Questioning the security and efficiency of the ESIoT approach

Aida Diop, Said Gharout, Maryline Laurent, Jean Leneutre, Jacques Traoré

Questioning the security and efficiency of the ESIoT approach

Aïda Diop - Orange Labs [aida.diop@orange.com]
Saïd Gharout - Orange Labs [said.gharout@orange.com]
Maryline Laurent - Télécom SudParis [maryline.laurent@telecom-sudparis.eu]
Jean Leneutre - Télécom ParisTech [jean.leneutre@telecom-paristech.fr]
Jacques Traoré - Orange Labs [jacques.traore@orange.com]

Abstract

ESIoT is a secure access control and authentication protocol introduced for Internet of Things (IoT) applications. The core primitive of ESIoT is an identity-based broadcast encryption scheme called Secure Identity-Based Broadcast Encryption (SIBBE). SIBBE is designed to provide secure key distribution among a group of devices in IoT networks, and enable devices in each group to perform mutual authentication. The scheme is also designed to hide the structure of the group from nodes outside of the group. We identify multiple efficiency and security issues in this primitive that prove SIBBE unsuitable for IoT applications. First, we show that contrary to what was claimed, the size of the ciphertexts generated by the encryption function is not constant but in fact linear in the number of devices in the group. Additionally, we demonstrate that the encryption and decryption costs are also linear in the number of nodes in the group, implying scalability issues thus inefficiency for IoT applications. In terms of security, we prove that SIBBE does not achieve the desired property of anonymity and allows an attacker to gain information on the structure of any given group. Finally, we demonstrate how SIBBE does not achieve the claimed chosen-ciphertext security. We however prove its security for a weaker security notion (namely selective-ID indistinguishability against chosen-plaintext attacks) under a variant of the GDDHE assumption.

1 INTRODUCTION

The Internet of Things (IoT) is a paradigm where distributed devices form a wireless network and communicate over the Internet. We consider the case of Smart buildings with hundreds of heterogeneous smart devices forming a low-power network. Each of these wireless nodes has an IP address, and can use IP protocols to communicate data over IPv6 Low-Power Area Network (6LoWPAN). In order to secure communications in such setting, it is important to establish a secure key distribution model.

ESIoT protocol. ESIoT [10] is a security protocol for managing access control and authentication in IoT networks. ESIoT's architecture is based on the *commissioner model* introduced by the THREAD industry consortium [9]. The model comprises a central server, devices divided into *policy groups* that perform a specific task (or policy), and a designated node called the *commissioner* that is in charge of managing the policy group. The group tasks can for example include sending temperature data from sensors, under the supervision of the commissioner. ESIoT differentiates itself from existing protocols that provide secure access control by not revealing the structure of the policy groups in clear. In ESIoT, authenticity, confidentiality, privacy, and access control are provided by the SIBBE primitive in conjunction with digital signatures, AES encryption and Hash-based Message Authentication Code (HMAC). The security properties achieved by SIBBE are therefore crucial to the overall design of the protocol.

IBBE/SIBBE. Broadcast Encryption (BE) schemes were introduced by Fiat and Naor in [6] as key-distribution mechanisms that allow a source to broadcast an encrypted message, such that only target receivers are able to decrypt the ciphertext using their private decryption keys. Identity-Based Encryption schemes (IBE) were introduced by Shamir in [11] to provide a more flexible alternative to the centralized certificate-based approach in standard Public-key Encryption schemes. Each encrypting party can use a public information (ID) to encrypt messages, and the private decryption key related to the public ID provides the authentication layer that was previously delegated to the certification authority. The combination of these two schemes is the concept of Identity-Based Broadcast Encryption (IBBE). The first IBBE scheme providing constant-size ciphertexts and private keys was introduced by Delerablée [5].
Kim et al. introduced a new Identity-Based Broadcast Encryption scheme [10] called SIBBE, as the main building block for their ESIoT protocol. Their construction draws inspiration from [5], but aims to differentiate itself

by providing a more efficient decryption function and by hiding the identities of the receivers (anonymous IBBE).

Our contribution. We analyse the efficiency and security of SIBBE, and show that it is neither practical nor secure for IoT networks. We first analyse in section 4.1 the efficiency of the scheme, and show that the size of the ciphertexts as well as the decryption cost are in fact linear in the number of receivers, and not constant as claimed in [10]. Second, we provide in sections 4.2 and 4.3 an analysis of the efficiency of SIBBE in practice. We also prove that SIBBE is not secure against chosen-ciphertext attacks in section 5.1, and that it does not satisfy the security definition of an anonymous IBBE in section 5.2. Finally, in section 5.3 we prove that SIBBE is secure under a weaker security definition.

2 PRELIMINARIES

In this section, we review the notations we use, the mathematical tools employed in the cryptographic primitives, and the assumption upon which we construct our security proof. We also give a formal description of an IBBE scheme, and the security definitions for IBBE schemes.

Notations. We use the notation $A \Leftarrow B$ for assignement, $\mathbb{Z}_p$ to designate a group of prime order p, and $x \xleftarrow{\$} G$ to denote that x is chosen uniformly at random in the group G. $\{0,1\}^*$ represents the set of bitstrings of arbitrary length, and $H : G_1 \to G_2$ is used to designate a function from G_1 to G_2.

2.1 Bilinear Maps

The construction of SIBBE is based on bilinear pairings. A bilinear pairing is defined as follows:

Let p be a prime number and $\mathbb{G}_1 = \langle P_1 \rangle$, $\mathbb{G}_2 = \langle P_2 \rangle$ two cyclic groups of order p generated by P_1 and P_2 respectively [1]. Let $\mathbb{G}_T$ be a finite field of order p. A pairing is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with the following properties:

Bilinearity: For $P \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$ and $(a, b) \in \mathbb{Z}_p^2$,
$e(P^a, Q^b) = e(P, Q)^{ab} = e(P^b, Q^a)$.

Non-degeneracy: For $P \neq 1$ and $Q \neq 1$, $e(P, Q) \neq 1$.

Computability: e is efficiently computable.

[1] Throughout this paper, for simplicity we will use the multiplicative notation for the binary operations in $\mathbb{G}_1$ and $\mathbb{G}_2$.

2.2 GDDHE Assumption

In section 5.3, we prove SIBBE to be secure under a weaker security notion than the one presented in [10]. The proof is based on a variant of the general decisional diffie-hellman exponent assumption (GDDHE) introduced by Delerablée in [5], and proven secure in the generic group model [5]. The decision problem is defined as follows:

Definition 1. *($(f,g,F) - GDDHE$ problem.)*
Let $\mathcal{B} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e(.,.))$ be the description of a bilinear map and let f, g be two coprime polynomials with pairwise distinct roots of respective orders t and n. Let g_0 be a generator of $\mathbb{G}_1$ and h_0 a generator of $\mathbb{G}_2$. Solving the $(f,g,F) - GDDHE$ problem consists, given:

$$g_0, g_0^{\gamma}, ..., g_0^{\gamma^{t-1}}, g_0^{\gamma.f(\gamma)}, g_0^{k.\gamma.f(\gamma)}$$
$$h_0, h_0^{\gamma}, ..., h_0^{\gamma^{2n}}, h_0^{k.g(\gamma)}$$

and $R \in \mathbb{G}_T$, in deciding whether R is equal to $e(g_0, h_0)^{k.f(\gamma).g(\gamma)}$ or to some random element of $\mathbb{G}_T$.
The advantage of an algorithm $\mathcal{A}$ in distinguishing the two distributions is denoted by $Adv_{GDDHE(f,g,F,\mathcal{A})}$. The security of schemes under the GDDHE assumption is evaluated over all polynomial-time adversaries. Therefore we will consider the following advantage: $Adv_{GDDHE(f,g,F)} = max(Adv_{GDDHE(f,g,F,\mathcal{A})})$ over all polynomial-time adversaries.

2.3 Identity-Based Broadcast Encryption Scheme

Formally, an IBBE scheme comprises the following algorithms:
Setup(λ, n). Takes as input a security parameter λ that polynomially bounds the running time of the algorithms, and n the maximal size of the set of receivers. The algorithm returns a master secret key MSK and public parameters *param*.
Extract(MSK, ID_i). Takes as input MSK and a user identity ID_i. The algorithm generates a private decryption key d_i.
Encrypt$(param, \mathcal{I})$. Takes as input *param* and a set of public identities $\mathcal{I} = \{ID_1, ..., ID_s\}$ with $s \leq n$, and outputs a pair (Hdr, K) where Hdr is called the header (broadcast ciphertext) and K is the symmetric encryption key. The broadcaster encrypts a message M under the symmetric key K: $C = Enc_K(M)$ (C is called the broadcast body), and broadcasts $(Hdr, \mathcal{I})$.
Decrypt$(param, \mathcal{I}, ID_i, d_i, Hdr)$. Takes as input the public parameters

param, a subset $\mathcal{I} = \{ID_1, ..., ID_s\}$ of identities, an identity ID_i, the corresponding private key d_i, and a header Hdr. If $ID_i \in \mathcal{I}$, the algorithm outputs K which is then used to decrypt the message M.

2.4 Security Definitions for Identity-Based Broadcast Encryption Schemes

The security definition SIBBE claims to satisfy is selective-ID indistinguishability against chosen-ciphertext attacks (IND-sID-CCA). We provide the formal definition of a selective-ID IND-CCA secure scheme introduced by Canetti et al. [4], where the adversary has access to a decryption oracle and must choose the identities he wants to attack at the beginning of the game. The formal notion of an IND-sID-CCA secure scheme is defined with the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:
Init: $\mathcal{A}$ outputs a set of identities $\mathcal{I} = \{ID_1, ..., ID_s\}$ it wants to attack ($s \leq n$ where n is the maximal number of receivers).
Setup: The challenger $\mathcal{C}$ runs $Setup(\lambda, n)$ to obtain the public parameters *param* and the master secret key MSK. He gives $\mathcal{A}$ the public parameters *param*.
Query Phase 1: $\mathcal{A}$ adaptively issues queries $q_1, ..., q_{s_0}$ where q_i can be one of the following:

- *Extraction query (ID_i)*: with the constraint that $ID_i \notin \mathcal{I}$. $\mathcal{C}$ runs $Extract(MSK, ID_i)$ and forwards the resulting private key d_i to $\mathcal{A}$.
- *Decryption query (param, $\mathcal{I}'$, ID_i, Hdr)*: with $\mathcal{I}' \subset \mathcal{I}$ and $ID_i \in \mathcal{I}'$. $\mathcal{C}$ gives $\mathcal{A}$ the output of Decrypt(*param*, $\mathcal{I}'$, ID_i, d_i, Hdr).

Challenge: When $\mathcal{A}$ decides that phase 1 is over, $\mathcal{C}$ runs $Encrypt(param, \mathcal{I})$ to obtain (Hdr^*, K). $\mathcal{C}$ randomly selects $b \xleftarrow{\$} \{0, 1\}$, sets $K_b = K$ and sets $K_{1-b} = R$ where R is a random value in the symmetric key space. $\mathcal{C}$ sends (Hdr^*, K_0, K_1) to $\mathcal{A}$.
Query Phase 2: $\mathcal{A}$ continues to adaptively issue queries $q_{s_0+1}...q_s$ where q_i is one of the following:

- *Extraction query (ID_i)*: as in phase1.
- *Decryption query (param, $\mathcal{I}'$, ID_i, Hdr)*: as in phase 1 but with the constraint that $Hdr \neq Hdr^*$. The challenger responds as in phase 1.

Guess: $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ and wins if $b' = b$.

Let us denote by t the total number of extraction queries and q_D the total number of decryption queries. We have $s_0 \leq t + qD$.

An algorithm $\mathcal{A}$ has the following advantage in winning the game: $Adv_{\text{IBBE},\mathcal{A}(t,n,q_D)} = |2 \cdot Pr[b' = b] - 1|$.

Definition 2. *An IBBE scheme is said to be (t, n, q_D)-IND-sID-CCA secure if for all probabilistic polynomial-time adversaries $\mathcal{A}$, its advantage $Adv_{IBBE,\mathcal{A}(t,n,q_D)}$ in winning the game is negligible.*

In section 5.1 we prove that SIBBE is insecure against selective-ID chosen-ciphertext attacks by exhibiting an attacker that wins the IND-sID-CCA game with advantage equal to 1. We however prove in section 5.3 that the scheme is selective-ID secure against chosen-plaintext attacks (IND-sID-CPA).
IND-sID-CPA: the security against chosen-plaintext attacks for an IBBE scheme is defined using the same game between $\mathcal{C}$ and $\mathcal{A}$ except that $\mathcal{A}$ cannot issue decryption queries.

Definition 3. *An Identity-Based Broadcast encryption scheme IBBE is said to be (t, n)-IND-sID-CPA secure if it is $(t, n, 0)$-IND-sID-CCA for all probabilistic polynomial-time algorithms $\mathcal{A}$.*

SIBBE is designed to provide anonymity between policy groups, which informally means that a node i in the network should not be able to tell if a node j belongs to a policy group p if i and j are not in the same group. Formally, the security definition associated with the scheme is anon-IND-sID-CPA [2] and is defined with the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:
Init: The adversary $\mathcal{A}$ outputs two sets of identities
$\mathcal{I}_0 = \{IDi_1, ..., IDi_s\}$ and $\mathcal{I}_1 = \{IDj_1, ..., IDj_s\}$ such that $|\mathcal{I}_0| = |\mathcal{I}_1|$ and we can have $\mathcal{I}_0 \cap \mathcal{I}_1 \neq \emptyset$.
Setup: The challenger $\mathcal{C}$ runs $Setup(\lambda, n)$ to obtain the public parameters *param* and the master secret key MSK. He gives $\mathcal{A}$ the public parameters *param*.
Query Phase 1: $\mathcal{A}$ adaptively issues extraction queries as follows:
Extraction query (ID_i): with the constraint that $ID_i \notin |\mathcal{I}_0 \cup \mathcal{I}_1|$. $\mathcal{C}$ runs $Extract(MSK, ID_i)$ and forwards the resulting private key d_i to $\mathcal{A}$.
Challenge: When $\mathcal{A}$ decides that phase 1 is over, $\mathcal{C}$ randomly selects $b \leftarrow \{0, 1\}$, sets $\mathcal{I}^* = \mathcal{I}_b$ and then runs *Encrypt* to obtain $(Hdr^*, K) = Encrypt(param, \mathcal{I}^*)$. $\mathcal{C}$ sends (Hdr^*, K) to $\mathcal{A}$.

Query Phase 2: $\mathcal{A}$ continues to adaptively issue extraction queries as in phase1.
Guess: $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$ and wins if $b' = b$.
An algorithm $\mathcal{A}$ has the following advantage in winning the game: $Adv_{\text{anon-IBBE},\mathcal{A}(t,n)} = |2 \cdot Pr[b' = b] - 1|$.

Definition 4. *An IBBE scheme is said to be Anon-IND-sID-CPA secure if for all probabilistic polynomial-time adversaries $\mathcal{A}$, its advantage $Adv_{anon\text{-}IBBE,\mathcal{A}(t,n)}$ in winning the game is negligible.*

We will build in section 5.2 a polynomial-time algorithm that wins the anon-IND-sID-CPA game against SIBBE with advantage equal to 1.

3 ESIoT protocol and the SIBBE primitive

3.1 ESIoT Protocol

In ESIoT, devices performing a similar task are grouped into policy groups, and communicate using a group *policy key* broadcasted by a central server. The commissioner in charge of managing the group is authenticated by every node during a policy update phase. At each update phase, the central server first uses the encryption function of SIBBE presented in section 3.2 to encrypt a symmetric policy key K_p where the corresponding ciphertext is $CT = (CT1, CT2, CT3)$ for the policy group. It then generates a policy packet (P) which is the encryption using K_p of a random nonce, the policy description (DR), and the assigned commissioner ID. The central server then signs the concatenation of P and CT and propagates P, CT, and the signature SIG to all nodes via multicast. Each node upon receiving the policy update message (P, CT, SIG), verifies the signature SIG generated by the central server, and then proceeds to decrypt the ciphertext CT in order to retrieve K_p using the SIBBE decryption function and their private decryption key. They then decrypt the policy packet P in order to recover the commissioner ID and the policy description DR.

3.2 SIBBE Primitive at the Heart of ESIoT

The main cryptographic primitive in ESIoT is Secure Identity-Based Broadcast Encryption (SIBBE). SIBBE is at the heart of the group key distribution phase, and is used every time there is an update in the network such as node addition, node revocation, or policy information update. SIBBE comprises

the following algorithms:
Setup(λ, n): Takes as input a security parameter λ and a maximal number of nodes n.
– Generates a bilinear pairing $B = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e(.,.))$, with p a prime number and $g \in \mathbb{G}_1$, $h \in \mathbb{G}_2$, $\gamma \in \mathbb{Z}_p$;
– chooses a hash function $H : \{0,1\}^* \longrightarrow \mathbb{Z}_p$;
– outputs the public parameters $param = (g^\gamma, e(g,h), h, h^\gamma)$;
– derives the *master secret key*: $MSK \Leftarrow (g, \gamma)$.
Extract$(param, MSK)$:
– Generates a private key $d_i = g^{\frac{1}{\gamma + H(ID_i)}}$ for each node in $\{1, ..., n\}$ with identity in $\{ID_1, ..., ID_n\}$.
Encryption$(param, \mathcal{I} = \{ID_1, ..., ID_s\}$[2]$)$:
– Generates at random $k \overset{\$}{\leftarrow} \mathbb{Z}_p^*$;
– $K_p \Leftarrow e(g,h)^k \in \mathbb{G}_T$;
– $f \Leftarrow \prod_{j=1, j\neq i}^{s} (\gamma + H(ID_j))$, $m \Leftarrow \prod_{j=1, j\neq i}^{s} H(ID_j)$;
– $CT1 = e(g^{-k\cdot\gamma}, h^{\frac{(f-m)}{\gamma}})$;
– $CT2 = h^{k\cdot(\prod_{i=1}^{s}(\gamma + H(ID_i)))}$; $CT3 = \frac{1}{\prod_{j=1, j\neq i}^{s} H(ID_j)}$.
The broadcast ciphertext CT = (CT1, CT2, CT3).
Decryption$(param, \mathcal{I}, ID_i, d_i,$ **CT**$)$: extracts policy key for node i as follows:
– $K_p \Leftarrow (CT1 \cdot e(d_i, CT2))^{CT3}$.

4 ON THE EFFICIENCY OF SIBBE

4.1 From Constant to Linear-Size Ciphertexts

There are several issues in the assessment of SIBBE's efficiency. The central server generates three ciphertexts $(CT1, CT2, CT3)$ as shown in 3.2. First, we show that the size of the ciphertext is in fact linear in the number of receivers, and thus generates non-negligble overhead during the ciphertext distribution phase. Indeed, each receiver gets $(CT1, CT2, CT3)$ and should be able to decrypt the ciphertext addressed to him using its private decryption key d_i. However, both CT1 and CT3 are dependent on the target node i ($i \in \{1, ..., s\}$ where s is the number of nodes in the target group) in the fol-

[2] Note that we use the following notations to refer to a subset $\mathcal{I}$ of s IDs: $\mathcal{I} = \{ID_1, ..., ID_s\}$ or $\mathcal{I} = \{ID_1^*, ..., ID_s^*\}$. They should be interpreted as $\mathcal{I} = \{ID_{\phi(1)}, ..., ID_{\phi(s)}\}$ where ϕ is a permutation: $\phi : \{1, ..., n\} \rightarrow \{1, ..., n\}$ and n is the maximal number of receivers. For convenience, we will sometime omit the function ϕ.

lowing manner: $CT1_i = e(g^{-k\cdot\gamma}, h^{\frac{(f-m)}{\gamma}})$ where $f = \prod_{j=1,j\neq i}^{s}(\gamma + H(ID_j))$, $m = \prod_{j=1,j\neq i}^{s} H(ID_j)$, and $CT3_i = \frac{1}{\prod_{j=1,j\neq i}^{s} H(ID_j)}$. CT2 is generated once because it is independent of the target node i. Therefore the size of the ciphertext the server generates is linear in the number of nodes.
Second, we show that the decryption cost for each node is linear in the total number of receivers. Indeed on one hand, considering the fact that one of SIBBE's security property is to conceal the structure of each policy group, node i does not know which triplet $(CT1_i, CT2, CT3_i)$ is addressed to him. It must therefore perform $O(s)$ decryptions, and obtain $O(s)$ session keys K_p in order to test which one gives him an intelligible payload data (namely a policy description and commissionner ID) after decryption of the policy packet (see section 3.1). The decryption time for a constrained node is unfeasible, especially considering the cost of pairing operations as demonstrated in section 4.3. On the other hand, we prove in section 5.2 that an attacker can distinguish between two sets of identities thus gaining information on the structure of policy groups (non-anonymity). Consequently, we may consider a constant time decryption for each receiver simply by allowing the server to attach the corresponding node identifier ID_i to each triplet $(CT1_i, CT2, CT3_i)$ $i \in \{1, ..., s\}$ of the ciphertext. However such consideration would not be compatible with the security goals presented in [10] as attackers can capture packets over the wireless network and thus gain clear access to the structure of the group.
Finally, an adversary could perform a simple Denial-of-Service (DoS) attack on the system by constantly replaying previous update protocol sessions (namely P, CT and SIG) in order to trigger costly decryption of broadcast ciphertexts CT on the receiver side. Given our bandwidth limitations, this could easily render the network unavailable to perform any policy task [3].

4.2 Pairings in Practice

The implementation of a pairing is often used as a black-box in practice, however it is important to understand their properties and the different

[3]Note that to prevent such a "replay attack", the authors of ESIoT suggest to use random nonces in policy update sessions. However these nonces are encrypted using the session key K_p (see section 3.1). To retrieve such a nonce and detect a replay attack, a node would have to first decrypt CT to recover K_p. This would therefore not prevent a DoS attack. A better solution to detect a replay attack would be for the nodes to store each valid signature made by the central server. However nodes are constrained devices which do not necessarily have external memory (EEPROM or Flash memory) to store such signatures.

types of pairings in order to determine which ones offer the best security and efficiency trade-off. There are two forms of cryptographic pairings: symmetric (where $\mathbb{G}_2 = \mathbb{G}_1$) and asymmetric (where $\mathbb{G}_2 \neq \mathbb{G}_1$). Galbraith, Paterson and Smart presented in [7] three types of pairings, one falls into the first form and the other two in the second form. In type 1 pairings we have $\mathbb{G}_1 = \mathbb{G}_2$. In type 2 pairings $\mathbb{G}_1 \neq \mathbb{G}_2$ but there is an efficiently computable homomorphism $\phi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$ and no efficiently computable homomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$. In type 3 pairings $\mathbb{G}_1 \neq \mathbb{G}_2$ and there are no efficiently computable homomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$. Type 1 pairings are implemented using *supersingular* curves, and type 2 and type 3 pairings are implemented using ordinary curves. The efficiency of any scheme based on pairings is dependent on the type of pairing used for implementation.

4.3 On the Efficiency of SIBBE in Practice

The implementation of SIBBE uses type 1 pairings for which group elements are points on *supersingular* elliptic curves. For a security level of 128-bit in type 1 pairings, $\mathbb{G}_1$ contains elements of size 3072 bits. Its computational complexity for a given security level is therefore closer to RSA performances than ECC. Additionally, recently there have been various discrete logarithm attacks on type 1 pairing curves [8]. Consequently for better security and efficiency, the good practice is to use type 3 pairings with Barreto-Naehrig curves over a finite field [1]. For a 128-bit security in type 3 pairings, operations in $\mathbb{G}_1$ are done on 256-bit elements, operations in $\mathbb{G}_2$ on 512-bit elements, and operations in $\mathbb{G}_T$ on 3072-bit elements [3].
Furthermore, let us consider the characteristics of the microcontrollers used to provide benchmarks for SIBBE. The node-side component was implemented on microcontrollers with 32 MHz processors, 512 Kb of ROM and 32 Kb of RAM. A node performing a SIBBE decryption will perform in the worst case s pairing operations and s exponentiations in G_T. The cost of one exponentiation in G_T (respectively in $\mathbb{G}_1$) is roughly equivalent to 2/3 (respectively 1/11) of one pairing operation [3]. In [10] we consider a network with groups of 250 devices, during an update phase each node will have to perform around 416 pairing operations ($5/3 \times 250$). Considering the most efficient pairings in practice (type 3), the node will have to store $[((250 \times (3072 + 256)) + 512)/8]/1024 = 101Kb$ of data during each decryption operation, which exceeds the current storage capacity of most IoT devices. Furthermore, in [10] the cost of one pairing operation is estimated to be around 700ms. In our example, each node will take $0.7 \times 416 = 291$s (roughly 5min) to perform one decryption and not 971ms as claimed in [10].

More efficient decryption of $\mathbf{K_p}$. Based on the discussion above, we can argue that exponentiations in $\mathbb{G}_T$ should be avoided when possible. However, in SIBBE the decryption algorithm retrieves the policy key K_p by computing $K_p = (CT1_i \cdot e(d_i, CT2))^{CT3_i}$ where $i \in \{1, ..., s\}$. We could avoid this costly exponentiation in $\mathbb{G}_T$ by replacing it by a far less costly exponentiation (scalar multiplication) in $\mathbb{G}_1$ as follows:
The server computes $CT1'_i = CT1_i^{CT3_i} i \in \{1, ..., s\}$. To decrypt the ciphertext, each node computes $CT1'_i \cdot e(d_i^{CT3_i}, CT2)$ $i \in \{1, ..., s\}$. Given that the server is more computationally powerful, this step reduces the number of operations on the node side and provides a more efficient decryption function.

5 SIBBE'S SECURITY SHORTCOMINGS

5.1 SIBBE is Not selective-ID CCA Secure

In this section, we prove that SIBBE is not selective-ID CCA secure as claimed in [10]. Indeed, we build a polynomial-time adversary that has non-negligible advantage in the IND-sID-CCA game, and prove the following theorem:

Theorem 1. *There exists a polynomial-time algorithm $\mathcal{A}$ in the IND-sID-CCA game with the following advantage:*
$$Adv^{IND-sID-CCA}_{SIBBE,\mathcal{A}(0,n,1)} = 1.$$

Proof 1. ***Init:*** *Adversary $\mathcal{A}$ outputs a set of identities* $\mathcal{I} = \{ID_1, ID_2\}$.
Setup: *The challenger $\mathcal{C}$ runs $Setup(\lambda, n)$ to obtain the public parameters param and the master secret key MSK. He gives param to $\mathcal{A}$.*
Query Phase 1: *$\mathcal{A}$ issues no query and decides that phase 1 is over.*
Challenge: *$\mathcal{C}$ runs $Encrypt_{SIBBE}(param, \mathcal{I})$ to obtain (C^*, K). $\mathcal{C}$ randomly selects $b \xleftarrow{\$} \{0, 1\}$, sets $K_b = K$ and sets $K_{1-b} = R$ where R is a random value in the symmetric key space. $\mathcal{C}$ sends $(C^* = (CT1_i^*, CT2^*, CT3_i^*)_{i \in \mathcal{I}}, K_0, K_1)$ to $\mathcal{A}$.*
Query Phase 2:
– $\mathcal{A}$ issues no extraction query.
– Decryption query: $\mathcal{A}$ deletes $(CT1_2^*, CT3_2^*)$ for node 2 from C^*. $\mathcal{A}$ sets $\mathcal{I}' = \{ID_1\}$ and sends a new ciphertext $C' = (CT1_1^*, CT2^*, CT3_1^*)$ to the decryption oracle. We can verify that $\mathcal{I}' \subset \mathcal{I}$ and $C' \neq C^*$. Therefore $\mathcal{C}$

sends back the result $K' = K_b$ of Decrypt(param, $\mathcal{I}'$, ID_1, d_1, C').
Guess: *$\mathcal{A}$ outputs 0 if $K' = K_0$, and 1 if $K' = K_1$.*
The success probability of the adversary $\mathcal{A}$ in this game is equal to 1, which proves theorem 1. ∎

5.2 SIBBE is Not an Anonymous IBBE

Selective-ID anonymity. Anonymous IBBE schemes were formalised by Barth et al. in [2]. In order to prove that SIBBE is not an anonymous IBBE for selective identities, we build a polynomial-time adversary that wins the anon-IND-sID-CPA game with probability 1.
Init: The adversary $\mathcal{A}$ outputs two sets of identities $\mathcal{I}_0 = \{ID_1, ID_2\}$ and $\mathcal{I}_1 = \{ID_2, ID_3\}$.
Setup: The challenger $\mathcal{C}$ runs $Setup_{\text{SIBBE}}(\lambda, n)$ to obtain the public parameters *param* and the master secret key MSK. He gives *param* to $\mathcal{A}$.
Query phase 1: $\mathcal{A}$ issues no query and decides that phase 1 is over.
Challenge: $\mathcal{C}$ randomly selects $b \xleftarrow{\$} \{0,1\}$, sets $\mathcal{I}^* = \mathcal{I}_b$ and then runs the encryption algorithm to obtain $(C^*, K) = \text{Encrypt}_{\text{SIBBE}}$ (*param*, $\mathcal{I}^*$). $\mathcal{C}$ sends $(C^* = (CT1_i^*, CT2^*, CT3_i^*)_{i\in\mathcal{I}^*}, K)$ to $\mathcal{A}$.
Query Phase 2: $\mathcal{A}$ issues no query.
Guess: Let $CT3^* = \{CT3_{b+1}^*; CT3_{b+2}^*\}$. $\mathcal{A}$ computes the CT3 corresponding to $\mathcal{I}_0$ as follows:
$CT3 = \{CT3_1 = 1/H(ID_2); CT3_2 = 1/H(ID_1)\}$. If $CT3 = CT3^*$ then $\mathcal{A}$ outputs 0, else $\mathcal{A}$ outputs 1.
The adversary wins the game with probability 1 because $CT3^*$ allows to uniquely identify for which set of IDs the ciphertext C^* is intended.
Non-selective-ID anonymity. We prove that even if the adversary has no idea of the IDs of the devices forming the group ($\mathcal{A}$ does not select target identites prior to starting the security game), it can still retrieve all members of the group in $O(n^2)$ multiplications, where n is the maximal size of the set of receivers.
Let the target ciphertext for the set of intended receivers $\mathcal{I} = \{ID_1, ..., IDs\}$ (see Footnote 2) be $CT^* = (CT1_i^*, CT2^*, CT3_i^*)$ where $i \in \{1, ..., s\}$. $\mathcal{A}$ fixes $\{i, j\} \in \{1, ..., s\}, i \neq j$ and computes $CT3_{i,j}^* = \frac{CT3_i^*}{CT3_j^*} = \frac{H(ID_j)}{H(ID_i)}$. By computing for all $n\cdot(n-1) = O(n^2)$ pairs (k, l), where k and $l \in \{1, ..., n\}$, the following value $CT3_{k,l} = \frac{H(ID_l)}{H(ID_k)}$ and testing which one matches $CT3_{i,j}^*$, $\mathcal{A}$ can easily identify ID_i and ID_j. To find the remaining receivers, $\mathcal{A}$ computes for each $k \in \{1, ..., s\}$ and $k \neq i$ and $k \neq j$: $CT3_{i,k}^* = \frac{CT3_i^*}{CT3_k^*} = \frac{H(ID_k)}{H(ID_i)}$. From $CT3_{i,k}^*$, $\mathcal{A}$ can easily retrieve, by exhaustive search, the corresponding ID_k

(in $O(n)$ multiplications). Therefore $\mathcal{A}$ can find all the remaining ID_k (for $k \neq i$ and $k \neq j$) in $O(n \cdot s)$ steps. Overall the complexity to identify all the intended receivers of CT^* is in $O(n^2 + n \cdot s) = O(n^2)$ multiplications. ∎

5.3 SIBBE's Security Analysis

We prove SIBBE to be selective-ID CPA secure under the variant of the GDDHE assumption presented in 2.2, using Shoup's game hoping method [12].

Game 0: We fix a polynomial-time adversary $\mathcal{A}$. Let us define *Game 0* to be the initial attack game against SIBBE (emulated here by the challenger $\mathcal{C}$). Both the adversary and the challenger are given as input n (the maximal size of a set of intended receivers), and t the total number of extraction and random oracle queries that can be issued by the adversary.
Init: $\mathcal{A}$ sends $\mathcal{C}$ the list of target identities $\mathcal{I} = \{ID_1, ..., ID_s\}_{s \leq n}$.
Setup: $\mathcal{C}$ generates (*param*, MSK) $\Leftarrow$ Setup(λ, n), and sends *param* to $\mathcal{A}$.
Query phase 1:
– $\mathcal{A}$ can issue an extraction query q_i as follows: if $ID_i \notin \mathcal{I}$, $\mathcal{C}$ generates $\gamma \xleftarrow{\$} \mathbb{Z}_p^*$, computes
$d_i = g^{\frac{1}{\gamma + H(ID_i)}}$, and sends $\mathcal{A}$ the private key d_i as the result of the extraction query. Else, $\mathcal{C}$ sends back d_i associated with ID_i.
– $\mathcal{A}$ decides when phase 1 is over.
Challenge: $\mathcal{C}$ generates the ciphertext and the session key using $\text{SIBBE}_{\text{Encrypt}(param,\mathcal{I})}$:
– $(CT1_i^*, CT2^*, CT3_i^*)$ = SIBBE encryption of K_p for $i \in \{1, ..., s\}$;
– $\mathcal{C}$ sends $(CT1_i^*, CT2^*, CT3_i^*, K_0, K_1)$ to $\mathcal{A}$.
Query phase 2: Same as phase 1.
Guess: $\mathcal{A}$ outputs $b' \in \{0, 1\}$ and wins the game if $b' = b$.
Let S_0 be the event that $b' = b$ in Game 0. The advantage of the adversary against SIBBE in the selective CPA attack is the following:

$$Adv^{\text{IND-sID-CPA}}_{\text{SIBBE},\mathcal{A}(t,n)} = |2 \cdot Pr[S_0] - 1| \tag{1}$$

Game 1: It is the same game as Game 0 except the public parameters, the private key d_i and the challenge are computed from a random GDDHE instance with $R = e(g_0, h_0)^{k \cdot f(\gamma) \cdot g(\gamma)}$ (see Definition 1) and the $\{x_i\}$'s denote the roots of f and g.
Init: $\mathcal{A}$ sends $\mathcal{C}$ the list of target identities $\mathcal{I} = \{ID_1, ..., ID_s\}_{s \leq n}$.
Setup: $\mathcal{C}$ has access to a random instance of GDDHE with $R = e(g_0, h_0)^{k \cdot f(\gamma) \cdot g(\gamma)}$ (see Definition 1). He then uses this GDDHE instance to simulate the parameters of SIBBE as follows:

– $\mathcal{C}$ sets $g = g_0^{f(\gamma)}$ without computing it and computes $g^\gamma = {g_0}^{\gamma \cdot f(\gamma)}$, $h = {h_0}^{g(\gamma)}$, $e(g,h) = e(g_0,h_0)^{f(\gamma)\cdot g(\gamma)}$.
– $\mathcal{C}$ sends $param = (g^\gamma, e(g,h), h, h^\gamma)$ to $\mathcal{A}$. $\mathcal{C}$ sets MSK $= (g, \gamma)$.
Hash queries: $\mathcal{A}$ can query for the hash of any identity not in $\mathcal{I}$. $\mathcal{C}$ maintains a list $\mathcal{L}_H = \{\{*, x_i, *\}_{i=1}^{s}; \{\mathrm{ID}_i, x_i, *\}_{i=s+1}^{s+n}\}$, and upon receiving a query on ID_i, he replies in the following way:
– if $\mathrm{ID}_i \in \mathcal{L}_H$, send x_i to $\mathcal{A}$;
– otherwise set $H(ID_i) = x_i$ and complete $\mathcal{L}_H$ with $(\mathrm{ID}_i, x_i, *)$, send x_i.
Query phase 1: $\mathcal{A}$ adaptively issues extraction queries, and $\mathcal{C}$ answers in the following way:
– if $\mathcal{A}$ has already issued an extraction query on ID_i, send back $d_i \in \mathcal{L}_H$;
– else if $\mathcal{A}$ has already issued a hash query on ID_i, use the corresponding x_i to compute the associated private key $d_i = {g_0}^{f_i(\gamma)} = {g_0}^{\frac{f(\gamma)}{\gamma + x_i}}$ [4]. Complete $\mathcal{L}_H$ with $(\mathrm{ID}_i, x_i, d_i)$;
– otherwise, $\mathcal{C}$ sets $H(ID_i) = x_i$, computes d_i as above, and completes $\mathcal{L}_H$. $\mathcal{C}$ sends d_i to $\mathcal{A}$.
Challenge: When $\mathcal{A}$ decides phase 1 is over, $\mathcal{C}$ proceeds as follows to generate the ciphertext and K_p for node $i \in \{1, ..., s\}$:
– $K_p \Leftarrow R = e(g_0, h_0)^{k \cdot f(\gamma) \cdot g(\gamma)}$;
– Compute $f \Leftarrow \prod_{j=1, j\neq i}^{s} (\gamma + x_j)$, and $m \Leftarrow \prod_{j=1, j\neq i}^{s} x_j$;
– $\mathrm{CT1}_i^* \Leftarrow e({g_0}^{-k\cdot\gamma\cdot f(\gamma)}, {h_0}^{g(\gamma)\cdot\frac{(f-m)}{\gamma}})$. [5];
– $\mathrm{CT2}^* \Leftarrow {h_0}^{k\cdot f(\gamma)\cdot g(\gamma)}$; $\mathrm{CT3}_i^* \Leftarrow \frac{1}{m_i}$.
We can verify that $CT_i^* = (CT1_i^*, CT2^*, CT3_i^*)$ is a valid ciphertext.
Pick $b \xleftarrow{\$} \{0,1\}$. Set $K_b = K_p$ and $K_{1-b} \xleftarrow{\$} \mathbb{G}_T$.
– Send (CT_i^*, K_0, K_1) to $\mathcal{A}$.
Query phase 2: Same as in phase 1.
Guess: $\mathcal{A}$ outputs $b' \in \{0,1\}$.
Let S_1 be the event that $b' = b$ in Game 1. We have $Pr[S_0] = Pr[S_1]$ because the distributions of the parameters, the private keys and the challenge are the same in the two games.

Game 2: This game is identical to Game 1 except that both K_0 and K_1 are generated at random (i.e. K_b is not equal to $K_p = e(g,h)^k$).
Init, Setup, Query phase 1: Same as in Game 1

[4] We can verify that d_i corresponds to $g^{\frac{1}{\gamma + \mathrm{H(ID)}_i}}$ and is thus a valid private key from $\mathcal{A}$'s point of view.
[5] We can retrieve the first argument of this pairing from the GDDHE instance,and can compute ${h_0}^{g(\gamma)\cdot\frac{(f-m)}{\gamma}}$ using the powers of ${h_0}^{\gamma}$.

Challenge: $\mathcal{C}$ picks $b \xleftarrow{\$} \{0,1\}$, $K_b \xleftarrow{\$} \mathbb{G}_T, K_{1-b} \xleftarrow{\$} \mathbb{G}_T$
– $\mathcal{C}$ generates $(CT1_i^*, CT2^*, CT3_i^*)$ the challenge CT_i^* for node i as in Game 1.
– $\mathcal{C}$ sends $((CT1_i^*, CT2^*, CT3_i^*), K_0, K_1)$ to $\mathcal{A}$.
Query phase 2: Same as in Game 1.
Guess: $\mathcal{A}$ outputs $b' \in \{0,1\}$.
Let S_2 be the event that $b' = b$ in Game 1. Under the GDDHE assumption, $\mathcal{A}$ cannot detect the change (i.e. that K_0 and K_1 are both generated at random). Indeed we can easily construct a GDDHE distinguisher $\mathcal{D}$ that would answer as follows: $\mathcal{D}$ outputs 1 if $b' = b$ and 0 otherwise. Therefore $Pr[\mathcal{D} = 1|K_b$ was generated as $e(g,h)^k]$
$= Pr[S_1]$ and $Pr[\mathcal{D} = 1|K_b$ is generated at random$] = Pr[S_2]$. Thus we have $|Pr[S_2] - Pr[S_1]| = |Pr[\mathcal{D} = 1|K_b$ is generated at
random$] - Pr[\mathcal{D} = 1|K_b$ was generated as $e(g,h)^k]|$
$\leq Adv_{GDDHE(f,g,F,\mathcal{D})} \leq Adv_{GDDHE(f,g,F)}$. Also observe that in Game 2, no information (in a strong information theoretic sense) is given to $\mathcal{A}$ about the bit b (since both K_0 and K_1 are chosen at random and are independent from the key K_p). Therefore we have $Pr[S_2] = \frac{1}{2}$. It results from the above that $(Adv_{SIBBE,\mathcal{A}(t,n)})/2 = |Pr[S_0] - \frac{1}{2}| = |Pr[S_0] - Pr[S_2]| \leq |Pr[S_0] - Pr[S_1]| + |Pr[S_1] - Pr[S_2]|$
$\leq Adv_{GDDHE(f,g,F)}$.
Thereby SIBBE satisfies the IND-sID-CPA requirement, in the Random Oracle Model, under the variant of GDDHE assumption. ∎

6 CONCLUSION

We have exhibited several issues in the recently introduced ESIoT protocol, and more specifically in their Identity-Based Broadcast Encryption scheme SIBBE. The efficiency of the scheme is hindered by important analysis failures in the encryption function, and in the assessment of the decryption cost. We showed that the scheme does not achieve the security properties claimed in [10] by providing attacks against the anonymous property, and the chosen-ciphertext security. We however proved SIBBE secure for the weaker notion of selective ID chosen-plaintext security under a variant of the GDDHE assumption. The previous points are strong arguments against the use of ESIoT to secure IoT networks.

References

[1] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In *Selected Areas in Cryptography, 12th International Workshop, SAC 2005*, 2005.

[2] Adam Barth, Dan Boneh, and Brent Waters. Privacy in encrypted content distribution using private broadcast encryption. In *Financial Cryptography and Data Security, 10th International Conference*, 2006.

[3] Sébastien Canard, Nicolas Desmoulins, Julien Devigne, and Jacques Traoré. On the implementation of a pairing-based cryptographic protocol in a constrained device. In *Pairing-Based Cryptography - Pairing 2012 - 5th International Conference*, 2012.

[4] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In *Advances in Cryptology - Proc of EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques*, 2003.

[5] Cécile Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. In *Advances in Cryptology - Proc of ASIACRYPT 2007, 13th International Conference on the Theory and Application of Cryptology and Information Security*, 2007.

[6] Amos Fiat and Moni Naor. Broadcast encryption. In *Advances in Cryptology - Proc of CRYPTO '93, 13th Annual International Cryptology Conference*, 1993.

[7] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 2008.

[8] Robert Granger, Thorsten Kleinjung, and Jens Zumbrägel. On the discrete logarithm problem in finite fields of fixed characteristic. *IACR Cryptology ePrint Archive*, 2015.

[9] Thread Group. Thread commissioning whitepaper. 2015.

[10] Jun Young Kim, Wen Hu, Dilip Sarkar, and Sanjay Jha. Esiot: enabling secure management of the internet of things. In *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks, WiSec 2017*, 2017.

[11] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology, Proceedings of CRYPTO '84*, 1984.

[12] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. *IACR Cryptology ePrint Archive*, 2004.