible to verify the integrity of the message while verifying σ_S and V_G, and also to authenticate the sender. The h_S of σ_S is a value generated as $h_S = H_4(ID_S, pu_S, Z_S, T_S)$, and its validity can be verified up to the sender's identifier and public key, and the ephemeral public key T_S used in the current session. In addition, using $H_5(ID_S, Pu_S, C_S, T_S)$ of V_G, it is possible to check the validity of C_S that decrypts the message and generates σ_S, including information on the sender and session. Both signatures are linked through the sender's identifier and public key and the ephemeral public key T_S used for the session. It is possible to publicly verify that the signature of the session using the T_S is generated by the sender with the current ID_S.

5. Efficiency Analysis of Proposed Scheme

To simulate our proposed scheme and other schemes, we used an Intel 3.50-GHz i5-4690 processor with 16 GB of memory and the Windows 10 operating system. To implement the ECC that afforded the same security level as a 1,024-bit RSA, a=1 and b was the Koblitz elliptical curve $y^2 = x^3 + ax + b \ (mod\ p)$, a 160-bit random prime defined in $F_{2^{160}}$.

5.1 Computation Cost

Table 2 compares the execution times of cryptographic operations, while Fig. 4 compares the efficiency of signcryption/unsigncryption operations between existing schemes and our proposed scheme. As is true of the existing MRSC schemes, our proposed CL-MRSC scheme enables message decryption and verification via unsigncryption using public parameters when the sender transmits a signcryption message to the receiver. The schemes [27]–[32] are efficient because they do not use pairing operations, and the schemes [27], [28], [30], [31] require very small amounts of computation but have the disadvantages described in Table 1.

The Wang et al. [27] satisfies the security requirements, but as signcryption can be used to generate only single messages, the number of messages is high. The Pang et al. [28], [30] and Qiu et al. [31] do not include public verification, so both confidentiality and integrity are under threat. The Pang et al. [29] is neither efficient nor secure. The Ming et al. [32] is more computationally efficient than the other schemes, but there is no public verification and both message and signature forgery are possible. Our proposed scheme is significantly less efficient than existing signcryption/unsigncryption schemes including public verification, but, importantly, it satisfies all security requirements while maintaining tolerable efficiency.

5.2 Communication Cost

We compared the cost of messages transmitted over the communication of the conventional and proposed schemes. Each element has a length of 20 bytes for G and Z_q^*, and the length of the AES symmetric key encryption message C_k

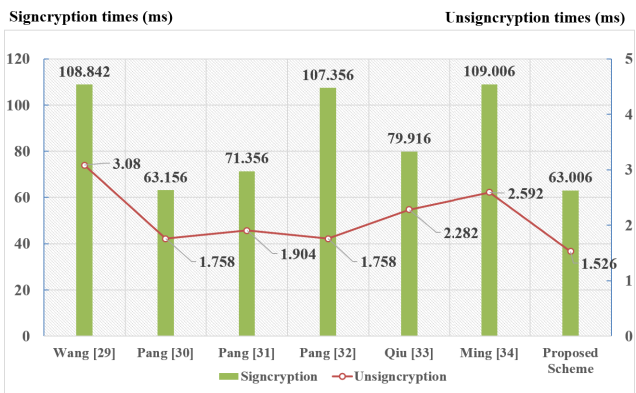

Fig. 4 Computational times of the proposed scheme and other schemes (n = 100 receivers).

Table 2 Comparison of the execution times of cryptographic operations.

Notation	Description	Runtime (ms)
T_{ECC-M}	The execution time of scalar multiplication operation in ECC	0.4417
T_{ECC-A}	The execution time of point addition operation in ECC	0.0018
T_H	The execution time of one-way hash operation	0.0081
T_{EXP}	The execution time of scalar exponential operation	5.3087

Table 3 Comparison of the data size with existing scheme.

Schemes	Data size
Wang et al. [27]	20n+188bytes
Pang et al. [28]	20n+188bytes
Pang et al. [29]	20n+80bytes
Pang et al. [30]	20n+188bytes
Qiu et al. [31]	20n+80bytes
Ming et al. [32]	20n+40bytes
Proposed Scheme	20n+40bytes

and message l_M is 128 bytes, which is the minimum block. Communication cost for n participants can be compared as shown in Table 3. The following is a communication parameter using the notation of the existing schemes and the contents expressed in standardized elements such as Z_q^* and G.

The size of the signcryption message generated in the Wang et al. [27] scheme can be calculated as follows.

$|Q| + |V| + |M| + |\beta| + |c_0| + \ldots + |c_{n-1}| = |G| + |G| + |l_M| + |Z_q^*| + n|Z_q^*| = 20n + 188bytes$.

The size of the signcryption message generated in the Pang et al. [28] scheme can be calculated as follows.

$|J| + |W| + |z| + |h| + |a_0| + \ldots + |a_{n-1}| = |C_k| + |G| + |Z_q^*| + |Z_q^*| + n|Z_q^*| = 20n + 188bytes$.

The size of the signcryption message generated in the Pang et al. [29] scheme can be calculated as follows.

$|c_0| + |c_1| + \ldots + |c_{n-1})| + |R| + |V| + |w| + |z| = n|Z_q^*| + |G| + |G| + |Z_q^*| + |Z_q^*| = 20n + 80bytes$.

The size of the signcryption message generated in the Pang et al. [30] scheme can be calculated as follows.

$|R| + |Z| + |h| + |v| + |a_0| + \ldots + |a_{n-1}| = |G| + |C_k| + |Z_q^*| + |Z_q^*| + n|Z_q^*| = 20n + 188bytes$.

The size of the signcryption message generated in the Qiu et al. [31] scheme can be calculated as follows.

$|S| + |R_2| + |v| + |h| + |A| = |G| + |G| + |Z_q^*| + n|Z_q^*| = 20n + 80bytes$.

The size of the signcryption message generated in the Ming et al. [32] scheme can be calculated as follows.

$|L_S| + |G_S| + |\sigma_S| = |G| + n|Z_q^*| + |Z_q^*| = 20n + 40bytes$.

The size of the signcryption message generated in the proposed scheme can be calculated as follows.

$|T_S| + |C_S| + |V_G| = |G| + n|Z_q^*| + |G| = 20n + 40bytes$.

This represents the increase in communication cost (bytes) as the number of participants n increases. The schemes [27], [28], [30] has a large transmission overhead because it transmits the encrypted message as it is. Since the schemes [29], [31] sends an additional verification value to verify the signcryption, it consumes 40 bytes more than the Ming et al. [32] and the proposed scheme. Therefore, it can be seen that the proposed scheme has less cost incurred in communication compared to the existing CL-MRSC schemes, and provides public verification that Ming et al. [32] does not provide.

6. Conclusions

In an increasingly expanding IoT service environment, signatures are required for secure message transmission. Digital signature protocols have been studied for a long time, and many studies have sought to make them more lightweight to suit the IoT, while also satisfying various security requirements unique to the IoT (which is a limited environment). Although many researchers aim to apply lightweight signature encryption techniques (such as CL-MRSC) to environments such as the IoT, public key replacement and message forgery attacks must be prevented while also ensuring efficiency. Thus, we developed an efficient and public verifiable CL-MRSC scheme that prevents public key replacement attacks and satisfies the various other security requirements analyzed in Background, including confidentiality, authentication, and non-repudiation of messages transmitted between two objects. Fast and efficient public verifiable signcryption is used to this end. Existing schemes are particularly susceptible to public key replacement; we solved this problem while ensuring public verifiability and computational efficiency. The scheme can be widely applied in IoT environments, including smart metering and certificateless ones (wherein a sender's authentication function is relatively weak), by ensuring that sender authenticity is public verifiable. In future, we will develop a scheme that is more computationally efficient but ensures MRSC.

References

[1] M.B. Yassein, S. Aljawarneh, and A. Al-Sadi, "Challenges and features of IoT communications in 5G networks," 2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA), Nov. 2017.

[2] S. Li, L. Da Xu, and S. Zhao, "5g internet of things: A survey," Journal of Industrial Information Integration, vol.10, pp.1–9, 2018.

[3] P.P. Ray, "A survey of iot cloud platforms," Future Computing and Informatics Journal, vol.1, no.1-2, pp.35–46, 2016.

[4] H. Arasteh, V. Hosseinnezhad, V. Loia, A. Tommasetti, O. Troisi, M. Shafie-khah, and P. Siano, "Iot-based smart cities: a survey," 2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC), pp.1–6, IEEE, 2016.

[5] L. Da Xu, W. He, and S. Li, "Internet of things in industries: A survey," IEEE Trans. Ind. Informat., vol.10, no.4, pp.2233–2243, 2014.

[6] M. Karaköse and H. Yetiş, "A cyberphysical system based mass-customization approach with integration of industry 4.0 and smart city," Wireless Communications and Mobile Computing, vol.2017, 2017.

[7] M.R. Asghar, G. Dán, D. Miorandi, and I. Chlamtac, "Smart meter data privacy: A survey," IEEE Communications Surveys & Tutorials, vol.19, no.4, pp.2820–2835, 2017.

[8] D.-H. Lee and I.-Y. Lee, "Dynamic group authentication and key exchange scheme based on threshold secret sharing for iot smart metering environments," Sensors, vol.18, no.10, p.3534, 2018.

[9] C.-H. Lin, S.-J. Chen, C.-L. Kuo, and J.-L. Chen, "Non-cooperative game model applied to an advanced metering infrastructure for non-technical loss screening in micro-distribution systems," IEEE Trans. Smart Grid, vol.5, no.5, pp.2468–2469, 2014.

[10] S.S. Al-Riyami and K.G. Paterson, "Certificateless public key cryptography," International conference on the theory and application of cryptology and information security, pp.452–473, Springer, 2003.

[11] D. He, J. Chen, and J. Hu, "A pairing-free certificateless authenticated key agreement protocol," International Journal of Communication Systems, vol.25, no.2, pp.221–230, 2012.

[12] T.K. Mandt and C.H. Tan, "Certificateless authenticated two-party key agreement protocols," Annual Asian Computing Science Conference, pp.37–44, Springer, 2006.

[13] D.H. Yum and P.J. Lee, "Generic construction of certificateless signature," Australasian Conference on Information Security and Privacy, pp.200–211, Springer, 2004.
[14] X. Huang, Y. Mu, W. Susilo, D.S. Wong, and W. Wu, "Certificateless signature revisited," Australasian Conference on Information Security and Privacy, pp.308–322, Springer, 2007.
[15] A.W. Dent, "A survey of certificateless encryption schemes and security models," International Journal of Information Security, vol.7, no.5, pp.349–377, 2008.
[16] B. Libert and J.-J. Quisquater, "On constructing certificateless cryptosystems from identity based encryption," International Workshop on Public Key Cryptography, pp.474–490, Springer, 2006.
[17] Y. Zheng, "Digital signcryption or how to achieve cost (signature & encryption) ≪ cost (signature) + cost (encryption)," Annual international cryptology conference, pp.165–179, Springer, 1997.
[18] S.S.D. Selvi, S.S. Vivek, and C.P. Rangan, "Identity based public verifiable signcryption scheme," International Conference on Provable Security, pp.244–260, Springer, 2010.
[19] M. Toorani and A.A. Beheshti, "An elliptic curve-based signcryption scheme with forward secrecy," arXiv preprint arXiv:1005.1856, 2010.
[20] S. Mandal, S. Mohanty, and B. Majhi, "Universally verifiable certificateless signcryption scheme for manet," Proc. International Conference on Microelectronics, Computing & Communication Systems, pp.77–89, Springer, 2018.
[21] L. Cui, B. Yun, S. Lin, and B. Wenhua, "A new certificateless signcryption scheme without bilinear pairing," 2018 13th International Conference on Computer Science & Education (ICCSE), pp.1–5, IEEE, 2018.
[22] J. Malone-Lee, "Identity-based signcryption," IACR Cryptol. ePrint Arch., vol.2002, p.98, 2002.
[23] M. Barbosa and P. Farshim, "Certificateless signcryption," Proc. 2008 ACM symposium on Information, computer and communications security, pp.369–372, 2008.
[24] G. Gao, X. Peng, and L. Jin, "Efficient access control scheme with certificateless signcryption for wireless body area networks," IJ Network Security, vol.21, no.3, pp.428–437, 2019.
[25] Q. Yang, Y. Zhou, and Y. Yu, "Leakage-resilient certificateless signcryption scheme," 2019 IEEE Globecom Workshops (GC Wkshps), pp.1–6, IEEE, 2019.
[26] S.S.D. Selvi, S.S. Vivek, D. Shukla, and P.R. Chandrasekaran, "Efficient and provably secure certificateless multi-receiver signcryption," International Conference on Provable Security, pp.52–67, Springer, 2008.
[27] B. Wang, J. Rong, S. Zhang, and L. Liu, "Research on data security of multicast transmission based on certificateless multi-recipient signcryption in ami," International Journal of Electrical Power & Energy Systems, vol.121, p.106123, 2020.
[28] L. Pang, M. Kou, M. Wei, and H. Li, "Efficient anonymous certificateless multi-receiver signcryption scheme without bilinear pairings," IEEE Access, vol.6, pp.78123–78135, 2018.
[29] L. Pang, M. Wei, and H. Li, "Efficient and anonymous certificateless multi-message and multi-receiver signcryption scheme based on ecc," IEEE Access, vol.7, pp.24511–24526, 2019.
[30] L. Pang, M. Kou, M. Wei, and H. Li, "Anonymous certificateless multi-receiver signcryption scheme without secure channel," IEEE Access, vol.7, pp.84091–84106, 2019.
[31] J. Qiu, K. Fan, K. Zhang, Q. Pan, H. Li, and Y. Yang, "An efficient multi-message and multi-receiver signcryption scheme for heterogeneous smart mobile iot," IEEE Access, vol.7, pp.180205–180217, 2019.
[32] Y. Ming, X. Yu, and X. Shen, "Efficient anonymous certificate-based multi-message and multi-receiver signcryption scheme for healthcare internet of things," IEEE Access, vol.8, pp.153561–153576, 2020.

Dae-Hwi Lee received the B.S., M.S. and Ph.D degrees in Depart of Computer Science Engineering from Soonchunhyang University, South Korea, in 2015, 2017 and 2021, respectively. He is now a Ph.D. candidate in Department of Software Convergence from Soonchunhyang University, South Korea.

Won-Bin Kim received the B.S. and M.S. degrees in Depart of Computer Science Engineering from Soonchunhyang University, South Korea, in 2015 and 2017, respectively. He is now a Ph.D. candidate in Department of Software Convergence from Soonchunhyang University, South Korea.

Daehee Seo received the B.S. degree in Electrical & Electronic Engineering from Dongshin University, South Korea, 2001. And the M.S. and Ph.D. degree in Computer Science from Soonchunhyang University, South Korea, in 2003 and 2006, respectively. During 2018 to 2020, he had been a Senior Researcher, College of Computing and Software Engineering at Kennesaw state university, USA. Now he is a professor in Faculty of Artificial Intelligence and Data Engineering from SangMyung University, South Korea.

Im-Yeong Lee received the B.S. degrees in Department of Electronic Engineering from Hongik University, South Korea, in 1981 and the M.S. and Ph.D. degrees in Department of Communication Engineering from Osaka University, Japan, in 1986 and 1989, respectively. During 1989 to 1994, he had been a senior researcher at Electronics and Telecommunications Research Institute, South Korea. Now he is a professor in Department of Computer Software Engineering from Soonchunhyang University, South Korea.