Peer-to-Peer Networking and Applications (2019) 12:209–226
https://doi.org/10.1007/s12083-018-0659-8

A lightweight signcryption scheme for defense against fragment duplication attack in the 6LoWPAN networks

Mohammad Nikravan[1] · Ali Movaghar[2] · Mehdi Hosseinzadeh[3,4]

Received: 3 December 2016 / Accepted: 3 May 2018 / Published online: 11 May 2018

Abstract
The Internet of Things (IoT) presents a new paradigm of the future Internet that intends to provide interactive communication between various processing objects via heterogeneous networks. The IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) is an IPv6 adaptation sub-layer and provides the requirements of IP connectivity between resource-constrained devices in lossy, low power networks. Since the size of a packet in the IPv6 is larger than the size of a frame in the IEEE 802.15.4, the 6LoWPAN adaptation layer performs packet fragmentation. In this paper, first, the 6LoWPAN fragmentation mechanism in terms of security issues is analyzed and then, fragment duplication attack which an attacker can selectively disrupt the reassembly of fragments of a particular packet at a receiver node is identified. Next, signcryption, which is a high performance cryptographic primitive, is discussed. Finally, a lightweight Offline-Online SignCryption (OOSC) scheme is proposed to counter fragment duplication attack. The evaluation shows that the proposed scheme is secure in the random oracle model and in terms of computational cost, and energy consumption efficiently counters with this attack.

Keywords Internet of things · Security · Signcryption · 6LoWPAN · Fragment duplication attack

1 Introduction

The Internet of Things (IoT) presents a new paradigm of the future Internet that intends to provide interactive communication between sensors, computing systems, and processing objects via heterogeneous networks [1]. The different things such as sensors, computers, personal digital assistant (PDAs), vehicles, and applications can communicate with each other anywhere at any time, via standard network protocols. There are a lot of meanings pertaining to the IoT, such as Machine-to-Machine communications (M2M), Wireless Sensor Networks (WSN), technologies such as Radio-Frequency Identification (RFID) and Low power Wireless Personal Area Networks (LoWPAN) [2]. The IoT has been broadly used in some areas such as automated home, healthcare, smart grids and smart cities. As the wireless communication environment is scalable, open, and heterogeneous and there are resource limitations of WSNs and RFIDs, establishing security in the IoT environment is difficult [3].

The IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) is an IPv6 adaptation sub-layer and provides IP connectivity over lossy and low power networks; the IETF standardized this sub-layer. Today, 6LoWPAN is the key technology for various models of network in the IoT such as home automation, controlling industrial systems, and smart cities [4]. The main task of 6LoWPAN adaptation layer is providing the requirements of multi-hop wireless communication between resource-constrained devices for IPv6 packets. Since the size of a packet in the IPv6 is 1280 bytes and the largest size of a frame in the IEEE 802.15.4 is 127 bytes, the 6LoWPAN adaptation layer was introduced. Packet fragmentation, which was provided in 6LowPAN adaptation layer, is the process of dividing an IP packet into smaller frames that can transmit these frames over the link layers with size limitation such as IEEE 802.15.4 [5]. Fragmentation mechanism realization in 6LoWPAN adaptation layer needs to buffering, processing, and forwarding mechanisms of fragmented packets, which is difficult in resource-constrained environments. Due to the fact that, there is no authentication mechanism at the 6LoWPAN

✉ Mohammad Nikravan
moh_nikravan@yahoo.com

[1] Faculty of Electrical and Computer Engineering, Islamic Azad University, Science and Research Branch, Tehran, Iran

[2] Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

[3] Iran University of Medical Sciences, Tehran, Iran

[4] Computer Science, University of Human Development, Sulaimaniyah, Iraq

adaptation layer, at the time of reassembling the packet by the recipient, it cannot recognize legitimate fragments from spoofed or forged fragments. Specially, sending the overlapping, forged or duplicate fragments are kinds of attacks that malicious nodes can mount. In the fragment duplication attack, an attacker can overhear the wireless channel and in harmony with the overheard fragment, generates a duplicate fragment. Therefore, it sends it towards the destination and disturbs the successful processing of fragmented packets.

Zheng in 1977 [6] initially presented the concept of signcryption. It is a high performance cryptographic primitive that is able to perform both the operations of public key encryption and digital signature at the same time, in a logically single step and at a cost considerably smaller than what needed by usual signature followed by encryption. It provides authentication, confidentiality, integrity, and non-repudiation. Because of its lower cost than signature followed by encryption, it is very appropriate for resource-constrained environments such as wireless sensor networks and Internet of things. In the public key cryptosystems, two basic approaches to authenticate the public keys are identity-based cryptography (IBC) [7] and public key infrastructure (PKI). In the PKI environment, having a trusted and un-forgeable link between the identity of a user and its public key is necessary, therefore to provide the link a signature certificate authority (CA) is required. Managing the certificates, including storage, distribution and revocation is the drawback of the PKI infrastructure. Additionally, the validity of certificates should be verified before using them. The PKI mechanism is very appropriate for the Internet. As shown in Fig. 1, in the IBC some user's unique personal information that is relating to its identity such as IP or email address are used to obtain user's public key. Here, a private key generator, which is a trusted third party entity and is called PKG, generates the private keys for the users. Using the user's identifier and system master secret key, the PKG generates private keys. The main advantage of the IBC in comparison with the PKI is that there is no need to public key

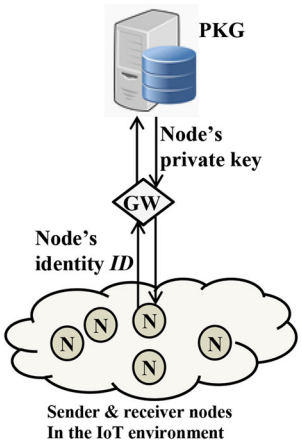

Fig. 1 An identity-based cryptosystem in the IoT

authenticity verification by a certificate and therefore IBC is the best option for the IoT. The IBC scheme has an inherent drawback called key escrow problem: since the PKG can inevitably generate private keys for all the users if the adversary compromises it, then all private keys will be obtainable. The signcryption scheme has two sub-schemes: (i) PKI-based signcryption in which a certificate authority's (CA) signature is required to provide un-forgeable and trusted link between the identity of a user and its public key. Since the cost of managing certificates including storage, distribution, and revocation is high, the PKI-based sub-scheme is very appropriate for the Internet and is not suitable for resource-constrained devices. (ii) identity-based signcryption (IBSC) which helps overcome the certificate management cost and makes it appropriate for resource-constrained networks such as WSNs and the IoT by eliminating the certificate.

The researchers in this paper aim at analyzing 6LoWPAN fragmentation process in terms of security in resource-constrained environments. To mitigate fragment duplication attack in the IoT environment, a signcryption scheme between sender and receiver nodes is proposed that provides confidentiality, integrity, authentication, and nonrepudiation by offering efficient per-fragment signcryption. It is necessary that the cost of computation of the proposed scheme be low. The proposed scheme, in this paper, is an online/offline signcryption (OOSC) scheme. A precise definition of OOSC scheme will be explained in the Sect. 4. It is depicted that the proposed scheme is existential un-forgeable against adaptive chosen messages attacks (EUF-CMA) under the q-strong Diffie-Hellman problem (q-SDHP) and indistinguishable against adaptive chosen ciphertext attacks (IND-CCA) under the bilinear Diffie-Hellman inversion problem (BDHIP) in the random oracle model.

The rest of this paper is structured as follows. Sect. 2 outlines related works. In Sect. 3 a brief overview of the security assumptions and bilinear pairings are given. In Sect. 4, the generic structure of OOSC is shown. In Sect. 5, the 6LoWPAN fragmentation mechanism is discussed. Sect. 6 gives a description of the network model, the attacker types, and the security of the 6LoWPAN fragmentation mechanism is explained. In line with the previous discussions, the signcryption scheme is proposed in Sect. 7. Sect. 8, gives the security and performance analysis of proposed scheme. Sect. 9 concludes the paper.

2 Related work

Recently, packet fragmentation is discovered as a risk factor of security in communications based on IP protocol such as 6LoWPAN-enabled networks. Kim H, et al. [8] concentrated on vulnerabilities at the 6LoWPAN adaptation layer. Attackers, using fragmentation and reassemble processes of 6LoWPAN adaptation layer, can do the denial of service (DoS) attacks. Additionally, the attackers may alter the fragment header fields such as datagram_offset, datagram_size, and datagram_tag. This

may cause resource exhaustion or buffer overflew in the receiver node [9].

With respect to IP communication, different types of fragmentation-based attacks were introduced in [10, 11]. However, these attacks generally concentrate on the weaknesses of implementation of the IP protocol. In contrast, this paper focuses on issues relating to designing and implementing the 6LoWPAN fragmentation mechanism in resource-constrained environments.

With respect to 6LoWPAN, authors in [8] suggested attaching a timestamp to unidirectional fragments of a packet and Nonce to bidirectional fragments of a packet at the 6LoWPAN layer. They claimed that the proposed mechanism could mitigate fragment replay attack. Note that in this work neither a clear identification of this attack nor an experimental result is given. Furthermore, because the timestamp and Nonce are not encrypted and the attacker may spoof them, this approach cannot mitigate fragment duplication attack. Authors in [12] proposed a scheme, named content-chaining scheme to mitigate fragment duplication attack. In fragment duplication attack, the attacker duplicates an eavesdropped fragment and disrupts fragmented packet reassembling. In the proposed scheme, a cryptographic hashed digest of a fragment is attached to the end of previous fragment. At the receiver node, these hash values are recomputed and verified. Thus, the receiver node can detect that the fragment is a legitimate or a spoofed one. Since no encryption is performed, the proposed scheme does not provide confidentiality. One more thing, it has some problems with out-of-order fragments processing.

Some signcryption schemes based on bilinear pairings were proposed in [13–15]. As mentioned in the introduction, the signcryption scheme has two sub-schemes: the PKI-based and IBSC. Some well-known PKI-based sub-schemes were proposed in [6] and some IBSC sub-schemes were proposed in [14–16]. In 2002, An et al. combined the signcryption with concepts of offline/online signature and proposed a new signcryption scheme; called offline/online signcryption (OOSC) [17] (see Sect. 4). Some OOSC schemes based on PKI were proposed in [18, 19]. An Identity Based OOSC (IBOOSC) was introduced in [20]. The drawback of the proposed scheme is that it needs receiver's identity to compute offline signature in the offline phase of signcryption. After that, a new IBOOSC scheme was suggested [21] by Liu et al. that was able to compute offline signature without the knowledge of receiver's identity in the offline part and therefore solved the problem. The great benefit in the new IBOOSC scheme proposed by Li et al. [22] was that they tried to reduce the ciphertext size and offline storage. A heterogeneous OOSC was offered by Li and Xiong [23]. In the heterogeneous OOSC, the receiver node was placed in the PKI environment, whereas, the sender node was placed in the IBC environment and allows a sensor node in an IBC environment to send a message to a server in a PKI environment. They claimed that their scheme could provide secure communication in the IoT. Again, Senthil et al. [24] applied a heterogeneous OOSC to provide a secure routing in wireless sensor networks. In 2016, Li et al. [25] proposed a heterogeneous ring signcryption scheme for the Internet of Things that simultaneously can provide the confidentiality, authentication, integrity, nonrepudiation, and anonymity in a logical single step. The anonymity protects the privacy of the sender. The heterogeneous characteristic makes this scheme suitable for data transmission in the IoT, since there is no need to certificate management. However, to mitigate fragment duplication attack both the sender and receiver were placed in the IBC environment; therefore, a simpler scheme is required.

3 Preliminaries

In this section an overview on fundamentals and properties of the bilinear pairings is given.

Let us consider groups G_1 and G_2 of the same prime order p, where G_1 is an additive group generated by P and G_2 is a multiplicative group, both groups are cyclic, for more details on theory of groups see [47]. A bilinear pairing is a map $\hat{e} = G_1 \times G_1 \rightarrow G_2$ with the below properties:

1) Bilinearity: $\forall R, Q \in G_1, \forall a, b \in Z_p^* \rightarrow \hat{e}(aR, bQ) = \hat{e}(R, Q)^{ab}$.
2) Non-degeneracy: $\exists R, Q \in G_1$ *such that* $\hat{e}(R, Q) \neq 1$.
3) Computability: $\forall R, Q \in G_1$, there exists an algorithm to compute $\hat{e}(R, Q)$ efficiently.

Modified versions of Tate pairing and Weil pairing can be used to construct above-mentioned maps [14–16]. The assumption of computations for the proposed scheme's security bases on the hardness of the following problems. Therefore, an adversary that wants to attack such systems will need to solve the q-BDHIP and q-SDHP problems.

Definition 1 Assume two groups G_1 (generated by P) and G_2 of the same prime order p and a bilinear map $\hat{e} = G_1 \times G_1 \rightarrow G_2$, q-bilinear Diffie–Hellman inversion problem (q- BDHIP) in $(G_1, G_2, \hat{e})$ is computing $\hat{e}(P, P)^{1/\alpha}$ given $(P, \alpha P, \alpha^2 P, \alpha^3 P, \ldots, \alpha^q P)$, that $\alpha \in Z_p^*$. If q = 1 it is called bilinear Diffie–Hellman inversion problem (BDHIP).

Definition 2 Assume two groups G_1 (generated by P) and G_2 of the same prime order p and a bilinear map $\hat{e} = G_1 \times G_1 \rightarrow G_2$, the q-strong Diffie–Hellman problem (q-SDHP) in $(G_1, G_2, \hat{e})$, given $(P, \alpha P, \alpha^2 P, \alpha^3 P, \ldots, \alpha^q P)$ as input, is finding a pair $(\theta, \frac{1}{\theta+\alpha} P)$ where $\alpha, \theta \in Z_p^*$ and $\frac{1}{\theta+\alpha} P \in G_1$.

Solving the BDHIP is as hard as calculating discrete logarithm in either groups G_1 or G_2. If the value of α is found by calculating the discrete logarithm of αP in group G_1, then $\frac{1}{\alpha}$ and after that $\hat{e}(P, P)^{1/\alpha}$ can be calculated as the answer of BDHIP.

Therefore, to be sufficiently secure, all calculations should be done in a group that all subgroups are as big as possible, and using a subgroup of prime order is an easy way to accomplish this. To make the hardness of calculating discrete logarithm in both groups $G_1$and G_2 equal to the hardness of attacking an 80-bit symmetric key encryption, and to obtain standard levels of security against well-known attacks, it is required that G_1 be of prime order (p) of at least 160 bits and having at least $|G_2| = 1024$ bits [48]. All these issues are considered in the proposed scheme, such that $|G_1| = 542$ bits, $|G_2| = 1084$ bits, $|p| = 252$ bits. In addition, If G_1 is an elliptic curve group with an order of p bits, calculating a discrete logarithm in G_1 (and similarly BDHIP) requires to $\boldsymbol{O}\left(\sqrt{\boldsymbol{p}}\right)$ time. This means that the problem is exponential in $|p|/2$, which, indeed, means the problem is hard.

In regard to q-SDHP, as will proved in *Theorem 2* (section 8.2.), if the probabilistic polynomial time adversary F is able to forge a signature within time t with probability

$$\epsilon \geq 10(q_s+1)\left(q_s+q_{H_2}\right)/2^k,$$

then the algorithm B can solve the $q-SDHP$ in the expected time

$$t' \leq 120686\, q_{H_1}\, q_{H_2}\, \frac{t+O(q_s\tau_p)}{\epsilon\left(1-1/2^k\right)\left(1-q/2^k\right)} + O\left(q^2\tau_{mult}\right)$$

Considering the following issues, the hardness of solving the q-SDHP is shown as:

1- The k is security parameter of scheme that is equal to the number of bits of p. In the proposed scheme, it is assumed that $k = |p| = 252$. Therefore, the 2^k has a very large value. Since the 2^k is in the denominator of first equation, the probability of successful forging a signature by F (the probability of B's success) is negligible. If F is unsuccessful, then B is unsuccessful too.

2- By substitution the ϵ from first equation in the second equation, through the following prove, solving the q-SDHP by algorithm B requires to $\boldsymbol{O}(2^{\boldsymbol{k}})$ time. Therefore, the algorithm B needs to an exponential time to solve q-SDHP, which, indeed, means the q-SDHP is hard.

$$t' \leq 120686\, q_{H_1}\, q_{H_2}\, \frac{t+O(q_s\tau_p)}{\frac{10(q_s+1)\left(q_s+q_{H_2}\right)}{2^k}\cdot\frac{\left(2^k-1\right)}{2^k}\cdot\frac{\left(2^k-q\right)}{2^k}} + O\left(q^2\tau_{mult}\right)$$

$$t' \leq 120686\, q_{H_1}\, q_{H_2}\, \frac{2^{3k}\left(t+O(q_s\tau_p)\right)}{10(q_s+1)\left(q_s+q_{H_2}\right).\left(2^k-1\right).\left(2^k-q\right)} + O\left(q^2\tau_{mult}\right)$$

$$t' \leq 12068\, q_{H_1}\, q_{H_2}\, \frac{2^{3k}\left(t+O(q_s\tau_p)\right)}{\left(q_s^2+q_s+q_{H_2}q_s+q_{H_2}\right).\left(2^{2k}-2^k-q2^k+q\right)} + O\left(q^2\tau_{mult}\right)$$

Because the numerator is dominated by 2^{3k}, and the denominator is dominated by 2^{2k}, therefore it is found that:

$$\lim_{K\to\infty}\left(12068\, q_{H_1}\, q_{H_2}\, \frac{2^{3k}\left(t+O(q_s\tau_p)\right)}{\left(q_s^2+q_s+q_{H_2}q_s+q_{H_2}\right).\left(2^{2k}-2^k-q2^k+q\right)} + O\left(q^2\tau_{mult}\right)\right) = \lim_{K\to\infty}\left(\frac{2^{3k}}{2^{2k}}\right) = 2^k$$

4 Offline-online Signcryption

In an OOSC scheme, the process of signcryption is divided into two phases: offline phase and online phase. In the offline phase, the heavier computational operations are done when the message is not available. In the online phase, only light-weight computations are done when the message is available. Since the signcryption runs in the resource-constrained environment, it is important to minimize the computational cost and energy consumption of the applied approaches. Therefore, the OOSC is very appropriate to provide a secure communication for resource-constrained environments such as sensor nodes, IoT and WSNs. The resource-limited environments include devices that have low bandwidth, low battery power, low computational power and low storage capacity. Here, the formal definition and security notions for the OOSC schemes are discussed.

4.1 Syntax

A common structure of a generic OOSC scheme includes five algorithms as detailed below. Fig. 2 shows the sequence of steps and exchanged messages between parties in an OOSC scheme.

Setup: is a probabilistic algorithm. The private key generator (PKG) runs this algorithm to output the public parameters *params*, master public key P_{pub}, and master secret key *msk* using a security parameter k as input. The *params* are published but the *msk* is kept secret.

Extract: is a probabilistic key generation algorithm. The PKG runs this algorithm to output the private key S_{ID}associated to the user's identity *ID* using the system parameters *params*, master secret key *msk* and user's identity *ID* as input. Then the PKG transmits the private key to user over a secure channel.

OffSignCrypt: is a probabilistic algorithm. The sender node runs this algorithm to output an offline signature π using the system public parameters *params*, receiver's public key Q_r and sender's private key S_{ID_s} as input.

***OnSignCrypt*:** is a probabilistic algorithm. The sender node runs this algorithm to output full signcryption ciphertext σ using the system public parameters *params*, message *m,* and an offline signcryption π as input.

***UnSignCrypt*:** is a deterministic algorithm. The receiver node runs this algorithm to output message m or $\perp$ for "reject" if σ is an invalid ciphertext between the sender and receiver. The inputs of algorithm are the system parameters *params*, ciphertext σ, sender's identity ID_s and receiver's private key S_{ID_r}.

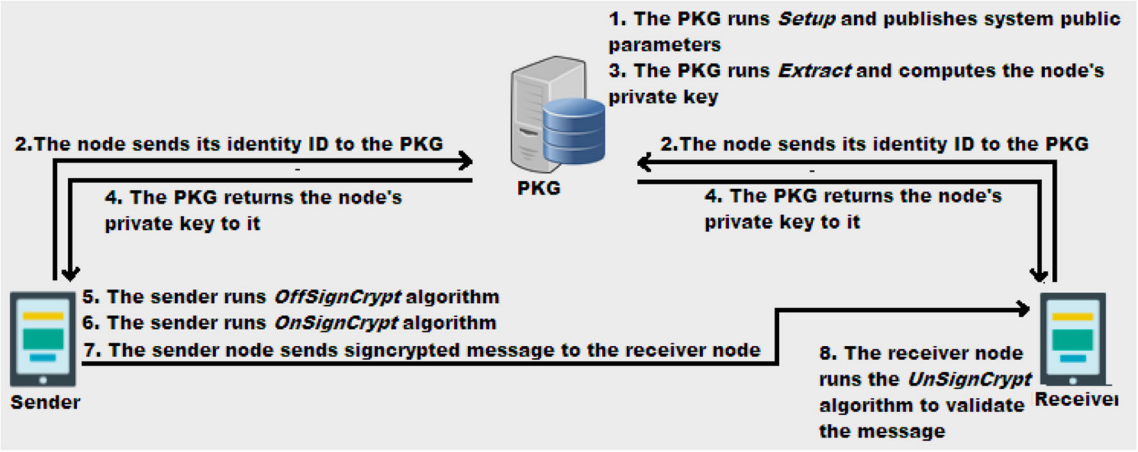

Fig. 2 Sequence of steps and exchanged messages between parties in an OOSC scheme

The proposed algorithms must obey the standard constraints of the OOSC consistency, i.e.

If $\pi = OffSignCrypt(S_{ID_s}, Q_r)$, and $\sigma = OnSignCrypt(\pi, m)$, then $m = UnSignCrypt(\sigma, ID_s, S_{ID_r})$.

Note that system public parameters *params* is omitted in *OffSignCrypt*, *OnSignCrypt,* and *UnSignCrypt* algorithms.

4.2 Security notions

The standard notions of security for signcryption schemes are message confidentiality (adaptive-chosen ciphertext security or IND-CCA) and message unforgeability (existentially signature-un-forgeable against adaptive chosen messages attacks or EUF-CMA). The notion in [13, 15] are somewhat adjusted to apply for the present scheme.

To provide IND-CCA security, it is assumed that the challenger *B* and an adversary *F* play the following game. Fig. 3 shows the phases of the game.

Setup phase: the challenger *B* runs *setup* algorithm. She takes the security parameter *k* as input and outputs the master secret key *msk* and public parameters *params. B* also computes the public and private keys of receiver $\left(Q_r^*, S_{ID_r^*}\right)$. Finally, she sends all of them, except $S_{ID_r^*}$, to *F.*

Phase 1: *F* can adaptively ask a polynomially bounded number of unsigncryption queries. Each time *F* runs an unsigncryption query, it sends an identity of sender ID_s and a cipertext σ to the *B. B* runs the unsigncryption algorithm $Unsigncrypt(\sigma, ID_s, S_{ID_r})$ and returns the result to *F.* Phase 1 is repeated until *F* decides to stop it.

Challenge phase: *F* produces two plaintexts m_0*and* m_1 with equal length and an identity of a sender ID_s^* and sends them to *B*. First, *B* runs *Extract* algorithm and generates private key of the sender $S_{ID_s^*}$, then *B* chooses $\tau \in \{0, 1\}$ randomly, computes $\pi^* = OffSignCrypt\left(S_{ID_s^*}, Q_r^*\right)$, and $\sigma* = OnSignCrypt(\pi^*, m_\tau)$. Finally, *B* returns σ^* to *F.*

Phase 2: similar to phase 1, *F* can adaptively ask a polynomially bounded number of unsigncryption queries. In this phase, he is not permitted to ask an unsigncryption query on $\left(\sigma^*, ID_s^*\right)$ therefore it is not able to obtain the relating plaintext. Phase 2 is repeated until *F* decides to stop it.

Guess phase: *F* generates a random bit τ' and if $\tau' = \tau$ then he conquers the game.

Definition 3 An *IBSC* scheme is IND-CCA secure if no probabilistic polynomial time adversary *F* has advantage at least ϵ in the IND-CCA game.

To provide EUF-CMA security it is assumed that the challenger *B* and an adversary *F* play the following game. Fig. 4 shows the phases of the game.

Setup phase: the challenger *B* runs *setup* algorithm. She takes the security parameter *k* as input and outputs the public parameters *params. B* also computes the public and private keys of receiver $\left(Q_r^*, S_{ID_r^*}\right)$. Finally, she sends all of them to *F.*

Attack phase: *F* adaptively asks a polynomially bounded number of signcryption and key generation queries. Each time *F* runs a key generation query, it sends an identity *ID* of a node to the *B. B* runs the *Extract* algorithm and returns the node's private key S_{ID} to *F.* Each time *F* runs a signcryption query, it sends a message *m* and an identity of a sender node ID_s to the *B.* First, *B* runs the *Extract* algorithm to produce private key of the sender node S_{IDs}, then *B* runs $\pi = OffSignCrypt\left(S_{IDs}, Q_r^*\right)$ and $\sigma = OnSignCrypt(\pi, m)$ algorithms and returns σ to *F.*

Forgery phase: *F* generates a ciphertext σ^* and an identity of a sender node ID_s^* and conquers the game if these conditions are met: $m^* = Unsigncrypt\left(\sigma^*, ID^*{}_s, S_{ID^*{}_r}\right)$, *F* has

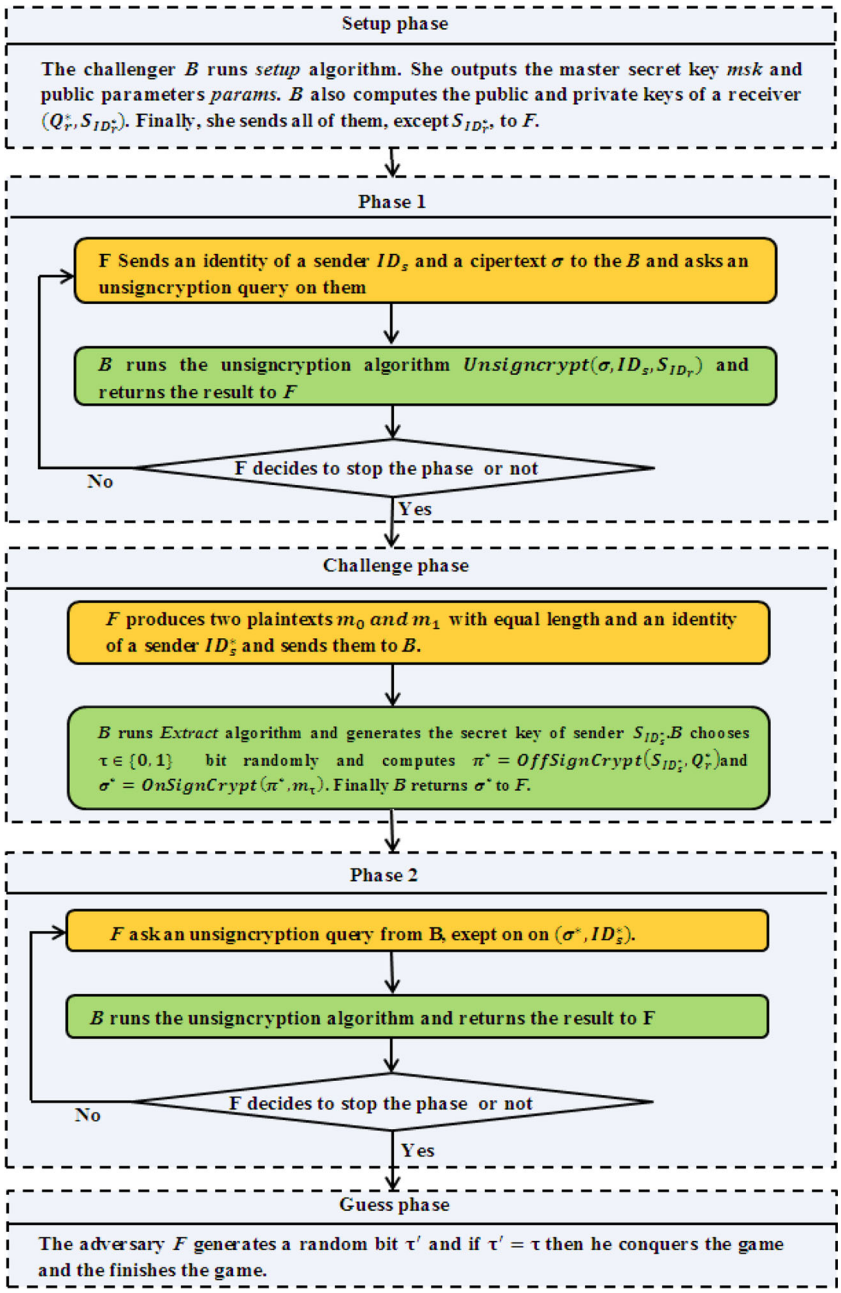

Fig. 3 Game phases played between adversary F and challenger B

not asked a key generation query on ID^*_s, F has not asked a signcryption query on (m^*, ID^*_s).

Definition 4 An *IBSC* scheme is EUF-CMA secure if no probabilistic polynomial time adversary F has advantage at least ϵ in the EUF-CMA game.

5 6LoWPAN fragmentation

In this section, the fragmentation mechanism of the 6LoWPAN will be briefly discussed. In addition, the mechanism of packet routing in the 6LoWPAN and its effect on the fragment forwarding will be detailed.

5.1 Fragmentation mechanism

Position of the 6LoWPAN adaptation layer in the IoT communication protocol stack is between the network layer and the link layer. The 6LoWPAN adaptation layer aims to optimize the usage of payload space of IEEE 802.15.4 link layer through packet header compression and packet fragmentation. The common size of an IPv6 packet is 1280 bytes, whereas the maximum size of the IEEE 802.15.4 frame is 127 bytes. Thus,

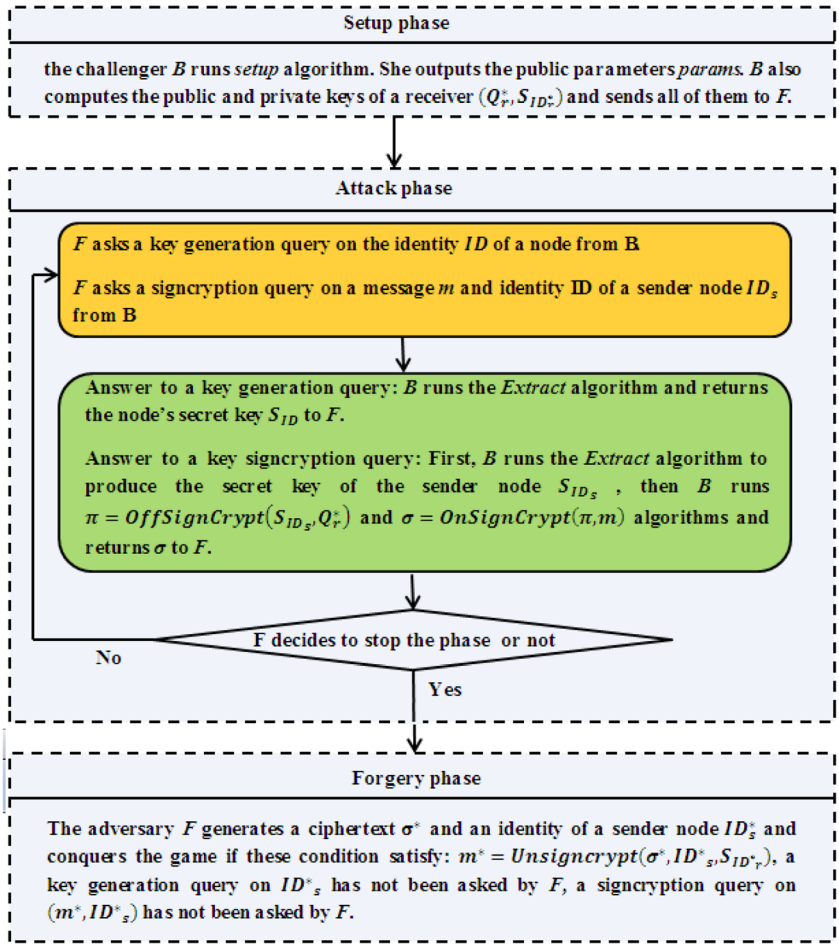

Fig. 4 Game phases played between adversary F and challenger B

usually an IPv6 packet needs to be broken into fragments to fit within a single IEEE 802.15.4 frame.

Each fragment has a fragment header (i.e., the size of this header is fixed). The content of IPv6 packet is laid in the remaining space of fragment payload. As shown in Fig. 5, each 6LoWPAN fragment header includes *datagram_tag, datagram_size, and datagram_offset* fields, which are used for in-place reassembly. The *datagram_tag* field is unique for each fragmented packet per sender and permits the recipient to relate the rest of the fragments to the first fragment. Additionally, after a receiver node has received the initial fragment of the packet, the recipient uses the *datagram_tag* to look up routing table for every subsequent fragments of the packet.

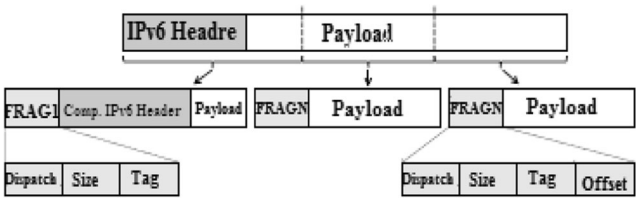

Fig. 5 Packet structure of the initial fragment FRAG_1 and subsequent fragments FRAG_N in the 6LoWPAN

In the 6LoWPAN fragments, only the initial fragment of a packet includes the IP header information. Therefore, just the initial fragment caries end-to-end routing information, whereas in the regular IP fragments each fragment includes this information.

The recipient reserves a reassembly buffer space according to the *datagram_size* field of the fragment. *Datagram_size* indicates the whole size of the IPv6 packet before fragmentation and compression. Each received fragment must be placed at correct location within the original IPv6 packet during the packet reassembling process that is done by *datagram_offset* field.

5.2 Fragment forwarding mechanism

According to which layer takes the routing decision, there exist two kinds of routing schemes in the 6LoWPAN networks: the route-over scheme and the mesh-under scheme. The difference between them is that in the former the network layer takes routing decision whereas in the latter the adaptation layer takes routing decision.

In the mesh-under scheme, the 6LoWPAN network layer does not participate in IP routing. The adaptation layer is in charge of routing and forwards packets towards the

destination over multi-hop wireless path. The 6LoWPAN adaptation layer breaks an IPv6 packet into fragments. These fragments are forwarded towards the next node using mesh-under routing, and finally reach the final recipient. Different fragments of an IPv6 packet can be forwarded over different paths and they are collected at the destination. Thus, mesh-under routing is not transparent to packet fragmentation [9, 26].

In the route-over scheme, each node operates as an IPv6 router and participates in routing process. Additionally, the network layer takes the routing decisions, and forwarding the packets between the nodes is supported by the IP. The network layer uses the extra encapsulated IP header to take correct decision to perform routing and forwarding. The 6LoWPAN adaptation layer builds up an immediate mapping between the IP headers and the frame. When the 6LoWPAN adaptation layer breaks an IPv6 packet into fragments, according to the routing table contents the fragments sent to the next node. The adaptation layer of the recipient (next node) is in charge of checking received fragments and reassembling them to generate the original IP packet and sending the IP packet to the network layer. If the recipient is the final destination of the packet, the network layer forwards this packet to the transport layer according to the routing table content otherwise, it forwards this packet towards the next node.

6 Security discussion

The security discussion in this paper, concentrates on the ways that an attacker can disturb the correct processing of legitimate fragments of a packet. The attacker maliciously uses the 6LoWPAN routing and fragmentation mechanisms to perform attacks. Particularly, a network-internal attacker will be focused on.

In this paper, the attacker's hardware resources are abstracted. Therefore, for every resource-constrained device that is comparable or equivalent to real devices in the network, the introduced attack is performable. Furthermore, due to inherent characteristic of the 6LoWPAN layer, detecting and mitigating these attacks are hard. Finally, the jamming attack is an external well-known attack that must be mitigated properly [27]. Here the network model, attacker model, fragment duplication attack, and routing scheme vulnerability are described.

6.1 Network model

Generally, in the following discussion there is no emphasis on using particular network topology, device type or link layer technology. The network model belongs to the proposed scheme is shown in Fig. 6. This model includes four parties: resource-constrained nodes, gateway, Internet, Private Key Generator (PKG). The gateway connects the local network

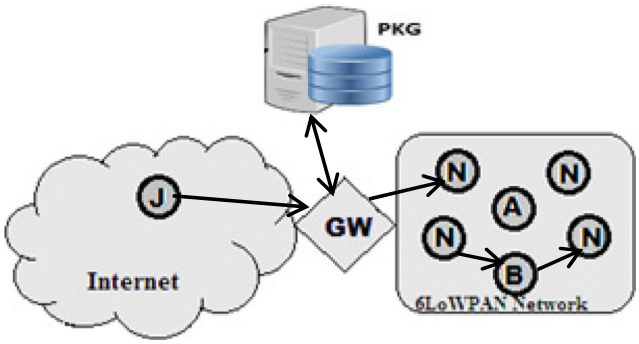

Fig. 6 Network model: resource-constrained nodes (N), Gateway (GW), Private Key Generator (PKG), the attackers Alice (**a**), Bob (**b**) and Jack (J), forwarding paths (arrows)

including resource-constrained nodes (6LoWPAN network) to a backbone infrastructure such as Internet and the PKG. The gateway sends messages form nodes to the PKG and vice versa. It also sends the identity ID of nodes to the PKG and returns their private keys from PKG to the nodes. Therefore, the PKG reaches nodes through the gateway. The PKG is in charge of generating and distributing the private keys of nodes using their identity ID. In addition, it generates the system public parameters and sends them to the gateway. The links between the gateway and PKG, the gateway and nodes are wireless link.

The communication link between the PKG and gateway is secure and the gateway uses a secure channel to send private keys to the IoT nodes. In fact, the proposed scheme relies on secure and authenticated channels established by DTLS [49] to distribute private keys. The private keys are exchanged over secure DTLS channels between the PKG, the gateway, and the IoT nodes. Therefore, the private keys are protected against eavesdropping.

However, this research's analysis concentrates on resource-constrained devices. Therefore, it is assumed that the networks includes devices with only a few of kilobytes of RAM and have low computational power. To obtain experimental result, it is supposed that 6LoWPAN nodes are MICA2 that are armed with an ATmega128 8-bit processor clocked at 7.3728 MHz, 4 KB RAM and 128 KB ROM, and are widely used in WSNs and the IoT. The communication links between these nodes are low-power wireless links. One more thing, the security mechanisms of link layer in the network may be used. Note that, the 6LoWPAN network nodes have a very limited ability to store, reassemble, and process fragmented packets.

6.2 Attacker model

Three types of attackers can mount fragment duplication attack against the 6LoWPAN fragmentation Alice, Bob, and Jack. Alice and Bob both are internal attackers, which are two nodes of the 6LoWPAN network. They are placed in different locations of the network, according to the forwarding path of a particular fragmented packet, whereas Jack is an external attacker [28].

Therefore, as shown in Fig. 6, Alice and Bob can send messages to all of the nodes in the network. Alice is an out-path attacker, that she is located outside but near the forwarding path of fragmented packets. As a result, she eavesdrops the communication channel, and reacts the eavesdropped messages by sending packets. Bob is an on-path attacker and is placed on the forwarding path. Thus, he can do everything Alice can do; additionally he can drop, reorder, alter, or delay legitimate fragments.

In contrast to Alice and Bob, Jack is an external attacker and is placed outside the 6LoWPAN network. Generally, the external attackers hold more powerful resources than the internal attackers (6LoWPAN nodes) do. They leverage this fact to flood a resource-constrained node by sending numerous large packets [29]. Since these packets are fragmented at the gateway, the receiver node must process more packets and it causes to amplify this attack. Using authenticated tunnels at the gateway may help to withstand against such flooding attacks from external attackers. If an external host behaves maliciously, the gateway excludes it from communication by using Authenticated tunnels. To highlight, counter with external attackers is out of scope of this article, therefore internal attackers will be the main focus in the due discussion.

6.3 Fragment duplication attack

Due to lake of authentication in the 6LoWPAN adaption layer, the receiver node cannot verify whether the source of a received fragment and the source of previously received fragments of the same IPv6 packet are the same or not. Therefore, the receiver node cannot distinguish spoofed duplicate fragments from legitimate ones at the time of reception. Instead, all fragments that have a same MAC address and the 6LoWPAN *datagram_tag* field need to be stored and processed by the receiver node. Alice can exploit this fact and selectively disrupt the reassembly of fragments of a particular packet at the receiver node. For instance, Alice eavesdrops the wireless communication channel and then injects forged fragments with dummy payload. However, the header of the spoofed fragment must link this fragment with legitimate ones, as shown in Fig. 7.

Since the receiver node cannot distinguish legitimate and spoofed fragments at the 6LoWPAN adaptation layer, it cannot determine to use which ones to reassemble the packet. Additionally, in order to provide reliable packet delivery, the packet may be retransmitted by higher layer protocols. In consequence, packet retransmission causes energy and resource exhausting of the receiver and forwarding nodes. Due to the complexity of withstanding against the fragment duplication attack, the 6LoWPAN standards suggest to drop the corrupted IPv6 packets. The information belonging to the reassembled packet can be used in the upper layers to distinguish the right composition of fragments for packets with duplicate fragments. However, these approaches offer considerable defects. Most notably, it needs the receiver node to store all received fragments. Furthermore, after receiving each fragment, the receiver node must reassemble the fragments at least once. Therefore, the receiver node cannot recognize the forged duplicate fragments early at the receiving time and it may overload the limited buffer at the receiver node [12].

6.4 Vulnerability of the routing schemes

According to whether the receiver node is the final destination of fragments or it reassembles them for forwarding purpose, the effect of introduced fragment duplication attack on the 6LoWPAN network is different. Therefore, depending on the used fragment forwarding mechanism (mesh-under or route-over) the target node, which is affected by fragment duplication attack, is different.

Since in the route-over packet forwarding mechanism the network layer takes routing decisions, each node reassembles fragmented packets. Therefore, Alice can disturb the fragment processing of her one-hop neighborhood. However, Alice cannot disturb the reassembling process at the nodes that are not topologically her direct neighbors.

For the mesh-under forwarding mechanism, the intermediate nodes only forward the attack fragments towards the destination and they do not reassemble fragmented packets. Therefore, Alice can mount the introduced attack against any 6LoWPAN destinations instead of only her direct neighbors.

7 The proposed scheme

In this section, the OOSC scheme to mitigate fragment duplication attack will be proposed. This scheme is based on Barreto et al.'s signcryption scheme [15] and it permits a

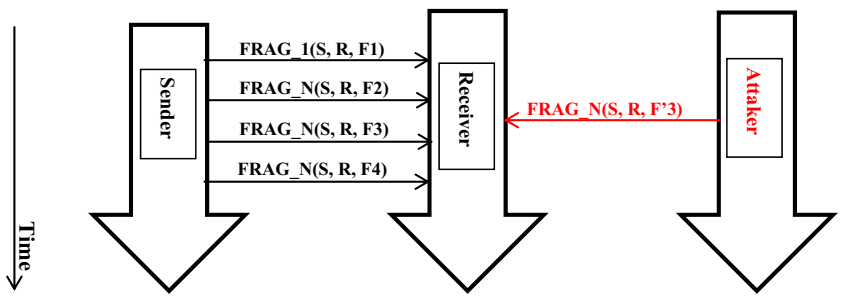

Fig. 7 Fragment duplication attack: A receiver must decide which fragment payload ($F_3 or\ F_3^{'}$) to use during reassembly

sender node to send a signcrypted fragment to a receiver node. It comprises of five algorithms.

Setup: takes a security parameter k as input, the PKG chooses groups G_1 and G_2 of the same prime order p, where G_1 is an additive group generated by P and G_2 is a multiplicative group, a bilinear map $\hat{e} = G_1 \times G_1 \rightarrow G_2$ and hash functions $H_1 : \{0,1\}^* \rightarrow Z_p^*$, $H_2 : \{0,1\}^n \times G_2 \times G_1 \times G_1 \rightarrow Z_p^*$, $H_3 : G_2 \rightarrow \{0,1\}^n$. The fragment size, which must be signcrypted, is n bits. Then the PKG randomly selects a master secret key $msk \in Z_p^*$, sets the master public key $P_{pub} = mskP$, and computes $g = \hat{e}(P,P)$. In addition, a symmetric encryption algorithm *(Enc, Dec)* such as AES [30] is required to encrypt and decrypt the fragment payload. This algorithm must satisfy the confidentiality and unforgeability, with n bits key space. The PKG also, selects this algorithm. Finally, the PKG publishes system parameters *params* and keeps the master secret key *msk* secret.

$$params = \{G_1, G_2, n, \hat{e}, P, P_{pub}, g, H_1, H_2, H_3, Enc, Dec\}$$

Extract: A node sends its identity ID to the PKG, The PKG generates the node's private key $S_{ID} = \frac{1}{H_1(ID)+msk} P$ and sends the private key to the node over a secure channel. Here, the public key of node is $Q_{ID} = H_1(ID)P + P_{pub}$. The identity ID of sender and receiver will be indicated by $ID_s$and ID_r respectively.

OffSignCrypt: takes as input the sender's private key S_{IDs} and receiver's public key Q_r, the steps of algorithm are as follows:

1. Choose x, λ from Z_p^* randomly.
2. Compute $r = g^x$.
3. Compute $T = xQ_r = x(H_1(ID_r)P + P_{pub})$.
4. Compute $S = \lambda^{-1}(S_{IDs} + P)$.

The offline signcryption is $\pi = (x, r, \lambda, S, T)$.

OnSignCrypt: takes as input the message m and an offline signcryption π, the steps of algorithm are as follows:

1. Compute $C = Enc_{H_3(r)}(m)$.
2. Compute $Tmp = SHA\_1(m)$.
3. Compute $h = H_2(Tmp, r, S, T)$.
4. Compute $\varphi = \lambda(x + h) \bmod p$.

The full ciphertext is $\sigma = (C, \varphi, S, T)$.

UnSignCrypt: takes as input the ciphertext σ, sender's identity ID_s and receiver's private key S_{IDr}, the steps of algorithm are as follows:

1. Compute $r = \hat{e}(T, S_{IDr})$.
2. Recover $m = Dec_{H_3(r)}(C)$.
3. Compute $Tmp = SHA\_1(m)$.
4. Compute $h = H_2(Tmp, r, S, T)$.
5. Compute $L = \varphi^* S$.
6. If $r = \hat{e}\left(L, \frac{PQ_{ID_s}}{P+H_1(ID_s)P+P_{pub}}\right) g^{-h}$ then, accept and return the message m, else return $\perp$.

When an IPv6 packet is broken into several fragments, each fragment size in 6LoWPAN adaptation layer is 81 bytes. Due to 8 bytes header in the first fragment of a packet and 6 bytes header in the subsequent fragments, the payload size to be signcrypted in the first and subsequent fragments are 73 and 75 bytes respectively (For more detail see Sect. 5.1.).

Therefore, in order to send a fragmented packet and mitigate fragment duplication attack, at the sender's side the sender node first runs the *OffSignCrypt* algorithm to build an offline signcryption π according to the receiver's public key Q_r. Note that the receiver's public key is computable by $Q_r = x(H_1(ID_r)P + P_{pub})$. Then, in the *OnSignCrypt* algorithm, first each fragment payload is encrypted using a symmetric encryption algorithm such as AES with $H_3(r)$ as the session encryption key, and the ciphertext C is obtained. Second, the fragment payload (about 600 bits) is given to the secure hash function SHA-1 [31] to obtain a fixed 160 bits digest of the fragment payload. Finally, the sender node builds the full ciphertext σ of the fragment payload as described above and sends it to the receiver. Note that, the sender node runs the *OffSignCrypt* algorithm once per packet to obtain offline signcryption π, but it runs the *OnSignCrypt* algorithm to signcrypt the fragment before sending it. It implies that each fragment is signcrypted individually and different packets are sent with different signcryptions.

Upon receiving the ciphertext σ at the receiver's side, the receiver node runs $UnSignCrypt(\sigma, ID_s, S_{ID_r})$ algorithm and outputs the message m or $\perp$ for "reject". If σ is not a valid ciphertext between the sender and the receiver the ciphertext is rejected. In the case of rejection, the fragment is assumed as a forged one and the receiver can drop it instantly. Note that, in the *UnSignCrypt* algorithm, each ciphertext C must be decrypted using the same symmetric encryption algorithm such as AES with $H_3(r)$ as the session decryption key and the plaintext m (which m is the fragment payload) must be obtained. Then, the fragment payload (about 600 bits) is given to the secure hash function SHA-1 to obtain a fixed 160 bits digest of the fragment payload. Finally, the hash value h is computed and the ciphertext is verified. This scheme provides integrity, confidentiality, authentication, and non-repudiation at the same time.

8 Analysis of the scheme

This section offers a detailed analysis in terms of consistency, security, and performance of the proposed scheme.

8.1 Consistency

Here the consistency of the proposed scheme is proved. First, Since:

$$T = xQ_r = x\left(H_1(ID_r)P + P_{pub}\right)$$
$$S_{IDr} = \frac{1}{H_1(ID_r) + msk}P$$

Therefore, it is found that:

$$\hat{e}(T, S_{IDr}) =$$
$$\hat{e}\left(x\left(H_1(ID_r)P + P_{pub}\right), \frac{1}{H_1(ID_r) + msk}P\right) =$$
$$\hat{e}\left(x(H_1(ID_r)P + mskP), \frac{1}{H_1(ID_r) + msk}P\right) =$$
$$\hat{e}(P,P)^{x(H_1(ID_r)+msk)\frac{1}{H_1(ID_r)+msk}} =$$
$$\hat{e}(P,P)^x = g^x = r$$

To recover the message m (fragment payload), the equation $m = Dec_{H_3(r)}(C)$ must be computed.

Second, since:

$$\varphi = \lambda(x + h) mod\ p$$
$$\mathrm{S} = \lambda^{-1}(S_{IDs} + P)$$
$$L = \varphi^* S = (x + h)(S_{IDs} + P)$$

The signature can be verified as follows:

$$\hat{e}\left(L, \frac{PQ_{ID_s}}{P + H_1(ID_s)P + P_{pub}}\right)g^{-h} =$$
$$\hat{e}\left(\varphi^* S, \frac{Q_{ID_s}}{P + H_1(ID_s)P + P_{pub}}P\right)g^{-h} =$$
$$\hat{e}\left(\lambda^{-1}(S_{IDs} + P).\lambda(x + h), \frac{H_1(ID_s)P + P_{pub}}{P + H_1(ID_s)P + P_{pub}}P\right)g^{-h} =$$
$$\hat{e}\left((x + h)(S_{IDs} + P), \frac{H_1(ID_s)P + mskP}{P + H_1(ID_s)P + mskP}P\right)g^{-h} =$$
$$\hat{e}\left((x + h)\left(\frac{1}{H_1(ID_s) + msk}P + P\right), \frac{P(H_1(ID_s) + msk)}{P(1 + H_1(ID_s) + msk)}P\right)g^{-h} =$$
$$\hat{e}\left((x + h)\left(\frac{1}{H_1(ID_s) + msk} + 1\right)P, \frac{H_1(ID_s) + msk}{1 + H_1(ID_s) + msk}P\right)g^{-h} =$$
$$\hat{e}\left((x + h)\left(\frac{1 + H_1(ID_s) + msk}{H_1(ID_s) + msk}\right)P, \frac{H_1(ID_s) + msk}{1 + H_1(ID_s) + msk}P\right)g^{-h} =$$
$$\hat{e}((x + h)P, P)^{\left(\frac{1+H_1(ID_s)+msk}{H_1(ID_s)+msk}\right)\left(\frac{H_1(ID_s)+msk}{1+H_1(ID_s)+msk}\right)}g^{-h} =$$
$$\hat{e}(P,P)^{(x+h)}g^{-h} = g^{x+h}g^{-h} = g^x = r$$

8.2 Security

In this section, it is shown that the proposed scheme provides adaptive-chosen ciphertext security (IND-CCA) and message unforgeability (EUF-CMA) through Theorems 1 and 2.

Theorem 1 Assume that an IND-CCA adversary F has an advantage ϵ against the proposed scheme when running in time t, asking q_{h_i} queries to random oracles H_i $(i = 1, 2, 3)$ and q_{un} unsigncryption queries, then an algorithm B exists that is able to solve the BDHIP with probability:

$$\epsilon' \geq \frac{\epsilon}{2q_{H_2} + q_{H_3}}\left(1 - \frac{q_{un}}{2^k}\right)$$

within a time

$$t' \leq t + O(q_{un})t_{pa} + O\left(q_{un}q_{H_2}\right)t_{exp},$$

Where t_{pa} is the cost of one pairing computation, and t_{exp} is the cost of an exponentiation computation in G_2.

Proof The algorithm B (challenger) takes as input $(P, \alpha P)$ and tries to extract $\hat{e}(P,P)^{1/\alpha}$ through interaction with F (adversary). In other words, B aims to solve a given instance $(P, \alpha P)$ of the BDHIP.

Setup phase: B selects a random master secret key $msk \in Z_p^*$ and calculates master public key $P_{pub} = msk.\ P$. Additionally, B computes the public key of receiver $Q_r^* = x\left(H_1\left(ID_r^*\right)P + P_{pub}\right)$ and $g = \hat{e}(P,P)$. Finally, B sends the public key of receiver Q_r^* and system parameters to F.

Phase 1: B is in charge of simulating the challenger of F in the IND-CCA game. Throughout the game, it is assumed that H_1 queries are distinct and a H_1 query on ID comes before all the queries including an identity ID. B stores and manages three lists L_1, L_2 *and* L_3 to simulate hash oracles H_1, H_2 *and* H_3.

- H_1 queries: While $H_1(ID_i)$ is queried, first B checks list L_1. If this value previously was assigned and is existed now, the value is retrieved from the list and returned. Otherwise B answers with a random value $h_{1,i} \in Z_p^*$ and appends the pair $(ID_i, h_{1,\ i})$ into the list L_1.
- H_2 queries: While $H_2(m_i, r_i, S_i, T_i)$ is queried, first B checks list L_2. If this value previously was assigned and is existed now, the value is retrieved from the list and returned. Otherwise, B returns a random value $h_{2,i} \in Z_p^*$ as an answer. To foresee possible further unsigncryption queries, B simulates the random oracle to get $h_{3,\ i} = H_3(r_i) \in \{0,1\}^n$ and computes $C_i = Enc_{h_{3,i}}(m)$ and $\vartheta_i = r_i.\hat{e}(P,P)^{h_{2,i}}$ and stores the information $(m_i, r_i, S_i, T_i, h_{2,\ i}, C_i, \vartheta_i)$ in the list L_2.
- H_3 queries: While $H_3(r_i)$ is queried, first B checks list L_3. If this value previously was assigned and is existed now for the input r_i, the value is retrieved and returned from the list. Otherwise, B selects the value $h_{3,\ i} \in \{0,1\}^n$ at random, returns it as an answer, and appends the pair $(r_i, h_{3,\ i})$ into the list L_3.
- Unsigncryption queries: For a ciphertext $\sigma = (C, \varphi, S, T)$ and identity of a sender ID_i, F can ask an unsigncryption

query without time restriction. B performs H_1 simulation algorithm to obtain $h_{1,\,i} = H_1(ID_i)$ and computes the private key of sender $S_{ID_i} = \frac{1}{h_{1,i}+msk} P$. For all valid ciphertexts

$$log_{(S_{ID_i}+P)}^{\varphi S - h(S_{ID_i}+P)} = log_{Q_r^*}^{T}$$

such that $h = H_2(m_i, r_i, S_i, T_i)$. Therefore, the equation

$$\hat{e}(T, S_{ID_i} + P) = \hat{e}\left(Q_r^*, \varphi S - h(S_{ID_i} + P)\right)$$

is correct. First, B computes $\vartheta = \hat{e}(\varphi S, Q_r^*)$, then she searches list L_2 for entries of the form $(m_i, r_i, S_i, T_i, h_{2,\,i}C, \vartheta)$ indexed by $i \in \{1, 2, \ldots, q_{H_2}\}$. If none found, then σ is rejected. Otherwise, B checks for the corresponding indexes if the equation

$$\frac{\hat{e}(T, S_{ID_i} + P)}{\hat{e}(Q_r^*, \varphi S)} = \hat{e}\left(Q_r^*, h(S_{ID_i} + P)\right)^{-h}$$

holds. If it is satisfied by a unique $i \in \{1, 2, \ldots, q_{H_2}\}$ then the matching tuple $(m_i, h_{2,\,i}, S_i)$ is returned which includes message m_i. Otherwise, σ is not accepted (reject). Here, a valid ciphertext is not accepted (reject) with the probability smaller than $\frac{q_{un}}{2^k}$. note that $Q_r^* = (H_1(ID_r^*)P + P_{pub})$.

Challenge phase: F produces two plaintexts m_0*and* m_1- with equal length and an identity of a sender ID_s^*, which it wants to be challenged. B randomly selects $C^* \in \{0,1\}^n, \zeta, \varphi^*, \in Z_p^*, S^* \in G_1$, computes $T* = \zeta P$, and produces the random ciphertext $\sigma^* = (C^*, \varphi^*, S^*, T^*)$. Finally, B returns σ^* to F. F is not able to distinguish that σ^* is a random ciphertext and is not a valid one unless he queries H_2 *or* H_3 on $\hat{e}(P,P)^{\beta}$. It is defined: $\beta = \zeta/\alpha$.

Phase 2: similar to phase 1, F can adaptively ask a polynomially bounded number of queries but he is not permitted to ask usigncryption query on σ^* to recover the relating plaintext. Again, B answers to the queries of F similar to phase 1.

Guess phase: During the guess phase, his vision is simulated as before, and his probable output is ignored (F generates a random bit τ', which is unknown for B).

To generate a result, B randomly fetches an entry $(m_i, r_i, S_i, T_i, h_{2,\,i}, C_i, \vartheta_i)$ or $(r_i, h_{3,\,i})$ from lists L_2 or L_3. As the number of records in the list L_3 does not exceed than $q_{H_2} + q_{H_3}$, the selected entry will contain the right element $r_i = \hat{e}(P,P)^{\beta}$ with the probability:

$$\frac{1}{2q_{H_2} + q_{H_3}}$$

Computing $\left(\hat{e}(P,P)^{\beta}\right)^{\zeta^{-1}}$ generates the solution of BDHIP.

In order to analyze the B's advantage, the following event is defined: E ≡ B abortion in result of a valid ciphertext rejection, in an unsigncryption query. As analyzed before, there is

$$Pr[E] = Pr[abort] \le \frac{q_{un}}{2^k}$$

Therefore, it is clear that the probability of not aborting B is equal to

$$Pr[\neg E] = Pr[\neg abort] \ge \left(1 - \frac{q_{un}}{2^k}\right).$$

Additionally, the probability of choosing the right element from L_2 *or* $L_3$by B is

$$\frac{\epsilon}{2q_{H_2} + q_{H_3}}$$

Therefore, it is found that:

$$\epsilon' \ge \frac{\epsilon}{2q_{H_2} + q_{H_3}} \left(1 - \frac{q_{un}}{2^k}\right)$$

B's computational time is obtained from the reality that in the unsigncryption queries, B performs $O(q_{un}\, q_{H_2})$ exponentiations in G_2 and $O(q_{un})$ pairing computations.

Theorem 2 In the random oracle model, if there existed an adaptively chosen message and identity attacker F that has an advantage

$$\epsilon \ge 10(q_s + 1)(q_s + q_{H_2})/2^k$$

Against the EUF-CMA security of the proposed scheme when making q_{H_i} queries to random oracles H_i $(i = 1, 2, 3)$ and q_s signcryption queries within a time t, then an algorithm B exists that is able to solve the q-SDHP for $q = q_{H_1}$ in the expected time

$$t' \le 120686\, q_{H_1}\, q_{H_2} \frac{t + O(q_s \tau_p)}{\epsilon(1 - 1/2^k)(1 - q/2^k)} + O(q^2 \tau_{mult})$$

Where τ_{mult} and τ_p respectively denote the cost of a scalar multiplication in G_1 and the required time for a pairing evaluation.

Proof Like [15], to prove that the proposed scheme is secure the forking lemma [32] is used. Before using this lemma, it is needed to prove that the proposed scheme is according to the signature scheme proposed in [32], the simulation phase in which the process of signature simulation can be done without the knowledge of sender's private key, and the way to solve q-SDHP based on the forgery.

Initially, it is clear that, the proposed scheme fulfills the prerequisite introduced in [32]. While the message m is signcrypted, a signature of the form $(\varepsilon_1, h, \varepsilon_2)$ is generated. Each of these elements corresponds to one of three requirements of the honest-verifier zero-knowledge protocol, where $\varepsilon_1 = r$ is the prover's obligation, $h = H_2(Tmp, r, S, T)$ is the value of hash function related to m and ε_1 replaced for the challenge of

verifier, and $\varepsilon_2 = \varphi S$ is the prover's reply that is dependent to ε_1, h and the sender's private key S_{ID_s} for signcryption.

Then, it is shown how B interacts with F to solve the q-SDHP through a faithful simulation between B and F. Algorithm B takes $(P, \alpha P, \alpha^2 P, \ldots, \alpha^q P)$ as input and aims to find a pair $\left(w, \frac{1}{\alpha+w} P\right)$. B simulates the challenger of F in the EUF-CMA game. Therefore, F asks signcryption and key generation queries in an adaptively manner. The process of the game is described as follows.

Setup phase: As in the proof technique of [33] first, B randomly picks $w_1, w_2, \ldots, w_{q-1} \in Z_p^*$ and takes $(P, \alpha P, \alpha^2 P, \ldots, \alpha^q P)$ as input and computes a generator $N \in G_1$ and another element $N_{pub} = \alpha N \in G_1$ such that it knows $q-1$ pairs $\left(w_i, \frac{1}{\alpha+w_i} P\right)$ for $w_1, w_2, \ldots, w_{q-1} \in Z_p^*$. It is defined:

$$U_i = \frac{1}{\alpha + w_i} N.$$

Then, B expands the polynomial

$$f(z) = \prod_{i=1}^{q-1} (z + w_i)$$

To compute $c_0, c_1, \ldots, c_{q-1} \in Z_p^*$ such that:

$$f(z) = \sum_{j=0}^{q-1} c_j z^j$$

It sets the generator N and an element N_{pub} respectively to:

$$N = \sum_{j=0}^{q-1} c_j \left(\alpha^j P\right) = f(\alpha) P$$

, and

$$N_{pub} = \sum_{j=1}^{q} c_{j-1} \left(\alpha^j P\right) = \alpha f(\alpha) P = \alpha N.$$

As in [33], B expands the polynomial

$$f_i(z) = \frac{f(z)}{z + w_i} = \sum_{j=0}^{q-2} d_j z^j$$

, and the equation

$$U_i = \sum_{j=0}^{q-2} d_j \left(\alpha^j P\right) = f_i(\alpha) P = \frac{f(\alpha)}{\alpha + w_i} P = \frac{1}{\alpha + w_i} N$$

to obtain the pair (w_i, U_i). Here, the public key of the PKG is N_{pub} and master secret key of the PKG is α. B sends the system public parameters, the generator N, N_{pub} and $g = \hat{e}(N, N)$ to F. Additionally, a randomly chosen identity $ID_s^* \in \{0, 1\}^*$ to be challenged is sent to F. Finally, B computes a pair of private/public keys of the receiver node $(Q_r^*, S_{ID_r^*})$ and sends them to F.

Attack phase: B is in charge of simulating the challenger of F in the EUF-CMA game. Throughout the game, it is assumed that H_1 queries are distinct and any query involving an identity ID comes after a H_1 query on ID. B stores and manages three lists L_1, L_2 *and* L_3 to simulate hash oracles H_1, H_2 *and* H_3.

- H_1 queries: First, B initializes a counter l to 1. On queries on an identity $ID \in \{0, 1\}^*$, B returns a random value $w_s \in Z_p^*$ if $ID = ID_s^*$ as the answer. Otherwise B answers a random value w_l and increments l. In both cases B inserts the pair (ID, w_s *or* w_l) into the list L_1.
- H_2queries: While $H_2(m_i, r_i, S_i, T_i)$ is queried, first B checks list L_2. If this value previously was assigned and is existed now, the value is retrieved from the list and returned. Otherwise, B returns a random value $h_{2,i} \in Z_p^*$ as the answer, and stores the information $(m_i, r_i, S_i, T_i, h_{2,i})$ in the list L_2.
- H_3queries: While $H_3(r_i)$ is queried, first B checks list L_3. If this value previously was assigned for the input r_i and is existed now, the value is retrieved from the list and returned. Otherwise, B selects a random value $h_{3,i} \in \{0, 1\}^n$, returns it as the answer, and appends the pair $(r_i, h_{3,i})$ into the list L_3.
- Key generation queries: while F asks a query to extract the private key of an identity ID_i, B aborts if $ID_i = ID_s^*$, otherwise B recovers the matching tuple (ID_i, w_i) from L_1 and returns $U_i = \frac{1}{\alpha+w_i} N$ to F (because knows that $H_1(ID_i) = w_i$).
- Signcryption query on message-identity pair (m, ID_i): while F selects an identity of a sender ID_i and a plain text m and asks a query to signcrypt m, if $ID_i \neq ID_s^*$ then B knows the private key of sender $(S_{ID_i} = U_i)$. Therefore, B can run the *OffSignCrypt* and *OnSignCrypt* algorithms and answer the asked query. If $ID_i = ID_s^*$ according to the irreflexivity assumption [14], B knows the private key of receiver $S_{ID_r^*}$. To reply this query, B first selects $\gamma, \varphi, h \in Z_p^*$ at random, computes the equations

$$S = \gamma \varphi^{-1} \left(S_{ID_r^*} + N\right), \quad T = \gamma \left(w_s^* N + N_{pub}\right) - \left(w_r^* N + N_{pub}\right)$$

and

$$r = \hat{e}\left(T, S_{ID_r^*}\right),$$

and sets the value of hash function $H_2(m_i, r_i, S_i, T_i)$ to h, computes $C = Enc_{H_3(r)}(m)$ and finally returns $\sigma = (C, \varphi, S, T)$ to F. Note that, to set H_2 to h, B fails if h way previously assigned but this occurs only with the probability:

$$\left(q_s + q_{H_2}\right)/2^k$$

Finally, the message m and the identity of sender ID_s^* must be integrated together to obtain a generalized forged message $\left(ID_s^*, m\right)$ such that the simulation of an adaptive-CMA existential forgery, independent of specific attacker is possible. For this simulation, the forking lemma is

proved. According to the forking lemma, in the previous interaction if there is an efficient forger F, then an algorithm F' exists that uses F to generate two valid signatures $(ID_s^*, m, h, \varphi, S, T)$ and $(ID_s^*, m, h^*, \varphi^*, S^*, T^*)$ such that $h \neq h^*$ but the commitment r is similar. Finally, an algorithm B to solve $q-SDHP$ based on the algorithm F' that derived from forger F, is defined. This algorithm works as follows: B takes two different signatures $(ID_s^*, m, h, \varphi, S, T)$ and $(ID_s^*, m, h^*, \varphi^*, S^*, T^*)$ as input which produced by F'. Then it computes

$$U^* = (h-h^*)^{-1}(\varphi S - \varphi^* S^*) - N.$$

From U^*, B can extract U^* as

$$U^* = \frac{1}{\alpha + w_s} N = \frac{f(\alpha)}{\alpha + w_s} P$$

Finally, using long division B can write the polynomial f as

$$f(z) = \Gamma(z)(z + w_s) + \Gamma_{-1}$$

such that.

$\Gamma(z) = \sum_{j=0}^{q-2} \Gamma_j z_j$ and $\Gamma_{-1} \in Z_p^*$.

Then, it is obtained that:

$$\frac{f(z)}{z + w_s} = \frac{\Gamma(z)(z + w_s) + \Gamma_{-1}}{z + w_s} = \Gamma(z) + \frac{\Gamma_{-1}}{z + w_s}.$$

Finally, B computes the equation

$$\frac{1}{\alpha + w_s} P = \frac{1}{\Gamma_{-1}} \left(U^* - \sum_{j=0}^{q-2} \Gamma_j (\alpha_i P) \right).$$

Therefore, the pair $\left(w_s, \frac{1}{\alpha + w_s} P\right)$ is the solution of the $q-SDHP$.

The relation between given-identity attack and chosen-identity attack is defined in [34]. Thus, from this and the forking lemma, it comes that if F is able to forge a signature within time t with probability

$$\epsilon \geq 10(q_s + 1)(q_s + q_{H_2})/2^k,$$

Then B can solve the $q-SDHP$ in the expected time

$$t' \leq 120686\, q_{H_1}\, q_{H_2} \frac{t + O(q_s \tau_p)}{\epsilon(1-1/2^k)(1-q/2^k)} + O(q^2 \tau_{mult})$$

8.3 Performance

In this section, the performance of the scheme will be detailed in terms of offline storage, computational cost, energy consumption, ciphertext size, private key size, and security. Table 1, compares the proposed scheme with LZZ [35], ECOOSC [36], LTX [37], and HOOSC [23]. The point multiplication in G_1 will be indicated by PM, the exponentiation in G_2 by E and the pairing computation by P. Since the other operations take negligible running time in comparing with pairing and point multiplication operations, they were ignored in the Table 1. |d| indicates the number of bits to store d. As summarized in the Table 1, it is clear that the LZZ, ECOOSC, LTX, HOOSC require 5, 4, 2, 2 point multiplication in the *OffSignCrypt* algorithm respectively, whereas the proposed scheme requires 3 point multiplication. It implies that the cost of computation of the proposed scheme is lower than the LZZ, ECOOSC and is somewhat higher than the LTX, HOOSC. Note that all of them perform one exponentiation in the *OffSignCrypt* algorithm. In the *OnSignCrypt* algorithm, since both the LZZ and LTX perform one point multiplication whereas the others do not perform any of these operations, the cost of computation of the proposed scheme is lower than the LZZ and LTX and is equal to the ECOOSC and HOOSC. Finally, in the *UnSignCrypt* algorithm, the cost of computation of the proposed scheme is lower than the LZZ and LTX and is higher than the HOOSC. In terms of offline storage, this scheme needs less storage than the LZZ, and needs more storage than the LTX and is equal to the others. In terms of private key size and ciphertext size, this scheme's key size is smaller than the LZZ, LTX and is equal to the others. As discussed in [38], the LTX is not secure.

A quantitative analysis in terms of computational cost, ciphertext size, offline storage, and energy consumption is offered for the LTX, LZZ, ECOOSC, HOOSC, and ours. It is assumed that $|m| = |\textit{fragment payload}| \approx 600$ bits, |node's ID| =

Table 1 Comparison of proposed schemes

Schemes	OffSignCrypt			OnSignCrypt			UnSignCrypt			Offline storage	Ciphertext size	Private key size	Security
	PM	E	P	PM	E	P	PM	E	P				
LTX[37]	2	1	0	1	0	0	4	0	3	$2\lvert G_1\rvert + \lvert G_2\rvert + \lvert Z_p^*\rvert$	$\lvert m\rvert + 2\lvert G_1\rvert + 2\lvert Z_p^*\rvert$	$\lvert G_1\rvert + \lvert Z_p^*\rvert$	No
LZZ[35]	5	1	0	1	0	0	5	1	5	$4\lvert G_1\rvert + \lvert G_2\rvert + 3\lvert Z_p^*\rvert$	$\lvert m\rvert + 4\lvert G_1\rvert + 2\lvert Z_p^*\rvert$	$\lvert G_1\rvert + \lvert Z_p^*\rvert$	YES
ECOOSC[36]	4	1	0	0	0	0	3	1	2	$2\lvert G_1\rvert + \lvert G_2\rvert + 2\lvert Z_p^*\rvert$	$\lvert m\rvert + 2\lvert G_1\rvert + \lvert Z_p^*\rvert$	$\lvert G_1\rvert$	YES
HOOSC[23]	2	1	0	0	0	0	2	1	2	$2\lvert G_1\rvert + \lvert G_2\rvert + 2\lvert Z_p^*\rvert$	$\lvert m\rvert + 2\lvert G_1\rvert + \lvert Z_p^*\rvert$	$\lvert G_1\rvert$	YES
Ours	3	1	0	0	0	0	3	1	2	$2\lvert G_1\rvert + \lvert G_2\rvert + 2\lvert Z_p^*\rvert$	$\lvert m\rvert + 2\lvert G_1\rvert + \lvert Z_p^*\rvert$	$\lvert G_1\rvert$	YES

Table 2 Computational time and energy consumption of the SHA-1 and AES128

Algorithm	Energy consumption(μJ)	Computational time(ms)
SHA-1	154	15
AES 128	680	10.26

80 bits, $|G_1| = 542$ bits, $|G_2| = 1084$ bits, $|p| = 252$ bits (|G| denotes the size of G's element). If standard compression technique is used, the size of each element in G_1 can be reduced to 272 bits [39]. The result of experiment in [40] is applied on MICA2 that is armed with an ATmega128 8-bit processor clocked at 7.3728 MHz, 4 KB RAM and 128 KB ROM. As said in [41], a point multiplication takes 0.81 s and as said in [40] an exponentiation operation in G_2 takes 0.9 s and a pairing operation takes 1.9 s, using the super-singular curve $y^2 + y = x^3 + x$ with an embedding degree 4 and implementing η_T pairing: $E(F_{2^{271}}) \times E(F_{2^{271}}) \rightarrow F_{2^{4.271}}$, that is equivalent to the 80-bit security level. Therefore, based on the computational cost of each operation and the number of operations in each algorithm summarized in the Table 1, the total computational cost of each scheme is summarized in the Table 3 and shown in Fig. 8. From [40, 42–44], it is known that the current draw of MICA2 in receiving mode is 10 mA, in transmission mode is 27 mA, and in active mode is 8.0 mA. Also, the power level is 3.0 V, and the data rate is 12.4 Kbps, a point multiplication operation consumes 19.44 mJ, an exponentiation operation in G_2 consumes 21.6 mJ and a pairing operation consumes 45.6 mJ. Thus, according to the energy consumption of each operation and the number of operations in each algorithm that are summarized in the Table 1, Table 3 summarizes the total energy consumption of each scheme and it is shown in Fig. 9. Note that, according to [45, 46] the computational cost and energy consumption of the AES and SHA-1 algorithms that are used in the proposed scheme are negligible on the MICA2, and are summarized in the Table 2 and 3.

In order to compute communication cost, the ciphertext size must be computed according to the used parameters. Therefore, in the LTX the sender needs to transmit 1648 bits to send one fragment:

$$|\boldsymbol{m}| + 2|\boldsymbol{G}_1| + 2\left|\boldsymbol{Z}_p^*\right| = 600 + 2^*272 + 2^*525 = 1648\ \boldsymbol{bits}$$

In the LZZ, the sender needs to transmit 2192 bits to send one fragment:

$$|\boldsymbol{m}| + 4|\boldsymbol{G}_1| + 2\left|\boldsymbol{Z}_p^*\right| = 600 + 4^*272 + 2^*252 = 2192\ \boldsymbol{bits}$$

In the ECOOSC, HOOSC and the proposed scheme, to send one fragment a total transmission of 1396 bits is needed:

$$|\boldsymbol{m}| + 2|\boldsymbol{G}_1| + \left|\boldsymbol{Z}_p^*\right| = 600 + 2^*272 + 252 = 1396\ \boldsymbol{bits}$$

As said in [39], the energy consumption to send one bit is $27*3 * 12{,}400 = 0.0065$ mJ and the energy consumption to receive one bit is $10 * 3 * 12{,}400 = 0.0024$ mJ. Therefore, in ECOOSC, HOOSC and the proposed scheme the energy consumption of communication in the sender and receiver nodes are 0.0065*1396 = 9.074 mJ and 0.0024*1396 = 3.350 mJ respectively. In LTX the communication energy consumption of the sender and receiver node are 0.0065*1648 = 10.712 mJ and 0.0024*1648 = 3.955 mJ respectively. In LZZ the communication energy consumption of the sender and receiver node are 0.0065*2192 = 14.248 mJ and 0.0024*2192 = 5.260 mJ respectively. Thus, the total energy consumption (including *OffSignCrypt*, *OnSignCrypt*, *UnSignCrypt*, send, receive) of the LTX, LZZ, ECOOSC, HOOSC and the proposed scheme respectively are equal to:

$$\mathbf{60.48 + 19.44 + 214.56 + 10.712 + 3.955 = 309.147\ mJ},$$
$$\mathbf{118.8 + 19.44 + 346.8 + 14.248 + 5.260 = 504.548\ mJ}$$
$$\mathbf{99.36 + 171.12 + 9.074 + 3.350 = 282.904\ mJ},$$
$$\mathbf{60.48 + 151.68 + 9.047 + 3.350 = 224.584\ mJ}\ \text{and}$$
$$\mathbf{79.92 + 171.12 + 9.074 + 3.350 = 263.464\ mJ}$$

The results are depicted in Fig. 9. The offline storage of the LTX, LZZ, ECOOSC, HOOSC, and the proposed scheme respectively are equal to:

$$2|\boldsymbol{G}_1| + |\boldsymbol{G}_2| + \left|\boldsymbol{Z}_p^*\right| = 2^*272 + 1084 + 252 = 1880\ \boldsymbol{bits}$$

$$4|\boldsymbol{G}_1| + |\boldsymbol{G}_2| + 3\left|\boldsymbol{Z}_p^*\right| = 4^*272 + 1084 + 3^*252 = 2928\ \boldsymbol{bits}$$

$$2|\boldsymbol{G}_1| + |\boldsymbol{G}_2| + 2\left|\boldsymbol{Z}_p^*\right| = 2^*272 + 1084 + 2^*252 = 2132\ \boldsymbol{bits}$$

Table 3 Computational time and energy consumption of proposed schemes

Schemes	OffSignCrypt		OnSignCrypt		UnSignCrypt	
	Time (s)	Energy (mJ)	Time (s)	Energy (mJ)	Time (s)	Energy (mJ)
LTX[37]	2.52	60.48	0.81	19.44	8.94	214.56
LZZ[35]	4.95	118.8	0.81	19.44	14.45	346.8
ECOOSC[36]	4.14	99.36	0	0	7.13	171.12
HOOSC[23]	2.52	60.48	0	0	6.32	151.68
Ours	3.33	79.92	0	0	7.13	171.12

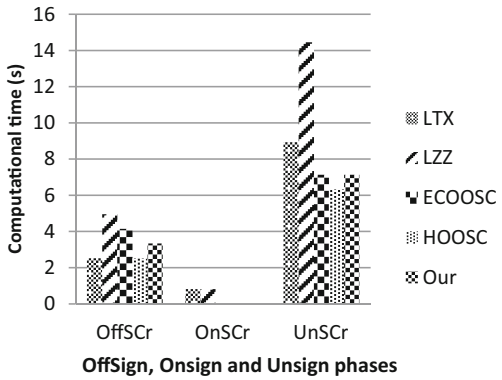

Fig. 8 Computational time of the schemes

The size of private key in LTX and LZZ is equal to

$$|\boldsymbol{G}_1| + \left|\boldsymbol{Z}_p^*\right| = 272 + 252 = 524\ \boldsymbol{bits}$$

whereas, in ECOOSC, HOOSC, and the proposed scheme it is equal to $|\boldsymbol{G}_1| = 272$ ***bits***. The results are depicted in Fig. 10.

Finally, the proposed scheme can signcrypt and send a fragment of 600 bits size whereas, the other schemes designed to operate on a message of 160 bits size. This is the most important advantage and novelty of this scheme. Thus, the relative computational time and energy consumption of the proposed scheme are significantly lower in comparison with the other schemes.

8.4 Processing out-of-order fragment

Since in the proposed scheme each fragment is signcrypted individually and there is no cryptographic dependency between the fragments of a packet, therefore, each fragment can be processed at the receiving time, processing out-of-order fragments is possible, and the spoofed ones can be dropped immediately. Since, in the content-chaining scheme [12] each fragment is authenticated by a hashed token value, and this token is attached to its previous fragment, therefore the content-chaining scheme is not able to process out-of-order fragments at the receiving time. A malicious node can leverage this fact to occupy the recipient's reassembly buffer with fragments that seems to be legitimate out-of-order fragments.

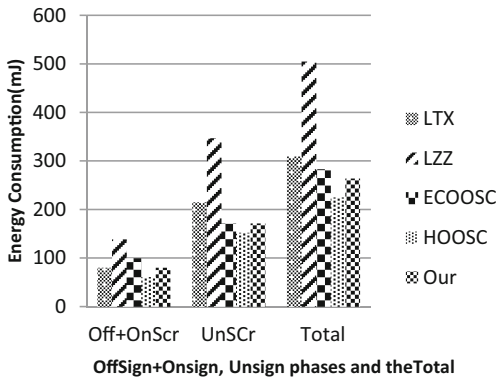

Fig. 9 Energy consumption of the schemes

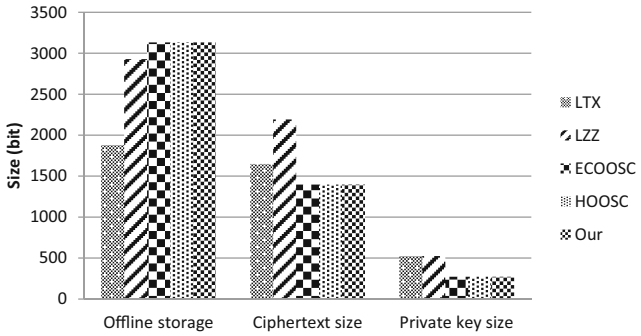

Fig. 10 Offline storage, ciphertext size and private key size of the schemes

8.5 Packet overhead

It is clear that, original fragments (without signcryption) do not contain signcryption elements(φ, S, T). But in the case of signcryption, φ, *S and T* are attached to the fragment. It means that the proposed scheme adds 796 bits ≈ 99 bytes per fragment. The size of an IPv6 packet is 1280 bytes and it is broken to 17 fragments, each fragment size is 81 bytes. Securing the 6LoWPAN fragments of an IPv6 packet needs to additional 2761 bytes. Since the signcryption elements are attached to the fragment payload, the fragment payload is decreased, thus more fragments must be created. These two factors cause the 2761 bytes extra overhead. Considering these numbers, this overhead is equal to 120% of the whole packet size. Note that before signcryption, to transmit an IPv6 packet, 2180 bytes must be transmitted but after signcryption, 4041 bytes must be transmitted.

9 Conclusion

In this paper, the vulnerabilities of fragmentation mechanism in the 6LoWPAN networks were analyzed. The analysis focused on attackers who are internal nodes of the network. The proposed 6LoWPAN network environment includes resource-constrained nodes and the fragment duplication attack was introduced. In addition, a lightweight offline/online signcryption scheme to mitigate fragment duplication attack in 6LoWPAN networks was proposed. The proposed scheme permits a sender node to send a signcrypted fragment to a receiver node. In comparison with two proposed approaches: content-chaining scheme and attaching timestamp to the fragments, the proposed scheme efficiently provides integrity, confidentiality, non-repudiation, and authentication in a logical single step and can easily process out-of-order fragments.

Comparing some offline/online signcryption schemes with the proposed scheme shows that this new scheme does not need any point multiplication operation in the online phase. Additionally, there is no need to public key authenticity verification by a certificate. These characteristics make this scheme appropriate for resource-constrained networks such as the IoT. The evaluation shows that the proposed scheme is secure in the random oracle model and efficiently counters this attack.

References

1. Tsai CW, Lai CF, Vasilakos AV (2014) Future internet of things: open issues and challenges. J Wireless Networks 20(8):2201–2217
2. Roman R, Zhou J, Lopez J (2013) On the features and challenges of security and privacy in distributed internet of things. J Computer Networks 57(10):2266–2279
3. Jing Q, Vasilakos AV, Wan J, Lu J, Qiu D (2014) Security of the internet of things: perspectives and challenges. J Wireless Networks 20(8):2481–2501
4. Kim E, Kaspar D, Vasseur J (2012) Design and application spaces for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs). https://tools.ietf.org/html/rfc6568. Accessed April 2012
5. IEEE. Part 15.4: wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (WPANs) (2006). IEEE 802.15.4, IEEE Computer Society, 2006
6. Zheng Y (1977) Digital signcryption or how to achieve cost (signature & encryption) < cost (signature) + cost (encryption). Adv Cryptol Lect Notes Comput Sci 1294:165–179
7. Boneh D, Franklin M (2001) Identity-based encryption from the weil pairing. Adv Cryptol, Lect Notes Comput Sci 2139:213–229
8. Kim H (2007) Protection against packet fragmentation attacks at 6LoWPAN adaptation layer. In: Convergence and Hybrid Information Technology, 2008. In: Proceedings. 2008 IEEE International Conference on, pp 796–801
9. Montenegro G, Kushalnagar N, Hui J, Culler D (2007) Transmission of IPv6 packets over IEEE 802.15.4 networks. https://tools.ietf.org/html/rfc4944, Accessed September 2007
10. Ziemba G, Reed D,Traina P (1995) Security considerations for IP fragment filtering. https://tools.ietf.org/html/rfc1858.html, Accessed October 1995
11. Ptacek T, Newsham T (1998) Insertion, evasion, and denial of service: eluding network intrusion detection. Eluding network intrusion detection. SECURE NETWORKS INC CALGARY ALBERTA
12. Hummen R, Hiller J, Wirtz H, Henze M, Shafagh H, Wehrle K (2013) 6LoWPAN fragmentation attacks and mitigation mechanisms. In: security and privacy in wireless and mobile networks, 2013. WiSec'13. In: Proceedings. 2013 6th ACM conference on, pp 55-66
13. Libert B, Quisquater JJ (2003) A new identity based signcryption schemes from pairings. In: Proceedings of the 2003 IEEE workshop on information theory, pp 155–158
14. Boyen X (2003) Multipurpose identity-based signcryption: a swiss army knife for identity-based cryptography. Adv Cryptol Lect Notes Comput Sci 2729:383–399
15. Barreto PSLM, Libert B, McCullagh N, Quisquater JJ (2005) Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. Adv Cryptol Lect Notes Comput Sci 3788:515–532
16. Jo HJ, Paik JH, Lee DH (2014) Efficient privacy preserving authentication in wireless mobile networks. Trans Mobile Comput IEEE 13(7):1469–1481
17. An JH, Dodis Y, Rabin T (2002) On the security of joint signature and encryption. Advances in cryptology, Eurocrypt 2002, lecture notes in computer science 2332: 83–107
18. Xu Z, Dai G, Yang D (2007) An efficient online/offline signcryption scheme for MANET. In: proceedings of the 2007 workshop on advanced information networking and applications, pp 171–176
19. Yan F, Chen X, Zhang Y (2013) Efficient online/offline signcryption without key exposure. J Grid Util Comput 4(1):85–93
20. Sun D, Huang X, Mu Y, Susilo W (2008) Identity-based on-line/off-line signcryption. In: network and parallel computing, 2008. In: proceedings. 2008 IFIP international conference on, pp 34–41
21. Liu JK, Baek J, Zhou J (2011) Online/offline identity based signcryption re-visited. In: information security and cryptology, Inscrypt 2010, lecture notes in computer science 6584: 36–51
22. Li F, Khan MK, Alghathbar K, Takagi T (2012) Identity-based online/offline signcryption for low power devices. J Network Comput Appl 35(1):340–347
23. Li F, Xiong P (2013) Practical secure communication for integrating wireless sensor networks into the internet of things. J IEEE Sensors 13(10):3677–3684
24. Senthil kumaran U, Ilango P (2015) Secure authentication and integrity techniques for randomized secured routing in WSN. J Wireless Networks 21(2):443–451
25. Li F, Zheng Z, Jin C (2016) Secure and efficient data transmission in the internet of things. J Telecommun Syst 62(1):111–122
26. Bormann C (2012) Guidance for light-weight implementations of the internet protocol suite. https://tools.ietf.org/html/draft-bormann-lwig-guidance-01, Accessed 24 January 2012
27. Wilhelm M, Martinovic I, Schmitt JB, Lenders V (2011) reactive jamming in wireless networks: how realistic is the threat?. In: wireless network security, 2011. WiSec'11. In: Proceedings. 2011 4th ACM conference on, pp 47–52
28. Becher A, Benenson Z, Dornseif M (2006) Tampering with motes: real-world physical attacks on wireless sensor networks. In: security in pervasive computing, 2006. SPC'06. In: Proceedings. 2006 3rd international conference on, pp 104–118
29. Heer T, Garcia-Morchon O, Hummen R, Keoh S, Kumar S, Wehrle K (2011) Security challenges in the IP-based internet of things. J. Wirel Pers Commun 61(3):527–542
30. Daemen J, Rijmen V (2002) The design of Rijndael: AES the advanced encryption standard. Springer, Berlin
31. Secure Hash Standard (1995) Nat'l Inst. of standards and technology (NIST), Fed. Inf Process Stand Publ 180(1)
32. Pointcheval D, Stern J (2000) Security arguments for digital signatures and blind signatures. J Cryptology 13(3):361–396
33. Boneh D, Boyen X (2004) Short signatures without random oracles. In: advances in cryptology. Lect Notes Comput Sci 3027:56–73
34. Cha JC, Cheon JH (2003) An identity-based signature from gap Diffie-Hellman groups. Public Key Cryptogr, Lect Notes Comput Scie 2567:18–30
35. Li J, Zhao J, Zhang Y (2015) Certificateless online/offline signcryption scheme. J Secur Commun Netw 8(11):1979–1990
36. Li F, Han Y, Jin C (2017) Certificateless online/offline signcryption for the internet of things. J. Wirel Netw 23(1):145–158
37. Luo M, Tu M, Xu J (2014) A security communication model based on certificateless online/offline signcryption for internet of things. J Sec Commun Netw 7(10):1560–1569
38. Shi W, Kumar N, Gong P, Chilamkurti N, Chang H (2015) On the security of a certificateless online/offline signcryption for internet of things. J Peer-to-Peer Network Appl 8(5):881–885
39. Shim KA (2012) CPAS: an efficient conditional privacy preserving authentication scheme for vehicular sensor networks. Trans Veh Technol IEEE 61(4):1874–1883

40. Shim KA, Lee YR, Park CM (2013) EIBAS: an efficient identity-based broadcast authentication scheme in wireless sensor networks. J Ad Hoc Netw 11(1):182–189
41. Gura N, Patel A, Wander A, Eberle H, Shantz SC (2004) Comparing elliptic curve cryptography and RSA on 8-bit CPUs. In: international workshop on cryptographic hardware and embedded systems, 2004. CHES'04. Lect Notes Comput Sci 3156:119–132
42. Cao X, Kou W, Dang L, Zhao B (2008) IMBAS: Identitybased multi-user broadcast authentication in wireless sensor networks. J. Comput Commun 31(4):659–667
43. Ma C, Xue K, Hong P (2014) Distributed access control with adaptive privacy preserving property for wireless sensor networks. J Secur Commun Netw 7(4):759–773
44. Shim KA (2014) S2DRP: secure implementations of distributed reprogramming protocol forwireless sensor networks. J. Ad Hoc Netw 19:1–8
45. Chang C, Muftic S (2007) Measurement of energy costs of security in wireless sensor nodes. In: computer communications and networks, 2007. ICCCN'07. In: proceedings. 2007 IEEE 16th international conference on, pp 95–102
46. Prasithsangaree P, Krishnamurthy P (2003) Analysis of energy consumption of Rc4 and AES algorithms in wireless Lans. In: Global telecommunications, 2003. GLOBECOM'03. 2003 IEEE Conference on 3: 1445–1449
47. Robinson DJS (1996) A course in the theory of groups. Springer, Heidelberg
48. Martin L (2008) Introduction to identity-based encryption. Artech House, Boston, London, England
49. Rescorla E, Modadugu N (2012) Datagram transport layer security, http://www.rfc-editor.org/rfc/rfc6347.txt. Accessed January 2012

Mohammad Nikravan received his B.Sc. in Computer Software Engineering from the Islamic Azad University, Mashhad branch, Iran in 2002. He also received M.Sc. degree in Computer Software from the South Tehran Branch, Islamic Azad University, Tehran, Iran in 2005. He is currently working toward the Ph. D. degree in Computer Software Engineering at Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran. Currently he is faculty of Department of Computer Engineering Shahr-e-Qods Branch, Islamic Azad University in Shahr-e-Qods, Iran. His research interests include Internet of things, 6LoWPAN networks and cryptography and network security.

Mailing address: Shahr-e-Qods Branch, Shahid Kalhor Blvd, Shahr-e-Qods, Tehran, Iran.
Post Office Box: 37541–374. Postal Code: 3754113115.
Tell: +98(21) 46896000
Email address: moh_nikravan@yahoo.com

Ali Movaghar is a Professor in the Department of Computer Engineering at Sharif University of Technology in Tehran, Iran and has been on the Sharif faculty since 1993. He received his B.S. degree in Electrical Engineering from the University of Tehran in 1977, and M.S. and Ph.D. degrees in Computer, Information, and Control Engineering from the University of Michigan, Ann Arbor, in 1979 and 1985, respectily. He visited the Institut National deRecherche en Informatique et en Automatique in Paris, France and the Department of Electrical Engineering and Computer Science at the University of California, Irvine in 1984 and 2011, respectively, worked at AT&T Information Systems in Naperville, IL in 1985–1986, and taught at the University of Michigan, Ann Arbor in 1987–1989. His research interests include performance/dependability modeling and formal verification of wireless networks and distributed real-time systems. He is a senior member of the IEEE and the ACM.

Mailing address: Department of Computer Engineeing, Sharif University of Technology, Azadi Ave, Tehran, Iran.
Post Office Box: 11386–8639
Tell: +98(21) 66013126
Email Address: movaghar@sharif.edu

Mehdi Hosseinzadeh received his B.E. degree in Computer Hardware Engineering from Islamic Azad University (IAU), Dezful branch, Iran in 2003. He also received his M.Sc. and the Ph.D. degree in Computer System Architecture from the Science and Research Branch, IAU, Tehran, Iran in 2005 and 2008, respectively. Mehdi is currently an Associate professor in Iran University of Medical Sciences (IUMS), Tehran, Iran, and his research interests include Information Technology, Data Mining, Big data analytics, E-Commerce, E-Marketing and Social Networks.

Mailing address: Iran University of Medical Sciences, Tehran, Iran.
Post Office Box: 14515–775.
Postal Code: 1477893855.
Tell: +98(21) 44865179
Email address: hosseinzadeh.m@Iums.ac.ir