Wireless Networks (2023) 29:1017–1029
https://doi.org/10.1007/s11276-019-02084-6

Identity-based and anonymous key agreement protocol for fog computing resistant in the Canetti–Krawczyk security model

Simone Patonico[1,2] · An Braeken[1,2] · Kris Steenhaut[1,2]

Published online: 11 July 2019

Abstract
Fog computing allows to connect the edge of the network, consisting of low cost Internet of Things devices, with high end cloud servers. Fog devices can perform data processing, which can significantly reduce the delay for the application. Moreover, data aggregation can be carried out by fog devices which decrease the bandwidth needed being very important for the wireless part of the communication with the cloud servers. The edge-fog-cloud architecture is currently being rolled out for several applications in the field of connected cars, health care monitoring, etc. In this paper, we propose an identity-based, mutual authenticated key agreement protocol for this fog architecture, in which end device and fog are able to establish a secure communication without leakage of their identities. Only the cloud server is able to control the identities of device and fog. We formally prove that the session keys are also protected in the Canetti–Krawczyk security model, in which adversaries are considered to have access to session state specific information, previous session keys, or long-term private keys. The scheme is very efficient as it only utilises elliptic curve operations and basic symmetric key operations.

Keywords Fog computing · Authentication · Canetti–Krawczyk · ECQV certificates · Session key security · Anonymity

1 Introduction

Fog computing extends the traditional cloud computing features, such as for instance computation, communication, controlling, and storage, to the edge of the network. To this end, a so called fog layer is placed between the end devices and the cloud. The fog layer typically consists of gateways, base stations, routers, etc. Fog devices can be either fixed (e.g. at train terminals, libraries, etc) or mobile if they are put on a moving object. Compared to a cloud server, the fog devices are much closer to the end devices, leading to low bandwidth costs and low energy consumption. Therefore, fog computation enhances the performance of applications which require low latency [1]. A popular application is in the domain of vehicular networks, which is the essential building block to realise intelligent transport systems [2]. In [3], the so called vehicular fog computing (VFC) paradigm is presented. Instead of using the existing solutions such as cellular networks and roadside units, the authors propose to utilise vehicles as infrastructure nodes for communication and computation, enabling aggregation of resources of individual vehicles in order to increase the quality of services and applications. Another popular application domain is in the health sector, where a large number of embedded and wearable devices, monitoring user's health, are used to derive a diagnosis or treatment. These devices are connected to a nearby fog, where the data is further processed, stored and forwarded [4]. In both use cases anonymity of the device to the fog is a very important feature to guarantee privacy. Authors in [5] analyzed privacy issues during data collection, aggregation and mining in fog devices. To guarantee privacy of identity information during data aggregation, they propose to use an anonymous mechanism based on k-anonymity and traffic

✉ Simone Patonico
spatonic@etrovub.be

An Braeken
abraeken@etrovub.be

Kris Steenhaut
ksteenha@etrovub.be

[1] Vrije Universiteit Brussel, Pleinlaan 2, B-1050 Brussel, Belgium

[2] IMEC, Kapeldreef 75, B-3001 Leuven, Belgium

detection techniques. Differential privacy when using machine learning for data processing is achieved adding Laplacian random noise to the output. In [6], the problem of false data injection from compromised IoT devices has been studied. The injection of fake data makes the aggregation results useless with the consequence of considerable waste of network resources in the fog device. The authors propose a hierarchical Bayesian space-time model to predict future sensor data and detect false aggregated data. A strategy based on anti-honeypot attacks in forensics analysis module is proposed in [7] to counteract Distributed Denial of Service (DDoS) attacks. These detection and forensics modules could be included in fog devices and cloud servers to enhance their security features. A recently proposed authentication scheme [8] includes device anonymity by establishing a common shared key between end device, fog and cloud server, where only the cloud server is aware of the identity of the device and responsible for the access control. Schemes in literature where a common key is shared among three users are also called tripartite schemes. As mentioned in [8], there are only a limited number of schemes that fit to this fog architecture, especially when privacy is required. This follows from the fact that in most of the tripartite schemes, the device in the middle is performing the first validity check on the identity. In the case of the fog architecture, the fog represents the device in the middle and thus when privacy is required, these schemes cannot be applied. Besides privacy, the protection against the Canetti–Krawczyk (CK) adversary model is another important security feature that is gaining more and more interest from the scientific community [9, 10]. The CK adversary model was designed to analyze key exchange protocols and the adequacy of the generated session keys. A key exchange protocol is considered secure if, under the allowed adversary actions, the attacker cannot distinguish the value of the key generated by the protocol from a random number. A scheme is said to offer protection of the session keys in the CK security model [11], if it is resistant against an adversary who is able to reveal session state specific information, previously used session keys, or long-term private keys. For instance, it can be caused by bad implementation of the pseudo random number generator [12–14] or by real leakage attacks exploiting power consumption patterns or timing side channels. Moreover, as the operations are running on end devices and fog devices, which are present in publicly available environments and often vulnerable to active attacks, this is a relevant assumption [10]. In the context of secure identity-based tripartite schemes, the CK security model is recently introduced in the scheme of [15], which is designed for mobile distributed computing environments. In this setting, the end device first communicates to the authentication server, which provides the access control and further forwards it to the application server. Consequently, there is no anonymity provided in the scheme. To the best of our knowledge, an identity-based tripartite scheme, that offers at the same time anonymity and protection against a CK adversary do not yet exist. Due to the importance of privacy in current society and the presence of very strong cyber attack threats, it is very important to combine both features. Therefore, we will present in this paper a scheme solving this issue. The scheme will be proposed as an application in the context of a fog architecture. Applying minimal changes, the proposed scheme can be easily transformed to a solution viable for mobile distributed computing environments, comparable to the one in [15]. What is more, our proposed scheme does not need computationally-intensive pairing operations like [8, 15]. Instead, it utilises only elliptic curve multiplications and additions, hash functions, and symmetric cryptographic operations. Thanks to the construction of the key material, it also becomes possible to construct pairwise secure keys among each of the two involved parties without additional communication. In particular, a common secret key between the end device and the fog device enables protection against an honest but curious cloud server. Similarly, a common secret key between the end device and the central server ensures protection against an honest and curious fog device. In the setting of an honest and curious entity, we assume that this entity is honest in the sense that it will execute all the required actions, but it might be curious and collect the data for other purposes like for instance selling to third parties. The scenario of an honest and curious central server is often considered in smart grid communication [16]. To summarize, the contributions of the paper are the following.

- We present the first identity-based and anonymous key agreement protocol, applicable in a fog computing setting, which offers session key protection in the Canetti–Krawczyk security model.
- We provide a formal proof in the random oracle model to show the security strength of the scheme.
- We compare the efficiency of the scheme with other related tripartite schemes in literature estimating the type and number of operations that the corresponding security algorithms need to perform.

The paper is organised as follows. First, we give an overview of the related work and we deal with preliminaries. Second, the proposed scheme is described and a formal proof of the security in the CK model is given, together with an analysis of several attacks. Then, we analyse the computational complexity and the communication cost of the security algorithm and we conclude the paper.

2 Related work

Many mutual identity-based authentication schemes have been proposed in literature. The main focus has been on client server-based authentication in which the client represents the end device that is more resource-restricted than the server. When the client device requires user interaction, many 2-factor and 3-factor authentication schemes exist in literature. An example of a scheme offering mutual authentication with anonymity and untraceability using solely symmetric key-based operations can be found in [17]. Also, the consideration of an honest but curious Trusted Third Party (TTP) has been taken into account in [18, 19]. In [18], the public key operations are based on the elliptic curve theory, whereas in [19] chaos-based operations are used. For client server authentication schemes with a client representing an autonomous device, there are only a limited number of mutual identity-based authentication schemes [9, 10, 20–22]. These schemes differ in several points. For instance, regarding the proposed architecture, in [21, 22], an active TTP is required during the key agreement phase, which is not the case in the other proposals. Only a limited number of these schemes allow the anonymity of the client [9, 10, 20] and even less schemes are resistant in the CK security model [9, 10]. Moreover, this additional security restriction has only been recently introduced. In the context of the so-called tripartite schemes, where three entities need to agree on a common key, we can also distinguish several identity-based mutual authentication schemes. Some of the schemes are based on symmetric key mechanisms, using a pre-shared common key [23–26]. In particular, [23, 24] study the minimum amount of communication rounds and messages needed to establish mutual authentication among three different parties, taking into account different assumptions. The disadvantage in these schemes is that the session key is only constructed by the authentication server and the other two entities do not participate in its construction, making these schemes vulnerable for key control resilience attacks [27]. In order to establish anonymity, as noticed in [28], public key-based operations need to be used. In [8], an example of a key agreement scheme for a fog-driven healthcare application is proposed in which anonymity of the end device is obtained. The scheme is an improvement of [29] in which the derived key was static and thus not able to establish past forward security. However, we see several shortcomings in [8]. First, the scheme is limited to devices possessing a smart card-based entry and the registration phase requires the presence of a secure channel between the user and the trusted cloud service provider. Second, CK security for the session keys has not been considered. Third, the scheme is not offering protection against an honest but curious central server. Finally, computationally-intensive pairing operations are involved in the scheme. On the other hand, in [15], a secure identity-based tripartite scheme resistant in the CK security model is given, which is designed for mobile distributed computing environments. However, this scheme does not provide anonymity to outsiders and also consists of a pairing operation at device side. In addition, it is also not able to compute pairwise keys using the available key material at the end of the protocol.

3 Preliminaries

We first provide some background on Elliptic Curve Cryptography (ECC). Next, the CK security model is further elaborated. We also describe in detail the Elliptic Curve Qu-Vanstone (ECQV) certificate scheme as it is an important building block in the registration phase of our proposed scheme.

3.1 Elliptic curve cryptography

Elliptic Curve Cryptography (ECC) [30] offers lightweight public key cryptography (PKC) solutions. For instance, corresponding with an 80-bit security parameter, a field size of 160 bits for ECC is sufficient, whereas RSA-based solutions require 1024 bits. ECC is based on the algebraic structure of elliptic curves (ECs) over finite fields. The curve in the finite field F_p is denoted by $E_{p(a,b)}$, whereas the base point generator of prime order q is denoted by G. All points on $E_{p(a,b)}$, together with the infinite point form an additive group. In [31, 32] standardised curve parameters are described. The product $R = rG = (R_x, R_y)$ with $r \in F_q$ and $R_x, R_y \in F_p$ results in a point of the EC and represents an EC multiplication. When we send an EC point, it suffices to send its x coordinate, together with one sign bit, cf. the SEC1-based encoding [33]. The scheme relies on two computational hard problems.

- The Elliptic Curve Discrete Logarithm Problem (ECDLP). This problem states that given two points R and Q of an additive group N, generated by an elliptic curve (EC) of order q, it is computationally hard for any polynomial-time bounded algorithm to determine a parameter $x \in Z_q^*$, such that $Q = xR$.
- The Elliptic Curve Diffie Hellman Problem (ECDHP). Given two points $R = xG$, $Q = yG$ of an additive group N, generated by an EC of order q with two unknown parameters $x, y \in Z_q^*$, it is computationally hard for any polynomial-time bounded algorithm to determine the EC point xyG.

3.2 Threat model

We consider as in [9] the CK-adversary model, as proposed in [11]. In this security model, the adversary can not only eavesdrop on the channel or actively manipulate (insert, change, reply) the transmitted messages, but can also reveal session state-specific information, session keys, or long-term private keys. The session state-specific information is defined as the local state of the session and its subroutines, excluding the ones where direct access to the long term secret information is performed.

3.3 Elliptic curve Qu-Vanstone certificates

The Elliptic Curve Qu-Vanstone (ECQV) certificate scheme [34, 35] is a very efficient mechanism to construct a key pair (private and public keys) together with a certificate for an entity in the scheme without the need of a secure channel between the TTP and the entity to share material for the generation of its secret private key. Consequently, the TTP is also not able to derive the private key of the entity and so there are no key escrow problems. Its security has been formally proven in [36]. The ECQV scheme, which is shown in Fig. 1, works as follows for an entity A requesting the generation of its secret key pair and corresponding certificate with the TTP. Consider the curve $E_{p(a,b)}$ in Z_p with generator point G of order q. Denote the private and public key of the TTP by (k, P_{TTP}) with $P_{TTP} = kG$. Define the hash function $H_0 : \{0,1\}^* \rightarrow Z_p^*$ and the concatenation operation between two parameters p_1 and p_2 as $p_1 \| p_2$. First the entity A with identity ID_A chooses a random value $r_A \in Z_p^*$ and computes $R_A = r_A G$. The message ID_A, R_A is sent to the TTP. Here, the TTP also selects a random value $r_T \in_R Z_p^*$ and computes $R_T = r_T G$. Next, it computes

$$\begin{aligned} cert_A &= R_A + R_T \\ r &= H_0(cert_A \| ID_A) r_T + k \end{aligned}$$

The values $(cert_A, r)$ are sent to A over a public channel. Using these values, A now computes its private key

$$d_A = H_0(cert_A \| ID_A) r_A + r$$

It accepts the registration if its public key $P_A = d_A G$ satisfies the following equality

$$P_A = H_0(cert_A \| ID_A) cert_A + P_{TTP} \quad (1)$$

Consequently, given ID_A, $cert_A$ and, of course, the public key of the TTP denoted P_{TTP}, any other entity is able to construct the corresponding public key of A by means of Eq. 1. Thanks to the certificate, the other entity is assured of the relation between identity and public key.

4 Proposed solution

The proposed scheme consists of three main phases, which allow the construction of a common shared key between all the entities. Besides this key, each entity has security material in common with just another entity of the system that can be used to build a secure channel between these entities.

4.1 Setup phase

In this phase, the TTP selects the EC $E_{p(a,b)}$ in Z_p with generator point G of order q. It determines seven hash functions $H_0 : \{0,1\}^* \rightarrow Z_q^*$, $H_1 : \{0,1\}^* \rightarrow Z_q^*$,

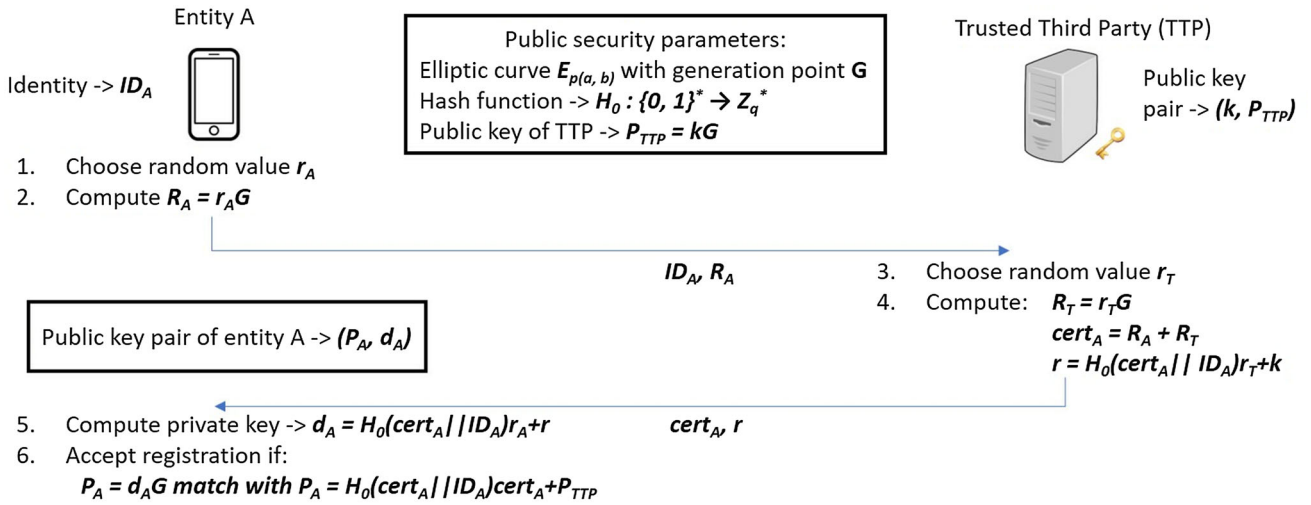


Fig. 1 The ECQV registration phase

$H_2 : \{0,1\}^* \rightarrow Z_q^*$, $H_3 : \{0,1\}^* \rightarrow Z_q^*$, $H_4 : Z_q^* \rightarrow Z_q^*$, $H_5 : \{0,1\}^* \rightarrow Z_q^*$, and $H_6 : \{0,1\}^* \rightarrow Z_q^*$. Also, a symmetric key encryption algorithm is chosen to encrypt a message M into the ciphertext C using the to-be-settled secret shared key SK, $C = E_{SK}(M)$, together with the corresponding decryption algorithm, $M = D_{SK}(C)$. A random value k is set as the private key of the TTP. The corresponding public key P_{TTP} is computed by $P_{TTP} = kG$. This public key P_{TTP}, together with the public parameters $\{E_{p(a,b)}, G, H_0, H_1, H_2, H_3, H_4, H_5, H_6, E_{SK}(), D_{SK}()\}$ are published.

4.2 Registration phase

The registration phase for sensor devices (SD), fog devices (FD) and central servers (CS) are similar and follow the ECQV certificate scheme, as explained above. As a result, each entity U is storing the public parameters $\{E_{p(a,b)}, G, H_0, H_1, H_2, H_3, H_4, H_5, H_6, E_{SK}(), D_{SK}(), P_{TTP}\}$, its public key P_u, certificate $cert_u$ and identity ID_u, together with its private key d_u. Note that only the private key needs to be stored in the tamper-resistant part of the memory. As in the other papers ([8, 15]), we assume that the SD and FD have stored the public key P_c of the CS. If not, they need to request the identity and certificate of the CS to compute the corresponding public key, cf. Eq. 1, before the key agreement phase.

4.3 Key agreement phase

In the key agreement phase, the actual symmetric secret key SK shared between SD, FD, and CS is established. We denote the SD by the entity with identity ID_s, key pair (d_s, P_s) and certificate $cert_s$. Similar, the FD is denoted by the entity with identity ID_f, key pair (d_f, P_f) and certificate $cert_f$. Finally, the CS has identity ID_c, key pair (d_c, P_c) and certificate $cert_c$. There are four communication passes in the scheme, leading to five different steps. The main interaction between the SD and FD is shown in Fig. 2. In this figure we also describe the function of each computed parameter.

(1) *Sensor device initialization:* The SD first chooses a random variable r_1 and computes $R_1 = (r_1 + d_s)G$. Next, it computes a common key $K_1 = H_4((r_1 + d_s)P_c)$ with the C in order to derive the ciphertext $C_1 = E_{K_1}(ID_s \| cert_s)$. The value $Q_1 = (r_1 + d_s)P_s$ represents a masked version of the public key of the SD for anonymity reasons. Finally, the hash value $A_1 = H_1(R_1 \| C_1 \| Q_1)$ is computed. The message $M_1 = \{R_1, C_1, Q_1, A_1\}$ is sent to the FD.

(2) *Fog device to central server:* Upon arrival of M_1, the hash value A_1 is checked to ensure the message integrity. If positive, the process continues. The following steps are similar as with the SD. A new random value r_2 is derived in order to compute $R_2 = (r_2 + d_f)G$, $Q_2 = (r_2 + d_f)P_f$, the common key $K_2 = H_4((r_2 + d_f)P_c)$ with the CS, and the ciphertext $C_2 = E_{K_2}(ID_f \| cert_f \| H_4^2(P_{12}))$. The point P_{12} is computed using $h_{11} = H_5(R_1 \| Q_1 \| R_2 \| Q_2)$, $h_{12} = H_5(R_2 \| Q_2 \| R_1 \| Q_1)$ and equals to $P_{12} = (r_2 + d_f + h_{11}(r_2 + d_f)d_f)(R_1 + h_{12}Q_1)$. Note that we send $H_4^2(P_{12}) = H_4(H_4(P_{12}))$ in C_2 as $H_4(P_{12})$ corresponds with a unique shared key between FD and SD. Finally, the hash value $A_2 = H_2(R_1 \| C_1 \| R_2 \| C_2)$ is computed and the message $M_2 = \{R_1, C_1, R_2, C_2, A_2\}$ is sent to the CS.

(3) *Central server to fog device:* First the hash value A_2 is checked in order to guarantee the integrity of the message M_2. Next, the keys $K_1 = H_4(d_c R_1)$, $K_2 = H_4(d_c R_2)$ are derived in order to decrypt C_1, C_2 and to derive the required information to find the public keys P_s, P_f of the SD and the FD respectively using the ECQV mechanism. In addition, $H_4^2(P_{12})$ is found, which will be later used for the construction of the session key SK. Next, a new random value r_3 is derived to compute $R_3 = (r_3 + d_c)G$. Using the hashes $h_{21} = H_5(R_2 \| P_f \| R_3 \| P_c)$, $h_{22} = H_5(R_3 \| P_c \| R_2 \| P_f)$, $h_{31} = H_5(R_1 \| P_s \| R_3 \| P_c)$, $h_{32} = H_5(R_3 \| P_c \| R_1 \| P_s)$, the points $P_{23} = (r_3 + d_c + h_{21}d_c)(R_2 + h_{22}P_f)$ and $P_{13} = (r_3 + d_c + h_{31}d_c)(R_1 + h_{32}P_s)$ can be computed. As a consequence, the SK is defined as $SK = H_6(H_4^2(P_{12}) \| H_4^2(P_{13}) \| H_4^2(P_{23}))$. Note that $H_4(P_{13})$ and $H_4(P_{23})$ represent the common shared key between CS on the one hand and the SD and FD respectively on the other hand. In order to share the point P_{23} with the SD and P_{13} with the FD, the CS computes the ciphertext $C_4 = E_{H_4(P_{13})}(H_4^2(P_{23}))$ and $C_3 = E_{H_4(P_{23})}(H_4^2(P_{13}))$ respectively. Finally, the hash value $A_3 = H_3(R_1 \| R_2 \| R_3 \| SK)$ is computed and the message $M_3 = \{R_3, C_3, C_4, A_3\}$ is sent to the FD.

(4) *Fog device to sensor device:* At the FD, first the hash values h_{21}, h_{22} are computed in order to derive the point $P_{23} = (r_2 + d_f + h_{22}d_f)(R_3 + h_{21}P_c)$. This point is used to decrypt C_3 and to find $H_4^2(P_{13})$. As a consequence, the FD has all the required information to derive the SK. Next, it checks the validity of A_3 and if positive, the message $M_4 = \{R_2, Q_2, R_3, C_4, A_3\}$ is sent to SD.

(5) *Sensor device termination:* When the message arrives, the SD first computes the hashes h_{11}, h_{12},

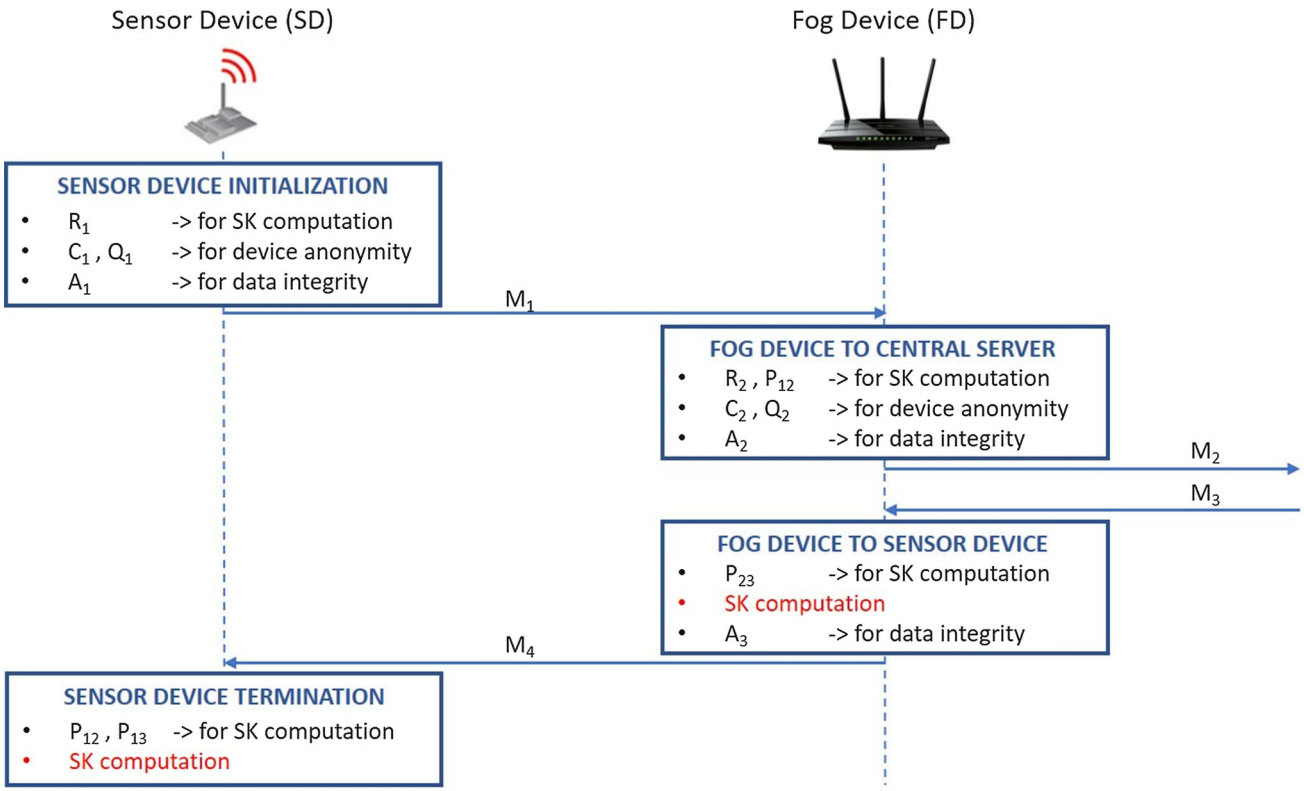


Fig. 2 The interaction between the SD and FD during the key agreement phase

h_{31}, h_{32} as defined above, in order to derive the points $P_{12} = (r_1 + d_s + h_{12}(r_1 + d_s)d_s)(R_2 + h_{11}Q_2)$ and $P_{13} = (r_1 + d_s + h_{32}d_s)(R_3 + h_{31}P_c)$. Using the last point, C_4 can be decrypted in order to derive $H_4^2(P_{23}) = D_{H_4(P_{13})}(C_4)$. Consequently, also SK can be computed and A_3 verified.

The described steps are represented in Fig. 3, where the reader can find all the details of the proposed key agreement algorithm.

5 Security analysis

First, we provide a formal proof of the security strength of our protocol. Then, we analyze some of the most used attacks and show that our key agreement scheme offers protection against such attacks.

5.1 Formal proof of security

We now show that our key agreement scheme is secure under the CK adversary model [11] in the random oracle model, following the method of [9, 37]. We focus on the actual key agreement and not on the registration phase, as we consider the TTP to be honest but curious entity. Note that this assumption is strong enough since the TTP is not able to derive the secret keys due to the usage of the ECQV security mechanism. The participants U in our scheme are the SD, FD, CS and a random oracle O, i.e. $U = \{SD, FD, CS, O\}$. Taking into account the CK adversary model, we assume that the attacker can run the following queries.

- Hash queries $H_i(m)$ with $i \in \{0, 1, 2, 3, 4, 5, 6\}$. If m already exists in the list L_{H_i}, the value $H_i(m)$ will be returned. Otherwise, a random value will be generated, added to the list L_{H_i}, and returned.
- Send queries. These queries simulate active attacks, in which the adversary is able to modify the transmitted messages. The random oracle O, which simulates a device of the system, replies to the attacker with the corresponding message of the key agreement protocol. Since there are four communication passes, five different send queries need to be defined.
  - *Send(START,SD).* Upon receiving this query, the random oracle chooses a random variable r_1 and computes $R_1 = (r_1 + d_s)G$. Next, $K_1 = H_4((r_1 + d_s)P_c)$ is derived to construct $C_1 = E_{K_1}(ID_s \| cert_s)$. Then, $Q_1 = H_4((r_1 + d_s)P_s)$ is computed. Finally, the hash value $A_1 = H_1(R_1 \| C_1 \| Q_1)$ is found. The output message $M_1 = \{R_1, C_1, Q_1, A_1\}$ is sent to the adversary.

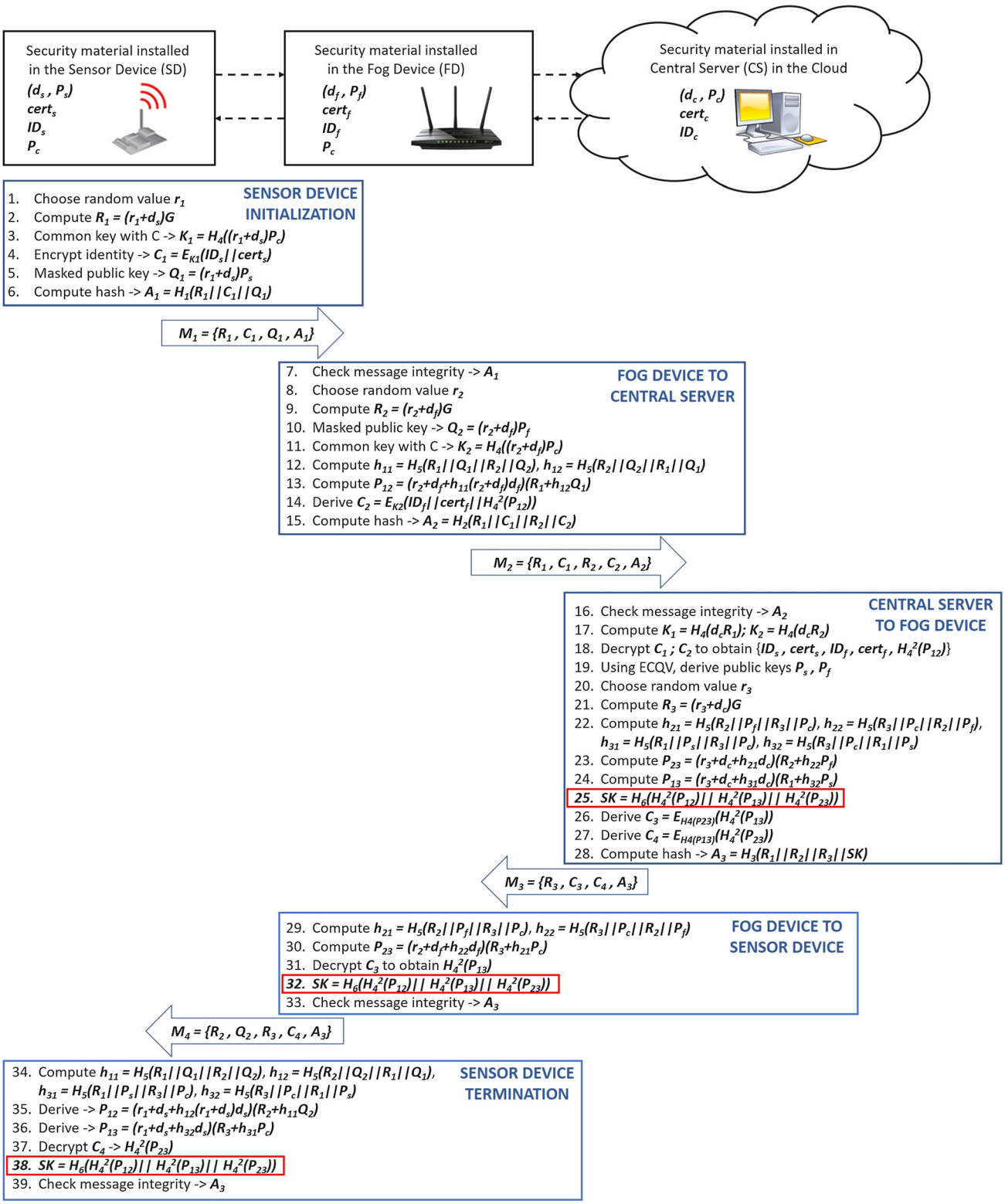


Fig. 3 The key agreement phase

– *Send* (M_1, FD). First, A_1 is checked and, if positive, a random value r_2 is chosen to compute $R_2 = (r_2 + d_f)G$, $Q_2 = H_4((r_2 + d_f)P_f$, and $K_2 = H_4((r_2 + d_f)P_c)$. Then, $h_{11} = H_5(R_1\|Q_1\|R_2\|Q_2)$ and $h_{12} = H_5(R_2\|Q_2\|R_1\|Q_1)$ are computed to derive the point $P_{12} = (r_2 + d_f + h_{11}(r_2 + d_f)\ d_f)$

$(R_1 + h_{12}Q_1)$. Next, using K_2, the ciphertext $C_2 = E_{K_2}(ID_f \| cert_f \| H_4^2(P_{12}))$ is constructed. Finally, the random oracle computes the hash value $A_2 = H_2(R_1 \| C_1 \| R_2 \| C_2)$ and the message $M_2 = \{R_1, C_1, R_2, C_2, A_2\}$ is the output of the query, which is received by the adversary.

- *Send* (M_2, CS). First, A_2 is checked and if positive $K_1 = H_4(d_c R_1)$, $K_2 = H_4(d_c R_2)$ are constructed in order to decrypt C_1, C_2 and to derive $ID_s \| cert_s$ and $ID_f \| cert_f \| H_4^2(P_{12})$ respectively. Second, $P_s = H_0(ID_s \| cert_s) cert_s + P_{TTP}$ and $P_f = H_0(ID_f \| cert_f) cert_f + P_{TTP}$ are found. Third, a random value r_3 is chosen to compute $R_3 = (r_3 + d_c)G$. Fourth, the hashes $h_{21} = H_5(R_2 \| P_f \| R_3 \| P_c)$, $h_{22} = H_5(R_3 \| P_c \| R_2 \| P_f)$, $h_{31} = H_5(R_1 \| P_s \| R_3 \| P_c)$, $h_{32} = H_5(R_3 \| P_c \| R_1 \| P_s)$ are computed to find the points $P_{13} = (r_3 + d_c + h_{21} d_c)(R_1 + h_{22} P_s)$ and $P_{23} = (r_3 + d_c + h_{31} d_c)(R_2 + h_{32} P_f)$. Fifth, $SK = H_6(H_4^2(P_{12}) \| H_4^2(P_{13}) \| H_4^2(P_{23}))$ is computed. Next, $C_4 = E_{H_4(P_{13})}(H_4^2((P_{23}))$ and $C_3 = E_{H_4(P_{23})}(H_4^2(P_{13}))$ are derived. Finally, the hash value $A_3 = H_3(R_1 \| R_2 \| R_3 \| SK)$ is computed and the message $M_3 = \{R_3, C_3, C_4, A_3\}$ is sent to the adversary.
- *Send* (M_3, FD). The random oracle computes $h_{21} = H_5(R_2 \| P_f \| R_3 \| P_c)$ and $h_{22} = H_5(R_3 \| P_c \| R_2 \| P_f)$ to find the point $P_{23} = (r_2 + d_f + h_{22} d_f)(R_3 + h_{21} P_c)$ and thus also $H_4^2(P_{13}) = D_{H_4(P_{23})}(C_3)$. Then, $SK = H_6(H_4^2(P_{12}) \| H_4^2(P_{13}) \| H_4^2(P_{23}))$ is computed and A_3 verified. If positive, the random oracle sends message $M_4 = \{R_2, Q_2, R_3, C_4, A_3\}$ to the adversary as the output of the query.
- *Send* (M_4, SD). First the four hashes $h_{11} = H_5(R_1 \| Q_1 \| R_2 \| Q_2)$, $h_{12} = H_5(R_2 \| Q_2 \| R_1 \| Q_1)$, $h_{31} = H_5(R_1 \| P_s \| R_3 \| P_c)$, $h_{32} = H_5(R_3 \| P_c \| R_1 \| P_s)$ are computed to find the points $P_{12} = (r_1 + d_s + h_{12} d_s)(R_2 + h_{11} Q_2)$ and $P_{13} = (r_1 + d_s + h_{32} d_s)(R_3 + h_{31} P_c)$. Consequently, $H_4^2(P_{23}) = D_{H_4(P_{13})}(C_4)$ is derived and SK is computed as $SK = H_6(H_4^2(P_{12}) \| H_4^2(P_{13}) \| H_4^2(P_{23}))$. If the check on A_3 is unsuccessful, the query is aborted.

- Execute queries. These queries simulate the passive attacks, in which the adversary can only eavesdrop onto the channel and is able to collect the transmitted messages. We can distinguish four different execute queries resulting from the first four send queries defined above, where a message has been transmitted over the public channel.
- Session specific state reveal queries (SSReveal). According to the CK adversary model, the attacker is able to retrieve session specific state information, derived by the *SD*, *FD* and *CS* respectively. Note that no values in which long term private keys are involved, can be revealed in this query.
  - *SSReveal*(*SD*). The output of this query results in r_1, R_1, C_1, Q_1, A_1, h_{11}, h_{12}, h_{31}, h_{32}, R_3, Q_2, R_2, C_4, A_3.
  - *SSReveal*(*FD*). The output of this query results in R_1, C_1, Q_1, A_1, r_2, R_2, C_2, Q_2, A_2, h_{11}, h_{12}, h_{21}, h_{22}, R_3, C_3, C_4, A_3.
  - *SSReveal*(CS). The output of this query results in R_1, C_1, Q_1, R_2, C_2, Q_2, A_2, r_3, R_3, h_{31}, h_{32}, h_{21}, h_{22}, C_3, C_4, A_3.
- Corrupt queries. These queries give the private key of the entity as result. Note that only Corrupt(*SD*), Corrupt(*FD*) and Corrupt(*CS*) exist and no corrupt queries with regards to the TTP. They are included to prove the perfect forward security of the scheme.
- Session key reveal query (SKReveal). In this query, the established symmetric *SK* between *SD*, *FD*, *CS* is returned in case it has been successfully generated.
- Test query. In this query, the random oracle returns to the adversary either the established *SK* or a random value having the same length, dependent on the output $c = 1$ or $c = 0$ respectively of a flipped coin c. The adversary can use this query only once. Note that the test query cannot be issued when SKReveal or corrupt queries have been executed.

In order to prove the semantic security of the scheme, we consider the following two definitions.

- The *SD*, *FD* and *CS* are partners if they are able to successfully derive an authenticated common shared key *SK*. The common shared key *SK* cannot be computed by other entities.
- The established shared secret key is said to be fresh if the *SK* has been established without exposure to SKReveal queries by the adversary or Corrupt queries of *SD*, *FD* and *CS*.

The final goal of the adversary $\mathcal{A}$ is to distinguish the difference between a real secret session key or a random value, i.e., to predict successfully the output of the test query. If $Pr(succ)$ denotes the probability that the adversary succeeds in its mission, the advantage of the adversary in breaking the semantic security of the proposed scheme equals to $Adv(\mathcal{A}) = |2Pr[succ] - 1|$. Consequently, our scheme offers semantic security under the CK adversary and random oracle model if the advantage for $\mathcal{A}$ winning the game satisfies $Adv(\mathcal{A}) \leq \epsilon$, for any sufficiently small $\epsilon \geq 0$. The difference lemma [38] is used to prove the statement.

Lemma 1 (Difference Lemma) Let E_1, E_2 be the events of winning game 1 and game 2. Denote an error event by E,

such that $E_1|\neg E$ occurs if and only if $E_2|\neg E$. Then, $|Pr[E_1] - Pr[E_2]| \leq Pr[E]$.

Theorem 1 Let $\mathcal{A}$ be a polynomial time adversary against the semantic security, which makes a maximum of q_s Send queries, q_e Execute queries and q_h Hash queries. The advantage of $\mathcal{A}$ is bounded by $Adv(\mathcal{A}) \leq \frac{O(q_s+q_e)^2}{2q} + \frac{O(q_h)^2}{2q} + \frac{O(q_s)^2}{2^l} + O(q_h T)$, with T the time to solve the ECDH problem.

Proof We proof the theorem by means of game hopping [38]. An attacker's success probability only increases by a negligible amount when moving between the games, as a consequence of Lemma 1. There are five games $\{GM_0, GM_1, GM_2, GM_3, GM_4\}$ to be defined. Denote by $succ_i$ the event that $\mathcal{A}$ wins the game GM_i, with $0 \leq i \leq 4$.

- Game GM_0. This is the real game, as defined in the semantic security framework. From the definition, we have that

$$Adv(\mathcal{A}) = |2Pr[succ_0] - 1|. \tag{2}$$

- Game GM_1. In this game, the oracles for the different queries are simulated and the resulting outputs of the queries are stored in the lists. In the random oracle model, it holds that

$$Pr[succ_1] = Pr[succ_0]. \tag{3}$$

- Game GM_2. In GM_2, all oracles are simulated, avoiding collisions in the output of the hash functions and the selection of random values r_1, r_2, r_3 among the different sessions. The probabilities of collisions between the outputs of the hash functions (E_1) and between the random values (E_2) are respectively

$$Pr[E_1] \leq \frac{O(q_h)^2}{2q} Pr[E_2] \leq \frac{O(q_s + q_e)^2}{2q} \tag{4}$$

Consequently, due to the difference lemma, it holds that

$$|Pr[succ_2] - Pr[succ_1]| \leq \frac{O(q_s + q_e)^2}{2q} + \frac{O(q_h)^2}{2q}. \tag{5}$$

- Game GM_3. In this game, the adversary $\mathcal{A}$ is able to find the hash value A_3 without input of the random oracle Send queries. In this case, the scheme is simply stopped. Consequently, GM_2 and GM_3 are indistinguishable, except when FD or SD rejects A_3. Thus, following the Difference Lemma, it holds that

$$|Pr[succ_3] - Pr[succ_2]| \leq \frac{O(q_s)^2}{2^l}. \tag{6}$$

- Game GM_4. In this game, we consider the CK adversary model and assume that either the session state variables or the long term secret variables are revealed at each of the involved participants. The goal of the adversary is to find the SK by performing Execute and Hash queries, with eight possible combinations of SSReveal and Corrupt queries. The session key is constructed by means of three EC points, P_{12}, P_{13}, P_{23}. Due to the definition of these points, P_{ij} (with $i \neq j$ and $i,j \in \{1,2,3\}$) can only be constructed by means of the knowledge of both the session information (random variable) and the private key of the involved entity i as both are independently involved in the definition. The knowledge of both these secrets is in contradiction with the CK security model. Only in the case of a Corrupt(CS) query, the key K_2 can be revealed and thus also $H_4^2(P_{12})$. As this is only a part of the SK, it is still insufficient to reveal the complete SK as P_{13}, P_{23} can still not be revealed with only the knowledge of d_c. Moreover, in the same setting, an impersonation attack on SD or FD is not possible, due to the usage of ECQV certificates. Consequently, the difference between GM_2 and GM_3 is negligible as long as the probabilities to solve the ECDH problem and to perform a successful hash query are small. Denote T as the time to solve the ECDH problem, then

$$|Pr[succ_4] - Pr[succ_3]| \leq O(q_h T). \tag{7}$$

Consequently, applying Lemma 1 on the games GM_0, GM_1, GM_2, GM_3 and GM_4, taking into account equations 2,3,5,6, results in the final proof of the theorem.

□

5.2 Attack analysis

We demonstrate that our authenticated key agreement protocol is secure against several attacks which can endanger the privacy of users and the confidentiality of the exchanged data.

- *User anonymity and untraceability* An adversary, which can be a malicious sensor device or fog device, cannot retrieve the identities of the other devices in the system even if it intercepts all the messages that are exchanged during the key agreement phase. Indeed, the identities are encrypted with the keys K_1, K_2 and only the central server is able to compute them using its private key d_c. Moreover, these keys change at each session because they depend on the random numbers r_1, r_2, r_3.
- *Perfect forward privacy* Even if the attacker is able to steal the long term private keys of the entities of the system, the previously generated common secret keys

are not compromised. Indeed, the generation of these session keys also require the random values r_1, r_2, r_3 which change at each session.

- *Man-in-the-middle attack* In this type of attack, the attacker is able to intercept and forge the four exchanged messages in the key exchange protocol. The resistance against this attack follows from the ECQV certificate scheme used in the registration phase. Indeed, the certificate of each entity is created by using the secret random numbers of both entity and TTP. Moreover, the private key of the TTP is used for the construction of the entity's private key. Therefore, the attacker will not be able to compute the private key correspondent to the entity's public key computed by the central server in step 19 of the key agreement phase. Consequently, the attacker cannot compute the same secret key *SK* calculated by the central server.
- *Session key leakage.* The session secrets are generated using both the random numbers and the private keys, hence they change at each session. The leakage of one session key does not compromise the security of the other session keys.
- *Key-compromised impersonation attack.* In this scenario, the attacker corrupts the private key of the sensor device to impersonate the central server and to cheat the sensor device and fog device. Although the attacker can compute the sensor device's public key, it is still not able to derive $H_4^2(P_{12})$ because it needs the central server's private key to decrypt the cipher text C_2. Therefore, the attacker will not be able to compute the common secret *SK*.
- *Key control attack* In the proposed scheme, the common secret *SK* is computed by using all entities' private keys and random numbers. Consequently, if the attacker corrupts one of the entities, it will still not be able to determine the *SK*.

6 Performance analysis

The performance analysis is split into the computation and communication costs. We compare our scheme with the schemes of [8, 15]. Recall that the scheme of [8] does not offer session key security in the CK security model and the scheme of [15] does not provide entity anonymity.

6.1 Computation costs

The computation costs are measured by counting the number of most computationally-intensive operations and taking their corresponding computational time into account. We denote the timing for the bilinear pairing as T_b, the point multiplication T_{mp}, point addition T_{ap}, a symmetric encryption/decryption T_s, a map to point T_H and hash operation T_h. To measure the timings of these operations for the fog device and the central server, we refer to [16]. The authors used a personal computer with a 2.5 GHz CPU and an 8 GB RAM, running Windows 7 for an 80-bit security level. This corresponds to a hash function resulting in a 160 bit output and an EC of order 160, i.e. $q = 160$. According to the NIST reccomendations, an EC of order 256 should be chosen resulting to 128-bit security level. However, we decided to maintain 80-bit security level to perform a fair comparison with [8, 15]. These timings, expressed in microseconds (μs) result in $T_b = 17.001$, $T_{mp} = 0.986$, $T_{ap} = 0.004$, $T_s = 0.001$, $T_H = 14.29$, and $T_h = 0.001$. On the other hand, we have tested the same operations on the constrained Zolertia RE-mote to simulate the sensor device. This platform is endowed with an ARM Cortex-M3 32 MHz clock speed as microcontroller, 512 KB of flash memory and 32 KB of RAM. The Contiki 3.0 operating system offers APIs that implement cryptographic operations. In particular, we used the AES/SHA cryptoprocessor to perform the hash and symmetric encryption/decryption operations and the public key accelerator (pka) engine to carry out the elliptic curve point multiplication and point addition. Unfortunately, Contiki 3.0 does not include any library to execute bilinear pairing and map to point operations for Zolertia RE-mote. In fact, these operations are too complex to be executed in reasonable time in the RE-mote's microcontroller [39]. Therefore, the RE-mote cannot be used in [8, 15] to act as a sensor device. These security schemes need a more powerful device. The computed timings for the Zolertia RE-mote expressed in milliseconds (ms) are $T_{mp} = 342.39$, $T_{ap} = 5.25$, $T_s = 0.12$, $T_h = 0.03$. In Table 1, the number of most computationally-intensive operations and the corresponding timing according to the above defined measurements have been determined for our scheme and the schemes of [8, 15]. As can be concluded from this table, our scheme considerably outperforms the other schemes for all three entities involved. This follows from the fact that our scheme does not involve the computationally-intensive pairing operations.

6.2 Communication costs

For the communication costs, we determine the number of transmitted bits in each of the four messages sent between the different entities of the scheme. Note that we consider, similar to the other schemes in the literature, the 80-bit security level. This corresponds with hash functions giving

Table 1 Comparison of computational complexity

Scheme	Cost at sensor device	µs	Cost at fog device	µs	Cost at central server	µs
[15]	$T_H + 5T_{mp} + T_b + 3T_{qp} + 4T_h$	–	$4T_H + 13T_{mp} + 7T_b + 7T_{qp} + 8T_h$	189.02	$T_H + 6T_{mp} + 3T_b + 4T_{qp} + 5T_h$	71.23
[8]	$T_b + 2T_{mp} + 6T_h$	–	$T_b + 2T_{mp} + 4T_h$	18.98	$T_b + 3T_{mp} + 11T_h$	19.97
Proposed	$7T_{mp} + 2T_{qp} + 2T_s + 12T_h$	$2407.83 * 10^3$	$7T_{mp} + 2T_{qp} + 2T_s + 13T_h$	6.92	$9T_{mp} + 4T_{qp} + 4T_s + 13T_h$	8.91

Note that in [15], the fog device and the central server are replaced by an authentication server and an application server, respectively. The time costs are measured on a personal computer with a 2.5 GHz CPU, 8 GB RAM, Windows 7 as OS for the FD and CS. Regarding the SD, the time costs to perform cryptographic operations are measured on the Zolertia RE-mote, which has an ARM Cortex-M3 with a 32 MHz MCU and 32 KB RAM. Note that T_b = time for bilinear pairing, T_{mp} = time for point multiplication, T_{qp} = time for point addition, T_s = time for symmetric encryption/decryption, T_H = time for map to point, T_h = time for hash operation

Table 2 Comparison of communication complexity

Scheme	$M_1(bits)$	$M_2(bits)$	$M_3(bits)$	$M_4(bits)$	$Total(bits)$
[15]	2112	2080	2080	1888	8160
[8]	864	1728	864	1216	4672
Proposed	672	1056	704	832	3264

outputs of length 160 bit, an EC with generator of order 160, and a pairing operation $e : G_1 \times G_1 \rightarrow G_2$ with $|G_1| = 512$, $|G_2| = 160$. For the symmetric key encryption, we consider the 128-bit and 192-bit AES variants. In addition, we assume that the length of identities and timestamps equals 32 bits. The Zolertia RE-mote, which acts as SD, runs the Contiki 3.0 operating system. To communicate with the FD, we use the default Contiki protocol stack that consists of IEEE 802.15.4 standard [40] for the physical layer, ContikiMAC as Radio Duty Cycle (RDC) protocol and the Carrier-Sense Multiple Access (CSMA) protocol as Medium Access Control (MAC) protocol. Since the maximum packet size defined by this standard is 127 bytes, considering the protocol headers, we only need two fragments for messages M_1 and M_4 during the key agreement phase. As can be concluded from Table 2, our scheme requires the smallest number of bits to be sent over the channel among the schemes consisting of 3 passes. More specifically, for the message M_1 sent by the most constrained device, our scheme is approximately 20% faster than [8] and 70% faster than [15].

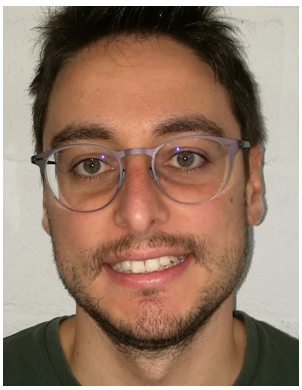
7 Conclusions

In this paper, we proposed an identity-based mutual authentication scheme to be applied in a fog architecture. The innovation of the paper is that we add to this type of scheme two very important features: the protection of session key security in the CK model and the anonymity of the sensor device with respect to the fog device and outsiders. Only the central server is responsible for the control of the identities of the sensor device and fog device. As an interesting side effect, after the execution of the scheme, every participating entity pair also possesses a unique common secret shared key. In particular, the shared key between the sensor device and the fog device enables the communication between both, which cannot be traced by the central server. It is also important to mention that no pairing operations are used in the scheme, leading to very low computation and communication overhead.

References

1. Hong, H. J. (2017). In *2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)* (pp. 331–334). IEEE. https://doi.org/10.1109/CloudCom.2017.53. http://ieeexplore.ieee.org/document/8241127/
2. Khabazian, M., Aissa, S., & Mehmet-Ali, M. (2011). Performance modeling of message dissemination in vehicular ad hoc networks with priority. *IEEE Journal on Selected Areas in Communications*, *29*(1), 61. https://doi.org/10.1109/JSAC.2011.110107.
3. Hou, X., Li, Y., Chen, M., Wu, D., Jin, D., & Chen, S. (2016). Vehicular fog computing: A viewpoint of vehicles as the infrastructures. *IEEE Transactions on Vehicular Technology*, *65*(6), 3860. https://doi.org/10.1109/TVT.2016.2532863.
4. Ahmad, M., Amin, M. B., Hussain, S., Kang, B. H., Cheong, T., & Lee, S. (2016). Health Fog: A novel framework for health and wellness applications. *The Journal of Supercomputing*, *72*(10), 3677. https://doi.org/10.1007/s11227-016-1634-x.
5. Du, M., Wang, K., Chen, Y., Wang, X., & Sun, Y. (2018). Big data privacy preserving in multi-access edge computing for heterogeneous internet of things. *IEEE Communications Magazine*, *56*(8), 62. https://doi.org/10.1109/MCOM.2018.1701148.
6. Yang, L., Ding, C., Wu, M., & Wang, K. (2017). Robust detection of false data injection attacks for data aggregation in an internet of things-based environmental surveillance. *Computer Networks*, *129*, 410. https://doi.org/10.1016/J.COMNET.2017.05.027.
7. Wang, K., Du, M., Sun, Y., Vinel, A., & Zhang, Y. (2016). Attack detection and distributed forensics in machine-to-machine networks. *IEEE Network*, *30*(6), 49. https://doi.org/10.1109/MNET.2016.1600113NM.
8. Jia, X., He, D., Kumar, N., & Choo, K. K. R. (2018). Authenticated key agreement scheme for fog-driven IoT healthcare system. *Wireless Networks*,. https://doi.org/10.1007/s11276-018-1759-3.
9. Odelu, V., Das, A. K., Wazid, M., & Conti, M. (2018). Provably secure authenticated key agreement scheme for smart grid. *IEEE Transactions on Smart Grid*, *9*(3), 1900. https://doi.org/10.1109/TSG.2016.2602282.
10. Chen, Y., Castillejo, P., López, L., Chen, Y., Martínez, J. F., Martínez, J. F., et al. (2017). An anonymous authentication and key establish scheme for smart grid: FAuth. *Energies*, *10*(9), 1354. https://doi.org/10.3390/en10091354.
11. Canetti, R., & Krawczyk, H. (2001). Analysis of key-exchange protocols and their use for building secure channels. In *International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 453–474). Berlin: Springer. https://doi.org/10.1007/3-540-44987-6_28. http://link.springer.com/10.1007/3-540-44987-6_28
12. Marvin, R. (2013). SD Times Blog: Google admits an Android crypto PRNG flaw led to Bitcoin heist—SD times. https://sdtimes.com/android/sd-times-blog-google-admits-an-android-crypto-prng-flaw-led-to-bitcoin-heist/. Accessed 18 Sept 2018.
13. Shumow, D., & Ferguson, N. (2007). Microsoft, on the possibility of a back door in the NIST SP800-90 dual Ec Prng. Tech. rep. https://rump2007.cr.yp.to/15-shumow.pdf
14. Zetter, K. (2013). How a Crypto 'Backdoor' pitted the tech world against the NSA | WIRED. https://www.wired.com/2013/09/nsa-backdoor/. Accessed 18 Sept 2018.
15. Liu, C. L., Chang, T. Y., Liu, T. M., Liu, C. L., Tsai, W. J., Tsai, W. J., et al. (2018). Ephemeral-secret-leakage secure ID-based three-party authenticated key agreement protocol for mobile distributed computing environments. *Symmetry*, *10*(4), 84. https://doi.org/10.3390/sym10040084.
16. He, D., Zeadally, S., Wang, H., & Liu, Q. (2017). Lightweight data aggregation scheme against internal attackers in smart grid using elliptic curve cryptography. *Wireless Communications and Mobile Computing*, *2017*, 1. https://doi.org/10.1155/2017/3194845.
17. Kumar, P., Braeken, A., Gurtov, A., Iinatti, J., & Ha, P. H. (2017). Anonymous secure framework in connected smart home environments. *IEEE Transactions on Information Forensics and Security*, *12*(4), 968. https://doi.org/10.1109/TIFS.2016.2647225.
18. Braeken, A., & Touhafi, A. (2016). In *2016 2nd International Conference on Cloud Computing Technologies and Applications (CloudTech)* (pp. 13–20). IEEE. https://doi.org/10.1109/CloudTech.2016.7847702.
19. Braeken, A., Kumar, P., Liyanage, M., & Hue, T. T. K. (2018). An efficient anonymous authentication protocol in multiple server communication networks (EAAM). *The Journal of Supercomputing*, *74*(4), 1695. https://doi.org/10.1007/s11227-017-2190-8.
20. Tsai, J. L., & Lo, N. W. (2015). Secure anonymous key distribution scheme for smart grid. *IEEE Transactions on Smart Grid*, *7*, 906–914. https://doi.org/10.1109/TSG.2015.2440658.
21. Wu, D., & Zhou, C. (2011). Fault-tolerant and scalable key management for smart grid. *IEEE Transactions on Smart Grid*, *2*(2), 375. https://doi.org/10.1109/TSG.2011.2120634.
22. Xia, J., & Wang, Y. (2012). Secure key distribution for the smart grid. *IEEE Transactions on Smart Grid*, *3*(3), 1437. https://doi.org/10.1109/TSG.2012.2199141.
23. Lee, T. F., & Hwang, T. (2017). Three-party authenticated key agreements for optimal communication. *PLoS ONE*, *12*(3), 1. https://doi.org/10.1371/journal.pone.0174473.
24. Gong, L. (1993). In *Proceedings of the 1st ACM Conference on Computer and Communications Security, CCS '93* (pp. 26–37). New York, NY, USA: ACM. https://doi.org/10.1145/168588.168592.
25. Lee, C. C., Chen, S. D., & Chen, C. L. (2012). A computation-efficient three-party encrypted key exchange protocol. Tech. rep. www.naturalspublishing.com/Journals.asp.
26. Li, X., Niu, J., Kumari, S., Khan, M. K., Liao, J., & Liang, W. (2015). Design and analysis of a chaotic maps-based three-party authenticated key agreement protocol. *Nonlinear Dynamics*, *80*(3), 1209. https://doi.org/10.1007/s11071-015-1937-0.
27. Ni, L., Chen, G., & Li, J. (2013). Escrowable identity-based authenticated key agreement protocol with strong security. *Computers & Mathematics with Applications*, *65*(9), 1339. https://doi.org/10.1016/J.CAMWA.2012.01.041.
28. Wang, D., & Wang, P. (2014). On the anonymity of two-factor authentication schemes for wireless sensor networks: Attacks, principle and solutions. *Computer Networks*, *73*, 41. https://doi.org/10.1016/J.COMNET.2014.07.010.
29. Al Hamid, H. A., Rahman, S. M. M., Hossain, M. S., Almogren, A., & Alamri, A. (2017). A security model for preserving the privacy of medical big data in a healthcare cloud using a fog computing facility with pairing-based cryptography. *IEEE Access*, *5*, 22313. https://doi.org/10.1109/ACCESS.2017.2757844.
30. Hankerson, D., Menezes, A. J., & Vanstone, S. (2003). *Guide to elliptic curve cryptography*. Berlin: Springer. https://doi.org/10.1007/b97644.

31. C. Research, STANDARDS FOR EFFICIENT CRYPTOGRAPHY SEC 2: Recommended elliptic curve domain parameters. Tech. rep. (2000). http://www.secg.org/SEC2-Ver-1.0.pdf
32. Dworkin, M. (2017). Digital signatures | CSRC. https://csrc.nist.gov/projects/digital-signatures. Accessed 19 Sept 2018.
33. C. Research, Standards for efficient cryptography SEC 1: Elliptic curve cryptography. Tech. rep. (2009). http://www.secg.org/sec1-v2.pdf
34. Qu, M., & Vanstone, S. A. (2004). Implicit certificate scheme. Tech. rep. https://patentimages.storage.googleapis.com/cf/af/a5/2fa3749417d71b/US6792530.pdf.
35. Tedeschi, P., Piro, G., & Boggia, G. (2018). In *2018 IEEE Globecom Workshops (GC Wkshps)* (pp. 1–6). IEEE. https://doi.org/10.1109/GLOCOMW.2018.8644494.
36. Brown, D. R. L., Gallant, R., & Vanstone, S. A. (2002). Provably secure implicit certificate schemes. In *International Conference on Financial Cryptography* (pp. 156–165). Berlin: Springer. https://doi.org/10.1007/3-540-46088-8_15.
37. Pointcheval, D., & Zimmer, S. (2008). Multi-factor authenticated key exchange. In *Applied Cryptography and Network Security* (pp. 277–295). Berlin: Springer. https://doi.org/10.1007/978-3-540-68914-0_17.
38. Shoup, V. (2004). Sequences of games: A tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332. https://eprint.iacr.org/2004/332
39. Malina, L., Hajny, J., Fujdiak, R., & Hosek, J. (2016). On perspective of security and privacy-preserving solutions in the internet of things. *Computer Networks*, *102*, 83. https://doi.org/10.1016/J.COMNET.2016.03.011.
40. Standards, M. (2006). Committee of the IEEE Computer Society, IEEE Std 802.15.4-2011, IEEE Standard for Local and metropolitan area networks—Part 15.4: Low-rate wireless personal area networks (WPANs). Tech. rep. http://ecee.colorado.edu/~liue/teaching/comm_standards/2015S_zigbee/802.15.4-2011.pdf



Simone Patonico obtained the Bachelor and Master degree in Electronics Engineering from Università Politecnica delle Marche (UNIVPM) respectively in 2014 and 2017. Currently, he is a Ph.D. student under the supervision of Prof. Kris Steenhaut and Prof. An Braeken at the Department of Electronics and Informatics (ETRO) at Vrije Universiteit Brussel (VUB). As member of the research group, he worked on the Horizontal-IoT project to investigate the interoperability between different application protocols using the oneM2M standard. He also contributed to the InterOM2M project which focuses on the creation of a common middleware to link different interoperable frameworks. His research interests include the investigation, design and implementation of communication and security protocols in wireless sensor networks.

An Braeken obtained her M.Sc. Degree in Mathematics from the University of Gent in 2002. In 2006, she received her Ph.D. in engineering sciences from the KULeuven at the research group COSIC (Computer Security and Industrial Cryptography). She became professor in 2007 at the Erasmushogeschool Brussel (currently since 2013, Vrije Universiteit Brussel) in the Industrial Sciences Department. Prior to joining the Erasmushogeschool Brussel, she worked for almost 2 years at the management consulting company Boston Consulting Group (BCG). Her current interests include security and privacy protocols for IoT, cloud and fog, blockchain and 5G security. She is (co-)author of over 120 publications. She has been member of the program committee for numerous conferences and workshops (IOP2018, EUC 2018, ICNS 2018, etc.) and member of the editorial board for Security and Communications magazine. She has also been member of the organizing committee for the IEEE Cloudtech 2018 conference and the Blockchain in IoT workshop at Globecom 2018. In addition, she is since 2015 reviewer for several EU proposals and ongoing projects, submitted under the programs of H2020, Marie Curie and ITN. She has cooperated and coordinated more than 12 national and international projects. She has been STSM manager in the COST AAPELE project (2014–2017) and is currently in the management committee of the COST RECODIS project (2016–2019).

Kris Steenhaut received the master in Engineering Sciences in 1984 and the master in Applied Computer Sciences in 1986 and the Ph.D. degree in Engineering Sciences from Vrije Universiteit Brussel (VUB) in 1995. Currently she is professor at the department of Electronics and Informatics (ETRO) and the department of Engineering Technology (INDI), Faculty of Engineering Sciences, Vrije Universiteit Brussel, Belgium. Her research interests focus on the design, implementation and evaluation of (Wireless) Sensor Network protocols for building automation, environmental monitoring and smart grids. She has authored over 150 journal and conference publications, including book chapters.