IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 19, 2019, accepted October 3, 2019, date of publication October 11, 2019, date of current version October 28, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2946713

Fog-Orchestrated and Server-Controlled Anonymous Group Authentication and Key Agreement

PLACIDE SHABISHA[1,2] AN BRAEKEN[2], PARDEEP KUMAR[3], (Member, IEEE), AND KRIS STEENHAUT[1], (Member, IEEE)

[1]Department of Electronics and Informatics (ETRO), Vrije Universiteit Brussel, 1050 Brussels, Belgium
[2]Department of Engineering Technology (INDI), Vrije Universiteit Brussel, 1050 Brussels, Belgium
[3]Department of Computer Science, Swansea University, Swansea SA2 8PP, U.K.

Corresponding author: Placide Shabisha (placide.shabisha@vub.be)

This work was supported in part by the VLIR-UOS (Project: IUC 2017 Phase 3 UB BI2017IUC022A102), and in part by the Vrije Universiteit Brussel (2 Pleinlaan, 1050 Elsene, Brussels, Belgium, VAT-number: BE 0449 012 406).

ABSTRACT Fog architectures are currently present in many applications. Constrained devices equipped with sensors produce measurements that will be sent to a nearby gateway, called the fog. The fog verifies, aggregates and forwards them to the server. Group authentication among these devices allows them to securely accept messages of the group members, resulting in faster updates in their process. When defining a security scheme, it should be considered that edge and fog devices are susceptible to attacks. Privacy of the devices should be guaranteed, with respect to outsiders and the fog. It should be impossible to track the connection pattern of devices with different fogs, even if several fogs are captured by an attacker. Inclusion of protection against potentially malicious fogs has not yet been considered in literature, especially not for group-based communications. We present a server-controlled group authentication and key agreement scheme, executed by the fog in collaboration with the devices that it can reach. The server, assumed to be fully trusted, is responsible for the registration and authorisation of the devices and initiates the key update process, whereas the fog takes care of the secure distribution process among its members. At the end, all entities in the group are ensured to possess the correct group key. Moreover, a pairwise secret key between device and server is obtained during the process. The proposed scheme is very efficient as it relies on elliptic curve cryptography and Lagrange interpolation. No initially shared secret key material among the entities needs to be pre-stored.

INDEX TERMS Edge-fog-cloud architecture, elliptic curve cryptography, group authentication, group key, Rubin logic.

I. INTRODUCTION

The Internet of Things (IoT) is currently involved in many applications, offering a wide variety of services to users. A few years ago, fog computing was proposed as a paradigm to extend the computation, storage and application services to the edge of the network [1]. Fog computing can be considered as a complement of the cloud computing near the IoT devices. The proximity of the fog to the end-users and the IoT devices enhances the QoS and reduces latency [2]. In the fog architecture, the fog takes care of local data analysis and storage at the ground (field) whereas the cloud performs global data analysis and storage [3]. Surveys [4]–[6] provide a clear summary of the advantages and the efficiency of fog computing compared to cloud computing in terms of location awareness, hardware size, easy deployment, decentralized and simplified operations, time criticalness, internet connectivity and bandwidth usage. Several use cases based on fog computing in the sphere of health care systems [6]–[8], smart traffic lights and connected cars, decentralized smart building control [9], big data analysis for smart cities [10], intelligent transport systems [11], [12] are well known.

The use of group keys in communication increases the efficiency as it allows to ensure authenticity of a broadcasted message and to identify potential issues by a larger group. The inclusion of group authentication into the fog architecture

The associate editor coordinating the review of this manuscript and approving it for publication was Junaid Arshad.

has the advantage that all devices in each other's vicinity are able to immediately verify the authenticity of the message if broadcasted. The devices do not need to wait for confirmation either from the fog or from the server. It is clear that the larger the group, the higher the attack surface, as more users are aware of the same key and can be subject to an attack. Therefore, it is very important to regularly update this group key. In case of moving devices, like in the use case of vehicular networks, the group structure changes frequently.

Most group authentication schemes in literature concentrate on communication between devices in a group and the gateway without involving the server. We believe that it is important to include the server into the process as it has lower probability of being compromised compared to the fog since it is not located in the field and is more powerful. Doing so avoids that devices completely depend on the fog for key distribution. The server can enforce strict security management, like imposing regular group key updates to the fog and involving it is the only possibility to guarantee device anonymity towards the fog. Anonymity is very relevant and should be taken into account due to growing privacy concerns.

This paper is organised as follows. We start by a discussion on related work in Section II. The required preliminaries with respect to security mechanisms and properties are provided in Section III. The actual scheme is presented in Section IV. We then proceed with protocol verification based on Rubin logic in Section V followed by security analysis in Section VI. Performance evaluation is presented in Section VII and we finish with a conclusion in Section VIII.

II. RELATED WORK

We first discuss related work in the context of client-server architectures, Wireless Sensor Networks (WSNs), and later for fog architectures.

A. CLIENT-SERVER ARCHITECTURE

Authentication of an IoT device to another entity, so-called client-server scenario, is well studied. Many surveys, like most recent [13], have been made on this topic. In particular, when the client represents a device with user interaction, e.g. requiring the usage of a smart card or smart phone, a multitude of two-factor and three-factor authentication schemes exist in literature. For instance, in [14], a symmetric key-based scheme offering mutual authentication with anonymity and untraceability has been proposed. In [15], [16], also protection against an honest but curious trusted third party (TTP) has been included, but requires public key based operations, being elliptic curves in [15] and chaos-based operations in [16].

When the client represents a device, the list of mutual identity based authentication schemes in literature is more restricted, with the most relevant ones [17]–[21]. In [20], [21], the architecture consists of an interactive TTP, which is not the case in [17]–[19]. Moreover, the schemes of [17]–[19] offer anonymity of the client.

B. WIRELESS SENSOR NETWORKS

WSNs present an important part of IoT. For this type of networks several group authentication and key management schemes have been proposed, based on symmetric, public key or hybrid mechanisms. There are two main categories in this type of schemes, corresponding with a hierarchical and non-hierarchical group structure. Symmetric key based proposals mainly consider a hierarchical architecture [22]–[24] as do several hybrid schemes [25]–[28]. Hierarchical network structures are often less flexible and efficient, since it is not possible to create custom and ad-hoc groups. They are mainly limited to static architectures.

The setting, mostly related to fog architecture, corresponds to the one of non-hierarchical or distributed networks, where one of the nodes takes the role of group leader. Here, we can distinguish the following schemes [29]–[33]. In [29], a symmetric key based scheme has been proposed in which a cloud server appoints one of the members of the WSN as group leader and shares the required key material with it. Another symmetric key based solution is proposed in [30], for which a robot-assisted network bootstrapping technique is put forward. In our opinion, the presence of a robot to distribute key material is not a very realistic assumption. The schemes in [31], [32] are based on a key tree which results in communication and computation costs which linearly increase with the number of group members. Finally, in [33], a secure and efficient key management scheme is proposed for multicast communication based on Elliptic Curve Cryptography (ECC). Due to the nature of their architecture, none of these proposed schemes offer anonymity of the group members.

C. FOG ARCHITECTURE

For architectures including the fog, often called tripartite architectures (architectures with three types of participants), several identity-based mutual authentication schemes have been proposed. We can distinguish symmetric key based schemes like e.g. [34]–[37], and public key based schemes like e.g. [38]–[40]. The main disadvantage of the symmetric key based schemes is their inability to offer anonymity, as discussed in [41]. The scheme in [40] proposes a secure identity based mutual authentication scheme, applicable to mobile distributed computing environments but fails to offer anonymity and needs compute intensive pairing operations. A key agreement scheme for a fog-based healthcare application is proposed in [39], as an improvement of [38] by including protection for perfect forward secrecy. The scheme in [39] is able to offer anonymity, but requires a smart card based entry and relies on compute intensive pairing operations. Contribution [42] is proposed for operations in mobile distributed computing environments in which the end device first communicates with the authentication server that provides the access control to the application server. However, no anonymity is offered in here. Finally, in [43], an efficient, elliptic curve and identity based mutual authentication protocol with anonymity protection for devices is proposed. This

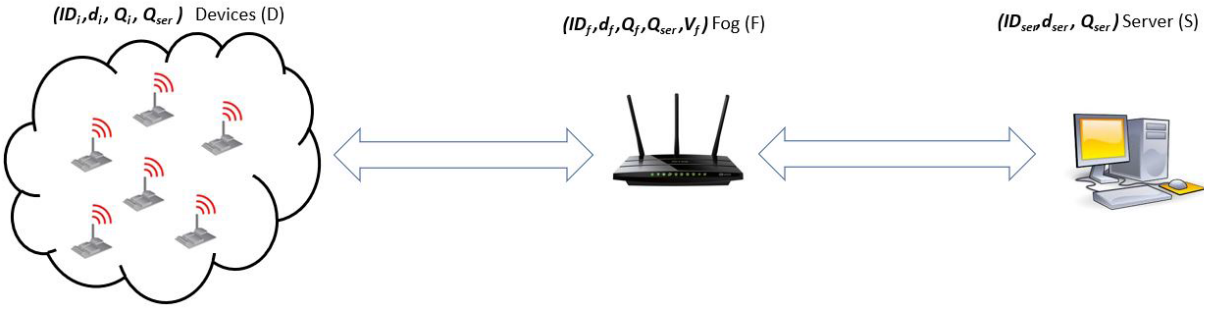

FIGURE 1. Fog architecture setup with three types of participants: Devices, fog and server.

scheme also offers protection in case of a Canetti-Krawczyk (CK) adversary model [44].

In [45], a group key is constructed by the fog based on partial secrets shared by the server with each of the IoT devices belonging to that group. This key is used to guarantee secure data uploading between the server and the fog. As far as the authors are aware, there is no real group key management scheme through which all entities including server, fog and devices, possess a common shared key.

III. PRELIMINARIES

A. SECURITY MECHANISMS

The proposed scheme heavily relies on ECC [46] as it offers lightweight public key cryptographic (PKC) solutions. Denote the elliptic curve EC in the finite field F_p by $E_{p(a,b)}$, and the base point generator of prime order q by P. All points on $E_{p(a,b)}$, together with the infinite point form an additive group. In [47], [48] standardised curve parameters are described.

The product $R = rP = (R_x, R_y)$ 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 is sufficient to send its x coordinate, together with a single sign bit, cf. the SEC1 based encoding [49]. 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 G, generated by an elliptic curve (EC) of order q, it is computationally hard for any polynomial-time bounded algorithm to determine a parameter $x \in F_q$, such that $Q = xR$.
- The Elliptic Curve Diffie Hellman Problem (ECDHP). Given two points $R = xP, Q = yP$ of an additive group G, generated by an EC of order q with two unknown parameters $x, y \in F_q$, it is computationally hard for any polynomial-time bounded algorithm to determine the EC point xyP.

We will use the Schnorr signature algorithm to compute signatures [50]. In the Schnorr algorithm, the sender chooses a random value u in F_q, to compute $U = uG$. Next, it computes a hash value $h = H(m\|U)$ of message m and random U. The parameter s is then defined as $s = u - hd_S$, with d_S the private key of the sender. The signature on m using the private key d_S is then defined as $\{m\}_{d_S} = \{m, h, s\}$. The receiver of this signature now computes $U = sG + hQ_S$ and checks if $h = H(m\|U)$ corresponds to the received value h. With respect to computational complexity, the impact for the generation is mostly determined by the EC multiplication, whereas for the verification there are two EC multiplications and one EC addition required.

We denote the operation H as the one-way cryptographic hash function (e.g. SHA2 or SHA3) that results in an element of F_p. The encryption of a message M using secret key K is denoted by $C = E_K(M)$ and decryption by $M = D_K(C)$. As encryption algorithm AES or even a lightweight cryptographic algorithm can be used. The concatenation of two messages M_1 and M_2 is denoted by $M_1\|M_2$. We assume that these functions and the EC parameters, together with the EC operations, are implemented in each entity participating the scheme.

B. SYSTEM ARCHITECTURE

We here consider a so-called fog architecture, as depicted in Figure 1. This architecture consists on the bottom layer of heterogeneous IoT devices, which are the most constrained devices in the network. In the middle, a gateway also called fog is situated, which collects, filters, and aggregates the data of the devices and further forwards them to the server. On the top, a powerful server mostly located in the cloud, is positioned. On the server, the data are securely stored and eventually analysed. This data can then be later used by a whole set of applications, if the required authorisation is assigned. The focus of this paper is on addressing the security issues present in the fog architecture and not on access control of applications or users to the data stored on the server (cf. [51] for survey).

C. SECURITY REQUIREMENTS

First of all, the proposed scheme satisfies general security features.

- Authentication: all entities in the scheme should be authenticated. It should not be possible for an attacker to take over the position of the one claimed to be involved in the system. As a result, this property avoids man-in-the-middle attacks and impersonation attacks.

- Integrity: it should not be possible to alter the message without it being noticed by the intended receiver.
- Anonymity: it should not be possible by an outsider and even group member (device and fog) to derive the identity of the sending device in the group.
- Unlinkability: an outsider, fog or other device should not be able to link the participation of a certain device to different groups coordinated by different fogs.
- Group forward secrecy: if one of the long-term private keys of the devices or fog is leaked, access is no longer allowed to group communication content.
- Backward confidentiality: devices who join the group cannot get the former group keys from their present key.
- Perfect forward secrecy: previous group keys should be still secure even if the long-term secrets are compromised during an attack.

The presented fog architecture realises additional security features being:

- By the end of the process, both fog and cloud are ensured that all entities of the group possess the required group key.
- The devices in the group are ensured that the fog has a secure relation with the server.
- By the end of the process, both a group key among all entities and a common pairwise key between device and fog, between device and server, and between fog and server, are derived.

IV. SECURITY SCHEME

The proposed scheme consists of three different phases: initialisation phase, actual group authentication and key agreement phase, and the communication phase.

A. INITIALISATION PHASE

All entities are identified by their public key. We here assume that devices, fog and server in the possession of a public key, are authorised by a trusted third party to participate in the scheme. Note that efficient certificate schemes, like Elliptic Curve Qu-Vanstone (ECQV) mechanisms [52], exist to construct the public key given identity and certificate, where the certificate has a length equal to the length of a point of the elliptic curve. For simplicity in the explanation, we use the public key of the entity as reference to the identity. We also assume that the server possesses the list of authorised devices. We further consider that the server has computed for each fog a random point $V_{fj} = v_{fj}G$ with v_{fj} a randomly chosen element of F_p.

This value V_{fj} is public. The server keeps the list $\{v_{fj}, V_{fj}\}_j$ for all the j fogs in its control range. The private/public key pairs of the i-th device is noted by (d_i, Q_i), the j-th fog by (d_{fj}, Q_{fj}) and server with identity ID_{ser} by (d_{ser}, Q_{ser}). The device and fog install all the required security mechanisms, together with the public key of the server Q_{ser}.

In the initialisation phase, the group structure should be established and stored both at the fog for its specific group and at the server for all the groups. We propose the following two set-ups, corresponding to a static and dynamic topology.

1) STATIC TOPOLOGY

This scenario is applicable in the smart grid domain, where smart meters first register to the operator and subsequently become assigned to the corresponding local aggregator. This means the device registers with a server and has to get a fog assigned. For simplicity in notation, let us assume that the fog with public key Q_f and associated random value V_f is selected. The different steps are presented in Figure 2.

In order to hide the identity of the device to the fog, we apply a mechanism used in Monero blockchains to achieve anonymity of the transactions [53]. Suppose the device sends the following message to the server $E_K(TS, Q_i), TS, W_i, H(d_iQ_{ser}\|TS\|W_i)$ with TS the current timestamp, $W_i = w_iG$ with w_i a randomly chosen element of F_q and $K = w_iQ_{ser}$. Upon arrival of this message, the server is able to decrypt the message, to retrieve the device identity, to verify the hash and to check the authorisation. Next, it computes a new public key for the device based on the fog to which it will be assigned, $Q_i^f = H(v_fQ_i\|d_{ser}Q_i\|TS)G + W_i$. The server then stores in its memory for each device Q_i, (TS, Q_i^f, Q_f, W_i). Next, it sends to the fog $TS, Q_f, V_f, H(Q_i^f\|Q_f\|TS)$ which stores the hash $H(Q_i^f\|Q_f\|TS)$ and forward all the received messages to the device.

Then, the device computes $d_i^f = H(d_iV_f\|d_iQ_{ser}\|TS) + w_i$ and $Q_i^f = d_i^fG$. With this value, the device is able to verify the hash value and to be ensured that the message comes from the server. If positive, the device installs $Q_f, Q_i^f, (d_i, Q_i)$ together with w_i, W_i in its memory. The server sends securely to the fog the credentials Q_i^f, W_i of its new member via the message $TS, E_K(Q_i^f, W_i)$ with $K = H(d_{ser}Q_f\|W_i)$. The fog verifies the hash $H(Q_i^f\|Q_f\|TS)$ and if the verification is successful it stores the values (Q_i^f, W_i).

2) DYNAMIC TOPOLOGY

In this scenario, the device moves across areas covered by different fogs, like for instance in vehicular ad hoc networks where the fog represents the roadside unit (RSU), the device computes for each fog upon its arrival at TS its new public key pair (d_i^f, Q_i^f) based on the random value V_f of the fog and timestamp TS like before. Note that this process is initiated after receiving a broadcast message of the fog containing (V_f, Q_f). The different steps for this topology are presented in Figure 3.

As a consequence, the device constructs the message $E_K(TS, Q_i), TS, W_i, H(Q_i^f\|W_i\|TS)$ with TS the current timestamp, $W_i = w_iG$ a random EC point, $K = w_iQ_{ser}$ and $Q_i^f = (H(d_iV_f\|d_iQ_{ser}\|TS) + w_i)G$. The message is then sent to the fog, which forwards the message, after adding its own identity, to the server. The fog also temporarily stores this hash value together with W_i, TS. The server first computes

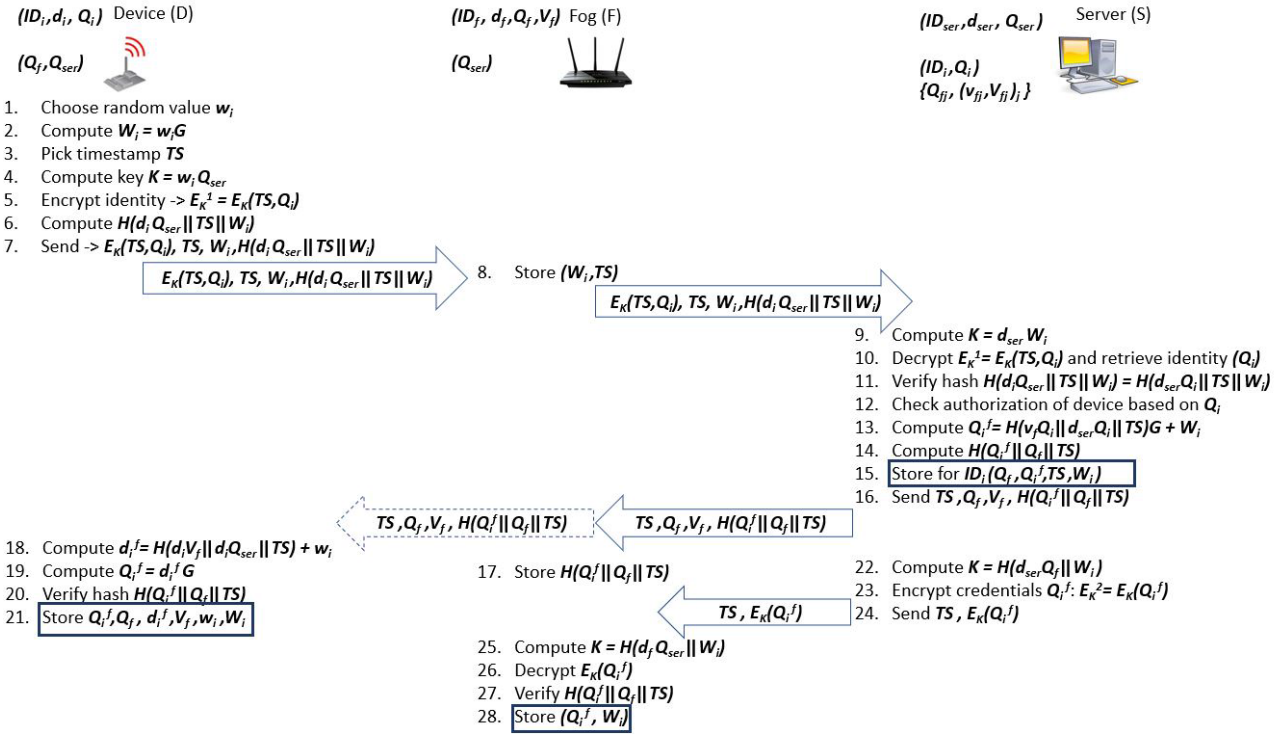

FIGURE 2. Initialization phase for static topology where devices are assigned to a fixed fog.

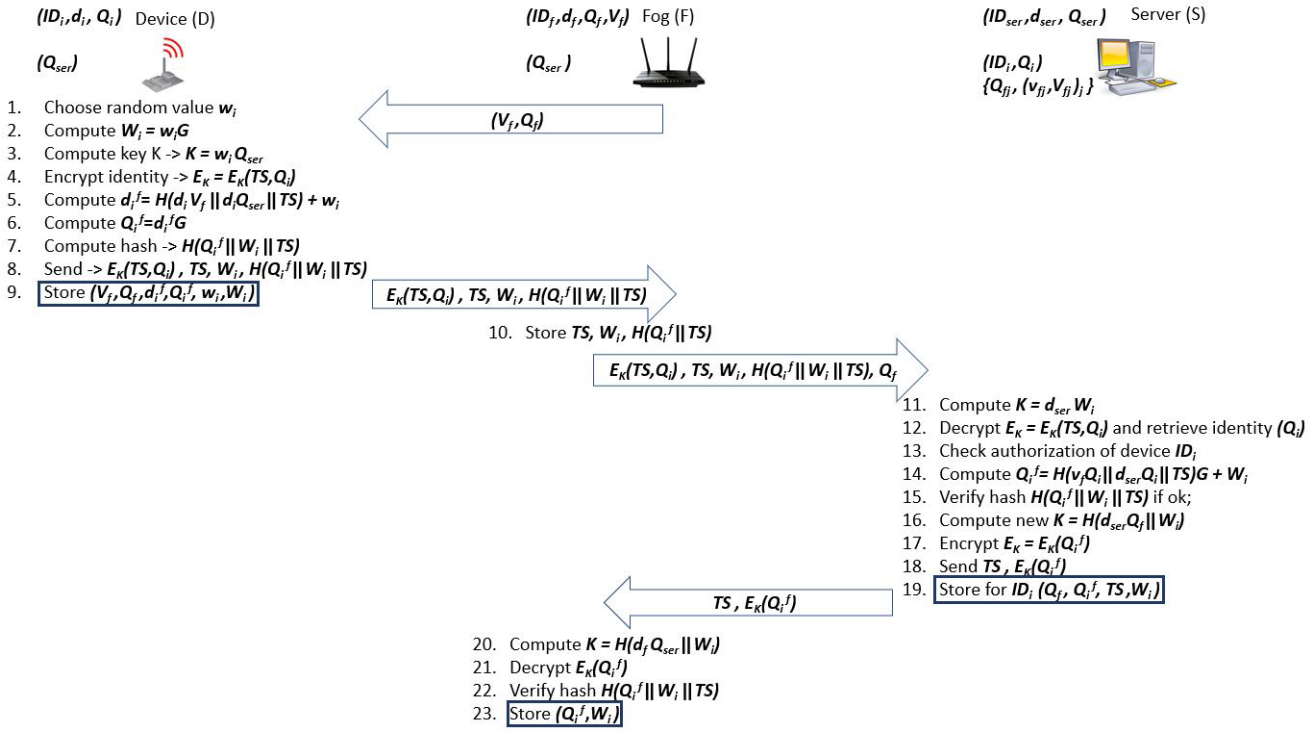

FIGURE 3. Initialization phase for dynamic topology where devices are moving and can be assigned to a different fog depending on their location.

$K = d_{ser} W_i$, decrypts the message, checks the identity of the device corresponding to Q_i and corresponding authorisation, computes the new public key $Q_i^f = H(v_f Q_i \| d_{ser} Q_i \| TS)G + W_i$, and verifies the hash value. If all is positive, the server notifies the device that it is in possession of the correct credential Q_i^f of the fog and also notifies the fog of the new member. Therefore, it sends $E_K(Q_i^f)$, TS with new $K = H(d_{ser} Q_f \| W_i)$ to the fog, which decrypts the message using

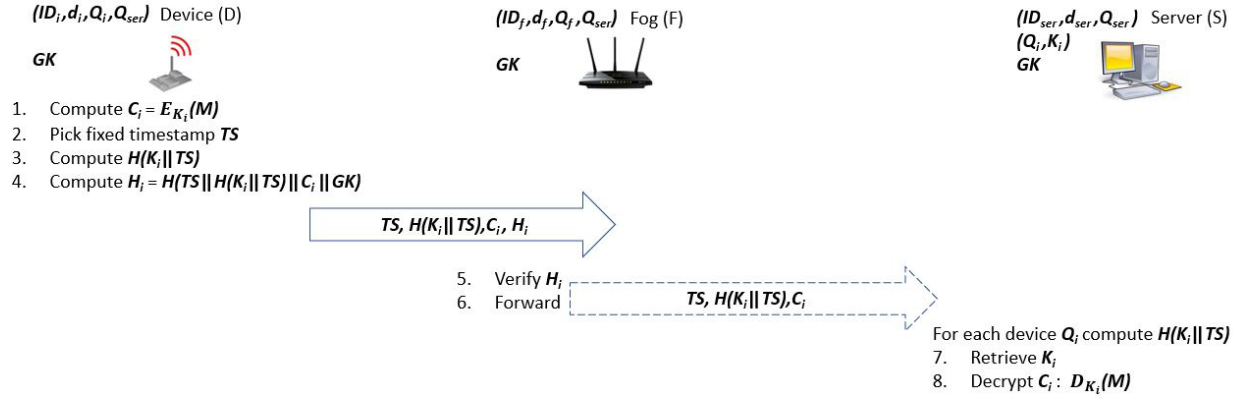

FIGURE 4. Encrypted communication message flow. The message from the device is sent to the server after authentication by fog.

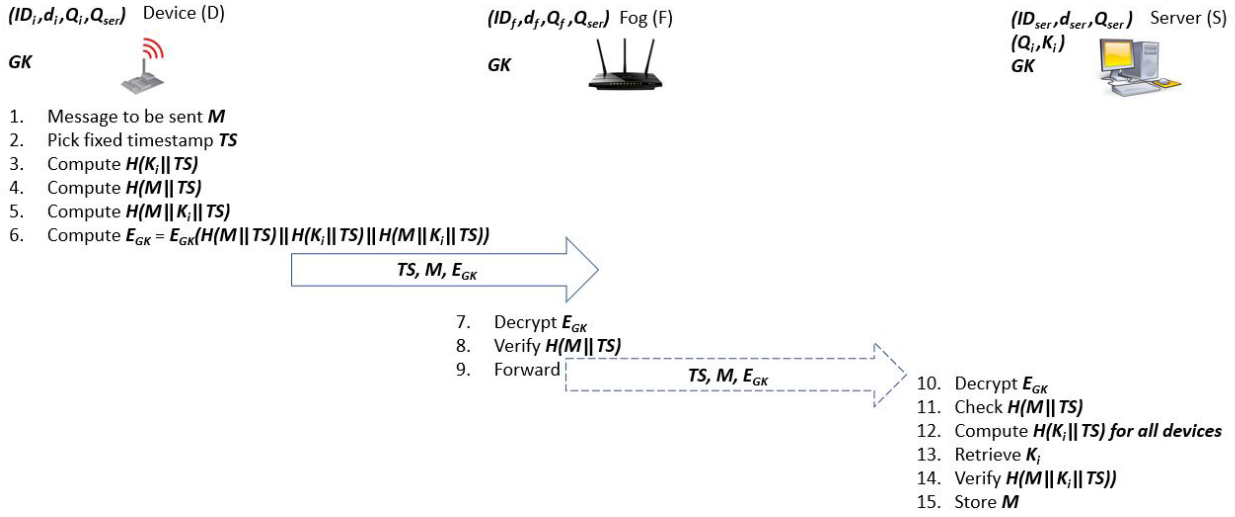

FIGURE 5. Non encrypted communication message flow. The message from the device is sent in plaintext to the server after authentication by fog.

$K = H(d_f Q_{ser} \| W_i)$ and verifies if the hash $H(Q_i^f \| W_i \| TS)$ of these values corresponds with the stored hash.

B. GROUP AUTHENTICATION AND KEY AGREEMENT PHASE

In this phase, we distinguish four major steps: request of update, response by devices, response by fog, and acknowledgements. These steps are depicted in Figure 7.

1) STEP 1 : REQUEST OF UPDATE

This phase should be initialised by the fog (potentially after a trigger of the server) by sending the request $ID_f, M_{ReqFog}, \{R_f\}_{d_f}$, with ID_f the identity of the fog. Here, M_{ReqFog} refers to the request message of update and the EC point $R_f = r_f G$, with r_f a unique random value (nonce) chosen by the fog. Moreover, the point R_f needs to be signed by the fog in order to guarantee the authenticity.

Upon arrival of this message, the server first verifies the signature using Q_f. After a positive check, the server chooses also a unique random value r_s (nonce) and computes $R_s = r_s G$. Then, for all of the n members in the group, the server computes the points $K_i = r_s Q_i^f + d_{ser} W_i$ and $K_i' = H(K_i \| R_f)$ for all $1 \leq i \leq n$. Also the operation $K_f = r_s Q_f + d_{ser} R_f$ and $K_f' = H(K_f)$ is done for the fog. Next, using these $n + 1$ points $\{K_1', ..., K_n', K_f'\}$ (x, y coordinates consist of the first and last 128 bits) a polynomial of degree n is constructed by means of the Lagrange interpolation. The y coordinate of the intersection of the polynomial with the Y-axis, i.e., $(0, GK)$, is now taken as the secret group key GK. To finalise this step, the message $M_{ReqSer}, \{R_f, R_s, H(GK)\}_{d_{ser}}$ is sent to the fog and further forwarded to the devices in its group after a positive verification of the signature.

The server stores the group key GK, the common shared key with the fog K_f, and for each group member the individual shared key K_i with $i \in \{1, ..., n\}$.

2) STEP 2 : RESPONSE BY DEVICES

Each of the n devices first start verifying the uniqueness of the request and the signature using Q_{ser}. After positive

verification, the two EC points, $K_i = d_i^f R_s + w_i Q_{ser}$ and $K_i^f = d_i^f R_f + w_i Q_f$, are computed. Then, the values $C_i^f = E_{K_i^f}(H(K_i \| R_f))$ and $N_i^f = H(K_i^f)$ are computed. Finally, C_i^f, N_i^f are sent to the fog.

3) STEP 3 : RESPONSE BY FOG

The fog first computes for each device $K_i^f = r_f Q_i^f + d_f W_i$. Then, thanks to the value N_i^f, the fog is able to identify the device sending the message and thus to decrypt the other part of the message C_i^f in order to find K_i'. Using all received values, together with its own point $K_f = d_f R_s + r_f Q_{ser}$, the fog is able to reconstruct the polynomial and to find the corresponding intersection with the Y-axis and thus the group key GK.

Next, the fog derives n other points P_i of the polynomial and sends the values $(P_1, ..., P_n)$ to the devices in its group.

4) STEP 4 : ACKNOWLEDGEMENTS

Upon arrival of this message, the device constructs also by means of Lagrange interpolation the function going through the points $(P_1, ..., P_n)$ and its own point K_i'. After intersection with the Y-axis, the point GK should be found for which $H(GK)$ corresponds with the value received during the message request of the server. If so, the device is ensured of the authenticity of the response of the fog. The device chooses a new random w_i^n, computes $W_i^n = w_i^n G, h = H(GK, W_i^n, W_i)$, $s = w_i^n - h d_i^f$ and sends W_i, W_i^n, s to the fog.

Upon receiving this message from the device, the fog computes $h = H(GK, W_i^n, W_i)$, and verifies $sG = W_i^n - hQ_i^f$. After a successful verification, it then replaces W_i by W_i^n and forwards the received messages to the server.

The server will now execute the same previous steps as the fog: compute h, verify sG and replace W_i by W_i^n. Now the server is ensured that all involved entities share the same group key GK and has updated the temporary key material of each IoT device.

C. COMMUNICATION PHASE

There are two possibilities in this phase for the communication of messages from devices to server, either encrypted or not encrypted, but in any case, authenticated by the group key.

1) ENCRYPTED COMMUNICATION

The device shares the key K_i with the server, which is used to encrypt the message $C_i = E_{K_i}(M)$. In order to hide the identity, the device is limited to sending at fixed timestamps TS and computes the corresponding value $H(K_i \| TS)$. Next, in order to allow the fog to verify the authenticity of the message, the following hash value should be included $H_i = H(TS \| H(K_i \| TS) \| C_i \| GK)$. Consequently, the message to be sent equals to $TS, H(K_i \| TS), C_i, H_i$. After verification of H_i by the fog, the fog forwards $TS, H(K_i \| TS), C_i$ to the server.

At server side, for each timestamp and each registered device Q_i, the value $H(K_i \| T_S)$ is computed. As a consequence, upon arrival of the message, the server can identify the origin of the device, link it with the stored key K_i and decrypt C_i in order to derive the message.

2) NON-ENCRYPTED COMMUNICATION

In this case, a device transmits the message M and timestamp TS accompanied with $E_{GK}(H(M \| TS) \| H(K_i \| TS) \| H(M \| K_i \| TS))$. After receiving these messages, the fog verifies $H(M \| TS)$ and in case of a positive verification, it forwards the received messages to the server. The second part allows precomputation at server side in order to facilitate the identification of the transmitting entity which can be verified by the third part.

V. RUBIN-LOGIC BASED VERIFICATION

This section formally verifies the proposed scheme, i.e., the group authentication and key agreement scheme, using the RUBIN logic as it is very close to the actual implementation of the protocol. Note that we have directly adopted RUBIN logic from [54]–[56].

A. BACKGROUND OF RUBIN LOGIC

Rubin logic integrates protocol analysis with specifications and uses the notions of global sets, local sets, and actions, as follows.

1) GLOBAL SET

The Global Set comprises of various sets that are required to represent information in the protocol. The content of the global sets may change as the protocol is running. Moreover, these sets are public to each principal in a protocol specification.

1) *Principal Set*: It contains the information of principals who participate in a protocol.
2) *Rule Set*: It contains inference rules and how to derive new statements from existing assertions.
3) *Secret Set*: This set refers to all the secrets that exist at any given time in the system.
4) *Observer set*: It contains all principals who possibly know the secrets by listening to the communication channel.

2) LOCAL SET

A Local Set comprises of various sets that are private to each participant and it consists of the following:

1) *Possession set (P_i)*: The possession set contains all the security parameters, known or in possession of an entity, which includes encryption keys, public keys, and other secrets that are not publicly available. The set is represented by $POSS(P_i) = (poss_1, poss_2, ..., poss_n)$.
2) *Belief set*: It contains all the beliefs held by a legitimate principal. For example, the beliefs about freshness, and the beliefs about the possessions of other involved participants and principals. This set is denoted by $BEL(P_i) = bel_1, bel_2, ...bel_n$.

TABLE 1. Actions in Rubin logic.

Action	Condition	Result	Description
$Hash(h(.); X)$	$h(.), X \in POSS(P_i)$	$POSS(P_i) := POSS(P_i) \cup h(X)$	This is for hashing the data
$Generate - nonce(N)$	–	N	This is a random nonce
$Encrypt$	$X, k \in POSS(P_i)$	$POSS(P_i) := POSS(P_i) \cup Xk$	The P_i encrypts own data.
$Decrypt(Xk, k)$	$Xk, k \in POSS(P_i)$	$POSS(P_i) := POSS(P_i) \cup X$	The P_i decrypts data
$Generate - secret(z)$	–	$S := S \cup z$, $Observers(z) = P_i$, $POSS(P_i) := POSS(P_i) \cup z, P_i$, $BEL(P_i) := BEL(P_i) \cup (z)$	Generates a secret for an entity, when needed. Thereafter, a new secret z, is added to secret S and the *Observers* and *Possession* sets are updated
$Check(X, Y)$	$X, Y \in POSS(P_i)$	Valid if $X = Y$, otherwise invalid	Verifies both values are correct
$Send(P_j, X)$	–	–	This action sends message X to P_j (i.e., P_i to P_j).
$Receive(P_j, X)$	–	–	This action receives message X from P_j
$Update(X)$	–	–	The purpose of update action is to update the Observer sets of all secrets.
$Abort$	–	–	Aborts the system, if protocol run is illegal

3) *Seen set (P_i)*: It contains plaintext message that P_i can see from messages sent over the network and it also contains a copy of the information.
4) *Behavior list (P_i)*: It contains elements list. $BL = AL, bev_1, bev_2, ...bev_n$, here AL is a list of actions, which are executed by P_i and bev_k is a pair, i.e., (message, AL). Note that the action list has two types of messages: Send (P_i, message) and Receive (P_i, message).
5) *Haskeys set (P_i)*: It contains keys that P_i can see, either as they are in an initial possession set or as they appear in a message sent across the communication and are added to the Seen set of P_i.

3) ACTIONS

The actions are paramount operations, which are used in the protocol specification. These actions are basically used to control the state of knowledge and possessions for involved entities. Considering our requirements, following actions are defined as shown in Table 1.

4) INFERENCE RULES

We defined the inference rules that are required as per our requirements. Note that these rules are directly adopted from [54]. Moreover, in order to understand the inference rules, few notations are adopted: **X contain Y**: Y appears as a sub-message of X; **S:= F(S)** : S is replaced by the value of F(S); **X from P**: X is received from P; and **LINK(N)**: this formula basically links a response to a challenge (e.g., if a principal generates a fresh signature or a fresh nonce) then **N** is added to the belief set of the principal [54].

1) *Message-meaning rule*:
$$\frac{\{X\}k \text{ from } P_i \in POSS(P_i), \{P_i, P_j\} \subseteq POSS(P_i)}{BEL(P_i) = BEL(P_i) \cup \{X \in POSS(P_i)\}}$$
2) *Origin rule*:
$$\frac{X \in POSS(P_i), X \text{ contain } x1, P_j \in Observers(x1)}{x1 \text{ from} P_j \in POSS(P_i)}$$
3) *Sub-message origin rule*:
$$\frac{X \in POSS(P_i), X \text{ contain } \{x1, x2\} \text{ from } P_j}{x2 \text{ from } E_j \in POSS(P_i)}$$

B. VERIFICATION USING RUBIN LOGIC

1) THE SPECIFICATIONS FOR THE GROUP AUTHENTICATION AND KEY AGREEMENT PHASE

a: GLOBAL SET

1) Principal Set: D,F,S. F is the initiator of the protocol.
2) Rule Set: The inference rules are defined as above-mentioned.
3) Secret Set: $\{d_i, d_f, d_{ser}, GK\}$
4) Observer Set:
Observer(d_f): {F}
Observer(d_{ser}): {S}
Observer(d_i): {D}
Observer(K_i): {S,D}
Observer(K_i^f): {D,F}
Observer(K_f): {S,F}
Observer(Q_i^f): {S,F,D}
Observer(W_i): {S,F,D}
Observer(GK): {S,F,D}

b: LOCAL SET

This set is defined for each principal, i.e., F, S, and D, respectively. As the communication is being initiated by F, we start as follows.

- Principal F
POSS(F): $\{Q_f, d_f, ID_f, V_f, Q_{ser}, Q_i^f, W_i, Q_i\}$
BEL(F): $\{d_f, \#r_f\, Q_{ser}, Q_i, Q_i^f\}$
BL(F) =
F1: Generate-nonce(r_f)
F2: $R_f \leftarrow r_f G$
F3: Send(S, M_{ReqFog}, ID_f, $\{R_f\}_{d_f}$)
F4: Update(r_f, $\{R_f\}_{d_f}$, ID_f)
F5: Receive(S, M_{ReqSer}, $\{R_f, R_s, H(GK)\}_{d_{ser}}$)
F6: Verify($\{R_f, R_s, H(GK)\}_{d_{ser}}$, R_f, signature)

F7: Send(D, M_{ReqSer}, $\{R_f, R_s, H(GK)\}_{d_{ser}}$)
F8: Update(M_{ReqSer}, $\{R_f, R_s, H(GK)\}_{d_{ser}}$)
F9: Receive(D, $\{C_i^f, N_i^f\}$)
F10: $K_i^f \leftarrow r_f Q_i^f + d_f W_i$
F11: Verify(N_i^f)
F12: Decrypt($\{C_i^f\}$) using K_i^f
F13: Derive (GK) and (P_1, ..., P_n)
F14: Send(D, $P_1 ... P_n$)
F15: Update(GK, P_1, ..., P_n)
F16: Receive(D, W_i^n, W_i, s)
F17: $h \leftarrow H(GK, W_i^n, W_i)$
F18: Verify $sG \leftarrow W_i^n - hQ_i^f$
F19: Send(S, W_i^n, W_i, s)
F20: Update W_i^n

- Principal S

POSS(S): $\{ID_{ser}, d_{ser}, Q_{ser}, Q_{fi}, V_{fi}, Q_i, Q_f, Q_i^f, W_i, V_f\}$
BEL(S): $\{d_{ser}, \#r_s, Q_i^f, Q_f, Q_i, W_i\}$
BL(S) =
S1: Receive(F, M_{ReqFog}, ID_f, $\{R_f\}_{d_f}$)
S2: Verify(R_f, signature) using Q_f
S3: Generate-nonce(r_s)
S4: $R_s \leftarrow r_s G$
S5: $K_i \leftarrow r_s Q_i^f + d_{ser} W_i$
S6: $K_i' \leftarrow H(K_i || R_f)$
S7: $K_f \leftarrow r_s Q_f + d_{ser} R_f$
S8: $K_f' \leftarrow H(K_f)$
S9: Send(F, M_{ReqSer}, $\{R_f, R_s, H(GK)\}_{d_{ser}}$)
S10: Update(GK, K_f, K_i)
S11: Receive(F, W_i^n, W_i, s)
S12: $h \leftarrow H(GK, W_i^n, W_i)$
S13: Verify $sG \leftarrow W_i^n - hQ_i^f$
S14: Update W_i^n

- Principal D

POSS(D): $\{ID_i, d_i, Q_i, Q_{ser}, Q_f, d_i^f, Q_i^f, w_i, W_i\}$
BEL(D): $\{d_i, Q_i^f, \#w_i\, Q_{ser}, Q_f\}$
BL(D) =
D1: Receive(D, M_{ReqSer}, $\{R_f, R_s, H(GK)\}_{d_{ser}}$)
D2: Verify($\{R_f, R_s, H(GK)\}_{d_{ser}}$) using Q_{ser}
D3: Update($H(GK)$)
D4: $K_i \leftarrow d_i^f R_s + w_i Q_{ser}$
D5: $K_i^f \leftarrow d_i^f R_f + w_i Q_f$
D6: $C_i^f \leftarrow Encrypt(H(K_i || R_f))$ using K_i^f
D7: $N_i^f \leftarrow H(K_i^f)$
D8: Send(F, $\{C_i^f, N_i^f\}$)
D9: Update(K_i, K_i^f, C_i^f, N_i^f)
D10: Receive(F, $P_1 ... P_n$)
D11: Derive (GK)
D12: Verify $H(GK)$ with stored value
D13: $W_i^n \leftarrow w_i^n G$
D14: $h \leftarrow H(GK, W_i^n, W_i)$
D15: $s \leftarrow w_i^n - hd_i^f$
D16: Send(F, W_i^n, W_i, s)

2) THE PROTOCOL VERIFICATION

Following the group authentication and key agreement phase and the protocol specifications, the actions in BL(F) are executed first as the Fog (F) is initiator of the proposed scheme. The first action (F1) generates r_f; the third action (F3) sends M_{ReqFog}, ID_f, $\{R_f\}_{d_f}$ to the server (S). Next, (S1)–(S10) actions in BL(S) are performed to verify signature, generate nonce, and compute K_i, K_i', K_f, K_f'. The action S9 sends M_{ReqSer}, $\{R_f, R_s, H(GK)\}_{d_{ser}}$ to F and then S updates GK, K_f, and K_i in action S10. Nevertheless, the local sets of server (S) are changed as described below:

- POSS(S) = $\{ID_{ser}, d_{ser}, Q_{ser}, Q_{fj}, V_{fj}, Q_i, Q_i^f, W_i, K_f, K_i, GK\}$
- BEL(S) = $\{R_f, LINK(R_f)\}$

Now the global sets are updated as follows:

- Secret set: $\{GK, K_f, K_i\}$
- Observer sets:
  Observer(GK, K_f, K_i): {S}

After the action S10, F starts the actions in BL(F) with the received message, i.e., $M_{ReqSer} = \{R_f, R_s, H(GK)\}_{d_{ser}}$ from S. In action F6, it verifies signature and R_f. It then sends $M_{ReqSer} = \{R_f, R_s, H(GK)\}_{d_{ser}}$ to the device D, and updates own database with $M_{ReqSer} = \{R_f, R_s, H(GK)\}_{d_{ser}}$. Now, the local sets of F are changed as follows.

- POSS(F) = $\{R_f, R_s, H(GK)\}$
- BEL(F) = $\{R_f, R_s\}$

In this case, the global sets remain unchanged as there is no new secret generated or added by the F. Upon receiving M_{ReqSer}, $\{R_f, R_s, H(GK)\}_{d_{ser}}$, D starts the actions in BL(D). The actions (D1) and (D2), receives and verifies the message, respectively. If the signature is verified in D2 then it updates in action D3 and computes the values of K_i, K_i^f, C_i^f, N_i^f in D4 – D7 actions. The D sends C_i^f, N_i^f to F. The local sets of D are changed as follows.

- POSS(D) = $\{K_i^f, K_i, C_i^f, N_i^f, R_s, R_f, H(GK)\}$
- BEL(D) = $\{R_f, R_s, LINK(\{R_f, R_s, H(GK)\}_{d_{ser}}))\}$

Now the global sets are updated as follows:

- Secret set: $\{K_i^f, K_i\}$
- Observer sets:
  Observer(K_i^f): {D}
  Observer(K_i): {D,S}

Upon finishing the action (D9), the actions (F9)-(F15) are performed in BL(F) with incoming message C_i^f, N_i^f (i.e., F9). The actions decrypt C_i^f and verify N_i^f in F10 and F11, respectively. If verification fails then the system will be aborted, otherwise F goes on with the next action. It derives GK and $P_1 ... P_n$ and sends $P_1 ... P_n$ to D. However, the local sets of F are changed as follows.

- POSS(F) = $\{K_i^f, K_f, R_f, R_s, GK\}$
- BEL(F) = $\{R_f\}$

Now the global sets are updated as follows:

- Secret set: $\{K_i^f, K_f, GK\}$

- Observer sets:
  Observer(K_i^f): {F,D}
  Observer(K_f): {F,S}
  Observer(GK): {F,S,D}

Now, D executes few actions (D10–D16) in BL(D). It derives GK, checks $H(GK)$ with the stored value. It then chooses an updated random value w_i called w_i^n, performs the actions in D13 – D16 and sends W_i, W_i^n, s to F. The local sets of D are finally changed as follows.

- POSS(D) = $\{GK, W_i, W_i^n, w_i^n, h, s\}$
- BEL(D) = $\{GK, W_i, W_i^n, w_i^n, h, s\}$

Now the global sets are updated as follows:

- Secret set: $\{GK, w_i^n, h\}$
- Observer sets:
  Observer(GK): {D,S}
  Observer(w_i^n): {D}
  Observer(h): {F}

Upon receiving message in action F16, F runs following actions (F16 – F20) in BL(F). It computes h verifies sG and if this verification is successful F sends W_i, W_i^n, s to S. Finally, F updates its own table. The local sets of F are changed as follows.

- POSS(F) = $\{GK, W_i^n, w_i^n, h, s\}$
- BEL(F) = $\{GK, W_i^n, w_i^n, h, s\}$

Now the global sets are updated as follows:

- Secret set: $\{GK, h\}$
- Observer sets:
  Observer(GK): {F,D,S}
  Observer(h): {F,D}

After receiving message in action S11, S performs few actions (S12 –S14) in BL(S). In S12, it computes h verifies sG and if this verification is successful, S finally updates its own database. The local sets of S are changed as follows.

- POSS(S) = $\{GK, W_i^n, w_i^n, h, s\}$
- BEL(S) = $\{GK, W_i^n, w_i^n, h, s\}$

Now the global sets are updated as follows:

- Secret set: $\{GK, h\}$
- Observer sets:
  Observer(GK): {F,D,S}
  Observer(h): {F,D,S}

This result implies that:

- r_f, r_s and w_i are fresh for each session, and are known by the legitimate F, S and Ds.
- The group key GK is only possessed by the legitimate entities, e.g., F, S, and Ds.
- F, S, and D are mutually verified and authenticated through the signatures during the protocol execution.
- h can only be computed by the legitimate entities F, S, and Ds.

This verifies the security claims for the proposed scheme.

VI. SECURITY ANALYSIS

We start with an informal analysis. For that, we need to verify if all requirements, stated in section III-C, are satisfied.

- Authentication:
  First in the initialisation phase, the device needs to be ensured that it is connected to a legitimated fog, the fog needs the guarantee that the device is legitimate, and the server is required to know that it is connected to the device claiming to be the one with that particular public key. We consider both the static and dynamic topology.
  Static topology: With the usage of the public key of the server, the device is able to send securely its public key to the server. Since it includes a hash value including an instance of the ECDHP, only the server can indeed validate that the message is coming from the device claiming to possess that public key. By sending the credentials of the fog and including the result of the new public key in the hash function, the device is ensured that the credentials are correct since only the server (and the device) is able to generate this hash value. The fog is ensured about the new device, since it is informed by the server through a message signed by its private key.
  Dynamic topology: Since the device is in possession of the public key of the server, it can securely initialise the procedure with the server, using the broadcast data that it received from the fog. However, the device still needs confirmation of the server that this data is correct. When receiving the message by the device, the server is the only entity able to retrieve the public key Q_i of the device due to the ECDHP and to compute the corresponding new public key Q_i^f. Again, as it is the only entity in possession of v_f, the server is the only one able to derive this key due to ECDHP. After verifying the hash included in the encrypted message, it can assure the device that it is connected to a legitimate fog. In order to do that, it sends Q_i^f, TS to the fog, encrypted with a key only derivable by the fog due to the ECDHP. As a consequence, the fog is ensured that Q_i^f is coming from a valid request since it was part of the original message sent by the device in the form of a hash value. Also the device has the guarantee that fog, server and members in the network of the fog are aware of its correct presence, as only the server was able to derive the correct public key.
  The next step is the update request. As both the fog and server are sending the message together with a signature on it, authentication is clear. The authentication of the devices in sending their response during the response phase is also evident due to the inclusion of the hash value and the ECDHP.
  The authenticity in the response of the fog can be verified by the devices after computing the resulting group key. If the hash of this group key corresponds with the part of the message sent in the original update request of the server, the devices are ensured that the intended fog is included in the process.
  Finally, as in the acknowledgement the unique pairwise keys are used, the fog and server have the guarantee that they are coming from the intended device.

Note that this feature guarantees protection against man-in-the-middle attacks and impersonation attacks. Moreover, also replay attacks are impossible, due to the usage of unique random values and timestamps.

- Integrity:
  The integrity of the messages is guaranteed due to the fact that hash values are included at all critical points in the protocol in order to detect at an early stage potential problems. Often, a hash value is sent beforehand (e.g. new public key in initialisation during dynamic topology, group key), which is then later verified after receiving additional information.
- Anonymity and unlinkability:
  Anonymity follows from the fact that each time a device is connected to the fog, it generates a new public key, which is unlinkable to the original [53]. In the initialisation for the dynamic topology, the ECDHP is exploited in order to share in an anonymous way the identity of the entity.
- Group forward secrecy: If one of the long-term private keys of the devices or fog is leaked, access is no longer allowed to group communication content.
  When an attacker is collecting the points $(P_1, ..., P_n)$, in order to form the group key from it, an additional point is required. Therefore, either the combination of private and random local variable (d_i^f, w_i) of one of the devices, or the combination (d_f, r_f) of the fog can be used. Since, these random values are temporarily stored in devices and fog, even with leakage of the private keys, the group keys cannot be derived.
- Backward Confidentiality: Devices who join the group cannot get the former group keys from its present key.
  When a device joins later on, its key material is not yet included in the previous group keys. Moreover, during the construction and the acknowledgement, the group key is only sent under hash construction, and thus is not possible to retrieve in case of a collision resistant hash function.
- Perfect forward secrecy: Previous group keys should be still secure even if the long-term secrets are compromised by the attacker.
  As explained before in the group forward secrecy scheme, even if all private keys of the devices and the fog are leaked, the scheme is still secure as it heavily relies on the random and temporarily stored values $(w_i)_i, r_f$.
- By the end of the process, both fog and server are ensured that all entities of the group possess the required group key.
  This follows from the acknowledgement phase.
- The devices in the group are ensured that the fog has a secure relation with the server.
  This follows from the fact that the hash of the group key, originally present in the first message of the server, and actually signed by the server, corresponds with the group key computed based on the information sent by the fog. If the fog had not a secure relation with the server, it would not have been able to derive the required material for the construction of the group key.
- By the end of the process, both a group key among all entities and a common pairwise key between device and fog, between device and server, and between fog and server, are derived.
  The pairwise key between device and fog equals to K_i^f, between device and server to K_i, and between fog and server to K_f. These keys are only known to the involved entities due to the ECDHP.

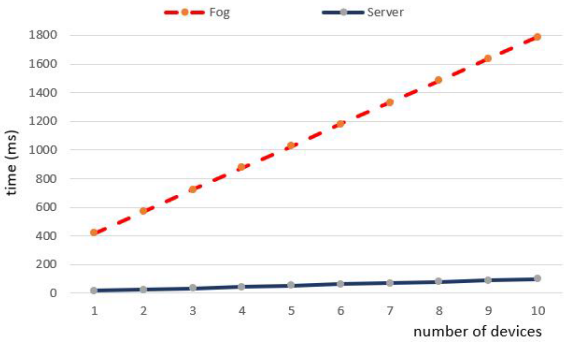

FIGURE 6. Computational complexity of the Authentication and Key Agreement phase for fog and server with *n* IoT devices.

TABLE 4. Communication costs of different messages.

Message	Size (bits
Initialisation	1925
M_1	805
M_2	1286
M_3	514
M_4	$128n$
M_5	$770n$
M_6	$770n$
Total (excl. initialisation)	$2605+1668n$

VII. PERFORMANCE

We here consider both performance from computational and communication point of view.

A. COMPUTATIONAL COSTS

The computational costs are dominated by the most compute intensive operations, being EC multiplication, EC additions, encryption and hash, denoted by T_m, T_a, T_e, T_h. Note that we aim for 128-bit security and use the appropriate parameters for that in the choice of the EC, the hash function SHA256 and AES128. The performance of these operations are evaluated on three different platforms, corresponding to the roles of edge devices, fog and server. To be more concrete, the Zolertia RE-mote ARM Cortex-M3 @ 32MHz, 32KB RAM is used as edge device, the Raspberry Pi 3B Quad Core 1.2GHz, 1GB RAM as fog, and a Personal Computer Intel Core i7-8750H CPU @ 2.2GHz, 16GB RAM as server. Table 2 provides the timings for these four operations using the specified libraries.

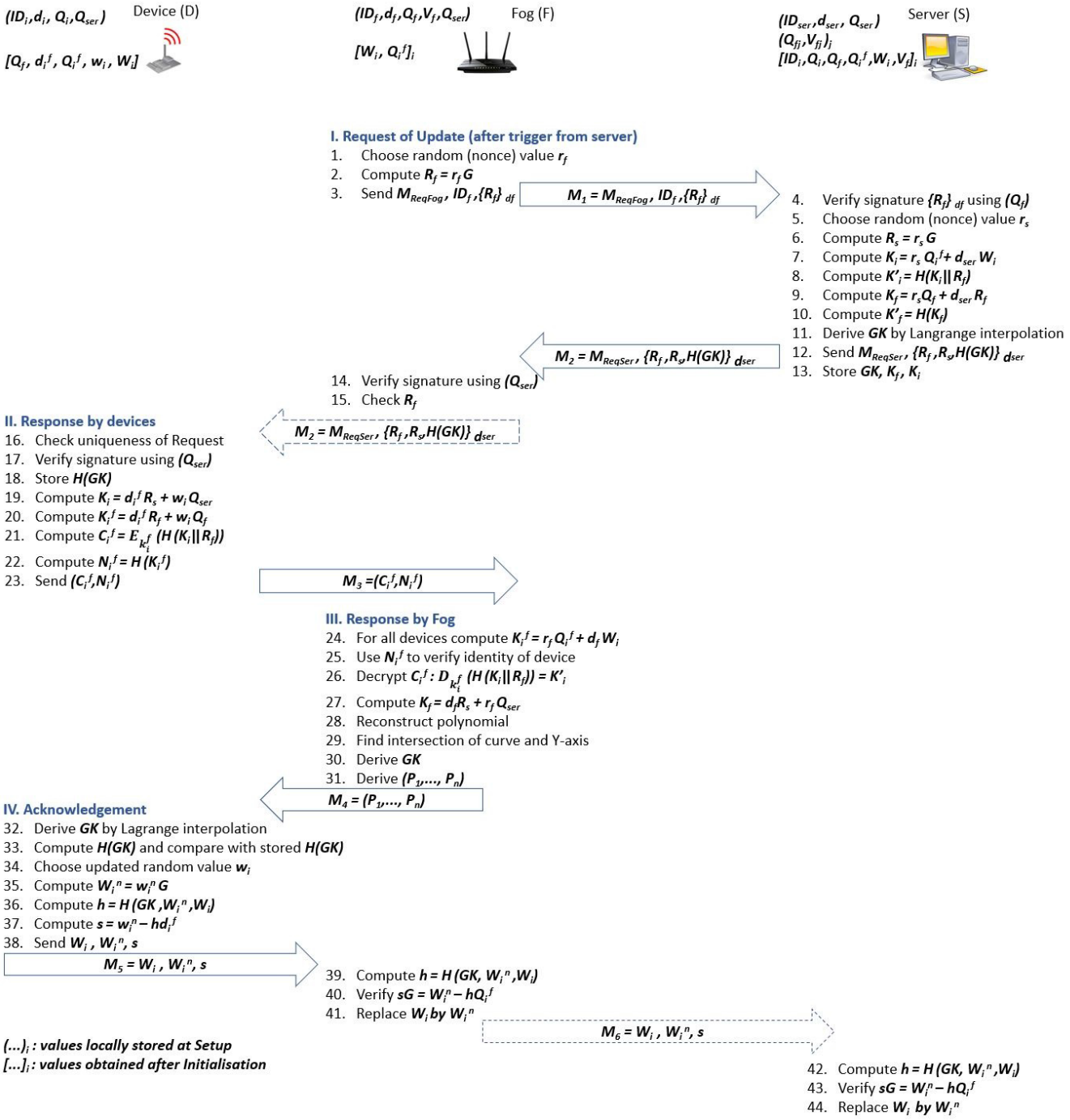

FIGURE 7. Group Authentication and Key Agreement Phase.

Table 3 provides an overview of the required cryptographic operations for the different phases in the scheme. As there are no other group authentication schemes for such fog architecture, we give as information the performance of the scheme in [43], proposing an anonymous pairwise mutual authentication scheme for a fog architecture. If we focus on the most constrained device, being the edge, we can conclude that our scheme is in the same range as [43]. As our scheme is a group authentication scheme, the complexity at fog and cloud server is higher in our scheme. However, if n=1, the complexity becomes more or less similar to [43].

Figure 6 illustrates the computational complexity for the fog and the server, using the measurements of Table 3 for a varying number of devices. The computational cost for the fog goes from 418 ms with one device to 1789 ms for ten devices to execute the group authentication and key

TABLE 2. The average time and 99.9% confidence interval of the EC point multiplication, EC point addition, AES_128_CCM_8 symmetric encryption/decryption and SHA256 hash function for the Zolertia RE-mote, Raspberry PI 3B and personal computer.

Device	Library	EC_mult(μs)	EC_add(μs)	AES(μs)	SHA256(μs)
Zolertia RE-mote ARM Cortex-M3 @ 32 MHz, 32 KB RAM	tinyDTLS microECC	993917 ± 7	17262 ± 8	2024 ± 7	1047 ± 7
Zolertia RE-mote ARM Cortex-M3 @ 32 MHz, 32 KB RAM	Hardware Acceleration Engine	344659 ± 7	5080 ± 8	150 ± 7	68 ± 7
Raspberry PI 3B Quad Core 1.2GHz, 1GB RAM.	BouncyCastle	37943 ± 85	181.7 ± 0.5	60.47 ± 0.07	15.50 ± 0.02
Personal Computer Intel Core i7-8750H CPU @ 2.2GHz, 16GB RAM.	BouncyCastle	1148.2 ± 0.8	4.866 ± 0.005	2.90 ± 0.03	1.023 ± 0.002

TABLE 3. The cryptographic operations that device, fog and server need to perform for a group of *n* devices.

Phase	Device	Fog	Server
Static initialisation	$5T_m + 1T_e + 3T_h$	$1T_m + 1T_e + 2T_h$	$4T_m + 1T_a + 2T_e + 4T_h$
Dynamic initialisation	$5T_m + 1T_e + 3T_h$	$1T_m + 1T_e + 2T_h$	$4T_m + 1T_a + 2T_e + 3T_h$
Group key construction	$7T_m + 3T_a + 1T_e + 5T_h$	$(4n+6)T_m + (2n+2)T_a + nT_e + (2n+2)T_h$	$(4n+6)T_m + (2n+2)T_a + (2n+4)T_h$
Total for n devices	–	$(4n+7)T_m + (2n+2)T_a + 2nT_e + (3n+2)T_h$	$(8n+6)T_m + (3n+2)T_a + 2nT_e + (6n+4)T_h$
[43]	$7T_m + 2T_a + 2T_e + 12T_h$	$7T_m + 2T_a + 2T_e + 13T_h$	$9T_m + 4T_a + 4T_s + 13T_h$

agreement. For the server, it will take between 16 ms for one device up to 99 ms for ten devices. We consider these costs to be reasonable. For the scalability of the system, we assume that in cases where more devices are present, there should be a more powerful fog or the number of fogs should be increased to keep the delay acceptable.

B. COMMUNICATION COSTS

For the communication, we measure the size of the messages, both in the initialisation phase and in the group authentication phase. We assume the length of the TS to be equal to 32 bits, the result of the SHA256 equals to 256 bits, an EC point transmitted using 257 bits and a message request update of 4 bits. For the initialisation phase, both in the static and dynamic topology, we need a total of 1925 bits. For the group authentication phase, the total required number of bits to be sent equals to $2605 + 1668n$, with n the number of devices in the group. Compared with [43], where a total of 3264 bits is required, this amount is certainly acceptable. A detailed calculation of these numbers can be found in Table 4. Note that the content of $M1$, ..., $M6$ are shown in Figure 7.

VIII. CONCLUSION

In this paper, we have proposed an efficient and novel authentication and key agreement mechanism for a set of IoT devices handled by a fog and managed by a server. In particular, the fog is considered to be potentially vulnerable to attacks and requires special attention in order not to leak sensitive information. After an initialization phase which is either static or dynamic, the authentication is carried on using the participants' public key. Finally, a group key is derived by all parties (devices, fog and server). We have used Rubin logic to show how throughout the execution of the protocol, no outsider is capable to derive any information on the secret key. The protocol also ensures authentication, integrity, anonymity and unlinkability, group forward secrecy, backward confidentiality, and perfect forward secrecy. As we believe that the fog architecture will become increasingly popular for its advantages, the proposed mechanisms can be used for authentication and key agreement purposes dealing with group authentication in a multitude of application domains like smart grid, vehicular networks, smart health care systems, smart cities, etc.

REFERENCES

[1] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st MCC Workshop Mobile Cloud Comput.*, Aug. 2012, pp. 13–16.

[2] J. Ni, K. Zhang, X. Lin, and X. S. Shen, "Securing fog computing for Internet of Things applications: Challenges and solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 601–628, 1st Quart., 2018.

[3] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges," *Future Gener. Comput. Syst.*, vol. 78, pp. 680–698, Jan. 2018.

[4] M. Chiang and T. Zhang, "Fog and IoT: An overview of research opportunities," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 854–864, Dec. 2016.

[5] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proc. Workshop Mobile Big Data*, Jun. 2015, pp. 37–42.

[6] H. Wadhwa and R. Aron, "Fog computing with the integration of Internet of Things: Architecture, applications and future directions," in *Proc. IEEE Int. Conf. Parallel Distrib. Process. Appl., Ubiquitous Comput. Commun., Big Data Cloud Comput., Social Comput. Netw., Sustain. Comput. Commun.*, Dec. 2018, pp. 987–994.

[7] W. Tang, K. Zhang, D. Zhang, J. Ren, Y. Zhang, and X. S. Shen, "Fog-enabled smart health: Toward cooperative and secure healthcare service provision," *IEEE Commun. Mag.*, vol. 57, no. 5, pp. 42–48, May 2019.

[8] M. Ahmad, M. Bilal, S. Hussain, B. Ho, T. Cheong, and S. Lee, "Health fog: A novel framework for health and wellness applications," *J. Supercomput.*, vol. 72, no. 10, pp. 3677–3695, 2016.
[9] I. Stojmenovic and S. Wen, "The Fog computing paradigm: Scenarios and security issues," in *Proc. Federated Conf. Comput. Sci. Inf. Syst.*, Sep. 2014, pp. 1–8.
[10] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, and Q. Yang, "A hierarchical distributed fog computing architecture for big data analysis in smart cities," in *Proc. ASE BigData Social Inform.*, Oct. 2015, Art. no. 28.
[11] M. Khabazian, S. Aissa, and M. Mehmet-Ali, "Performance modeling of message dissemination in vehicular ad hoc networks with priority," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 1, pp. 61–71, Jan. 2011.
[12] J. A. Onieva, R. Rios, R. Roman, and J. Lopez, "Edge-assisted vehicular networks security," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8038–8045, Oct. 2019. doi: 10.1109/JIOT.2019.2904323.
[13] M. El-Hajj, A. Fadlallah, M. Chamoun, and A. Serhrouchni, "A survey of Internet of Things (IoT) authentication schemes," *Sensors*, vol. 19, no. 5, p. 1141, Mar. 2019.
[14] P. Kumar, A. Braeken, A. Gurtov, J. Iinatti, and P. H. Ha, "Anonymous secure framework in connected smart home environments," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 4, pp. 968–979, Apr. 2017.
[15] A. Braeken and A. Touhafi, "Efficient anonymous user authentication on server without secure channel during registration," in *Proc. 2nd Conf. Cloud Comput. Technol. Appl. (CloudTech)*, May 2016, pp. 13–20.
[16] A. Braeken, P. Kumar, M. Liyanage, and T. T. K. Hue, "An efficient anonymous authentication protocol in multiple server communication networks (EAAM)," *J. Supercomput.*, vol. 74, no. 4, pp. 1695–1714, 2018.
[17] V. Odelu, A. K. Das, M. Wazid, and M. Conti, "Provably secure authenticated key agreement scheme for smart grid," *IEEE Trans. Smart Grid*, vol. 9, no. 3, pp. 1900–1910, May 2018.
[18] Y. Chen, J. G. Martínez, P. Castillejo, and L. López, "An anonymous authentication and key establish scheme for smart grid: FAuth," *Energies*, vol. 10, no. 9, p. 1345, 2017.
[19] J.-L. Tsai and N.-W. Lo, "Secure anonymous key distribution scheme for smart grid," *IEEE Trans. Smart Grid*, vol. 7, no. 2, pp. 906–914, Mar. 2016.
[20] D. Wu and C. Zhou, "Fault-tolerant and scalable key management for smart grid," *IEEE Trans. Smart Grid*, vol. 2, no. 2, pp. 375–381, Jun. 2011.
[21] J. Xia and Y. Wang, "Secure key distribution for the smart grid," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1437–1443, Sep. 2012.
[22] S. Zhu, S. Setia, and S. Jajodia, "LEAP+: Efficient security mechanisms for large-scale distributed sensor networks," in *Proc. 10th ACM Conf. Comput. Commun. Secur. (CCS)*, 2003, vol. 2, no. 4, pp. 500–528.
[23] A. Durresi, V. Bulusu, V. Paruchuri, M. Durresi, and R. Jain, "WSN09-4: Key distribution in mobile heterogeneous sensor networks," in *Proc. IEEE Globecom*, Nov./Dec. 2006, pp. 1–5.
[24] S. Hussain, F. Kausar, and A. Masood, "An efficient key distribution scheme for heterogeneous sensor networks," in *Proc. Int. Conf. Wireless Commun. Mobile Computing (IWCMC)*, New York, NY, USA, Aug. 2007, pp. 388–392.
[25] G. Ou, J. Huang, and J. Li, "A key-chain based key management scheme for heterogeneous sensor network," in *Proc. IEEE Int. Conf. Inf. Theory Inf. Secur.*, Dec. 2010, pp. 358–361.
[26] Y. Zhang, Y. Shen, and S. Lee, "A cluster-based group key management scheme for wireless sensor networks," in *Proc. 12th Int. Asia–Pacific Web Conf.*, Apr. 2010, pp. 3–5.
[27] L. Li and X. Wang, "A high security dynamic secret key management scheme for wireless sensor networks," in *Proc. 3rd Int. Symp. Intell. Inf. Technol. Secur. Inform.*, Apr. 2010, pp. 507–510.
[28] K. N. Shnaikat and A. A. Alqudah, "Key management techniques in wireless sensor networks," in *Proc. Int. J. Netw. Secur. Appl. (IJNSA)*, Nov. 2014, pp. 49–63.
[29] M. Carlier, K. Steenhaut, and A. Braeken, "Symmetric-key-based security for multicast communication in wireless sensor networks," *Computers*, vol. 8, no. 1, p. 27, 2019.
[30] K. Ren, W. Lou, B. Zhu, and S. Jajodia, "Secure and efficient multicast in wireless sensor networks allowing ad hoc group formation," *IEEE Trans. Veh. Technol.*, vol. 58, no. 4, pp. 2018–2029, May 2009.
[31] L. Q. Chen, C. F. Sun, and Q. Y. Zhu, "A novel group key agreement scheme for wireless sensor networks based on Merkle identity tree," *Adv. Mater. Res.*, vols. 846–847, pp. 869–875, Nov. 2013.
[32] W. Yao, S. Han, and X. Li, "LKH++ based group key management scheme for wireless sensor network," *Wireless Pers. Commun.*, vol. 83, no. 4, pp. 3057–3073, Aug. 2015.
[33] P. Porambage, A. Braeken, C. Schmitt, A. Gurtov, M. Ylianttila, and B. Stiller, "Group key establishment for enabling secure multicast communication in wireless sensor networks deployed for IoT applications," *IEEE Access*, vol. 3, pp. 1503–1511, 2015.
[34] L. Gong, "Lower bounds on messages and rounds for network authentication Protocols," in *Proc. 1st ACM Conf. Comput. Commun. Secur.*, Nov. 1993, pp. 26–37.
[35] C.-C. Lee, S.-D. Chen, and C.-L. Chen, "A computation-efficient three-party encrypted key exchange protocol," *Appl. Math. Inf. Sci.*, vol. 6, no. 3, pp. 573–579, 2012.
[36] X. Li, J. Niu, S. Kumari, M. K. Khan, L. Liao, and W. Liang, "Design and analysis of a chaotic maps-based three-party authenticated key agreement protocol," *Nonlinear Dyn.*, vol. 80, no. 3, pp. 1209–1220, 2015.
[37] T.-F. Lee and T. Hwang, "Three-party authenticated key agreements for optimal communication," *PloS ONE*, vol. 12, no. 3, 2017, Art. no. e0174473.
[38] H. A. Al Hamid, S. M. M. Rahman, M. S. Hossain, A. Almogren, and A. Alamri, "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*, vol. 5, pp. 22313–22328, 2017.
[39] X. Jia, D. He, N. Kumar, and K.-K. R. Choo, "Authenticated key agreement scheme for fog-driven IoT healthcare system," in *Wireless Networks*. New York, NY, USA: Springer, 2018, pp. 1–14.
[40] C.-L. Liu, Q.-J. Tsai, T.-Y. Chang, and T.-M. Liu, "Ephemeral-secret-leakage secure id-based three-party authenticated key agreement protocol for mobile distributed computing environments," *Symmetry*, vol. 10, no. 4, p. 84, 2018.
[41] D. Wang and P. Wang, "On the anonymity of two-factor authentication schemes for wireless sensor networks: Attacks, principle and solutions," *Comput. Netw.*, vol. 73, pp. 41–57, Jul. 2014.
[42] C.-L. Liu, W.-J. Tsai, T.-Y. Chang, and T.-M. Liu, "Ephemeral-secret-leakage secure ID-based three-party authenticated key agreement protocol for mobile distributed computing environments," *Symmetry* vol. 10, no. 4, p. 84, 2018.
[43] S. Patonico, A. Braeken, and K. Steenhaut, "Identity-based and anonymous key agreement protocol for fog computing resistant in the Canetti–Krawczyk security model," in *Wireless Networks*. New York, NY, USA: Springer, Jul. 2019, pp. 1–13.
[44] R. Canetti and H. Krawczyk, *Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels*. Berlin, Germany: Springer, 2001, pp. 453–474.
[45] J. Shen, A. Wang, L. Yan, Y. Ren, and Q. Liu, "Identity-based group devices authentication scheme for Internet of Things," in *Proc. 11th EAI Int. Conf. Mobile Multimedia Commun.*, MOBIMEDIA, China, Jun. 2018.
[46] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*. New York, NY, USA: Springer-Verlag, 2003.
[47] *SEC 2: Recommended Elliptic Curve Domain Parameters*, SECG, Consortium Certicom, Canada, Sep. 2000. [Online]. Available: http://www.secg.org/collateral/sec2 final.pdf
[48] *Recommended Elliptic Curves for Federal Government Use*, Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Aug. 1999. [Online]. Available: http://csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf
[49] D. R. L. Brown. (2009). *Certicom Research, SEC 1—Standards for Efficient Cryptography Group*. [Online]. Available: http://www.secg.org/sec1-v2.pdf
[50] C. P. Schnorr, "Efficient identification and signatures for smart cards," in *Advances in Cryptology—CRYPTO* (Lecture Notes in Computer Science), vol. 435, G. Brassard, Ed. New York, NY, USA: Springer, 1989.
[51] M. Wazid, A. K. Das, R. Hussain, G. Succi, and J. J. P. C. Rodrigues, "Authentication in cloud-driven IoT-based big data environment: Survey and outlook," *J. Syst. Archit.*, vol. 97, pp. 185–196, Aug. 2018. doi: 10.1016/j.sysarc.2018.12.005.
[52] M. Qu and S. A. Vanstone, "Implicit certificate scheme," U.S. Patent 6 792 530, Sep. 14, 2004.
[53] N. van Saberhagen. *CryptoNote V2.0*. Accessed: Oct. 2013. [Online]. Available: https://cryptonote.org/whitepaper.pdf
[54] A. D. Rubin and P. Honeyman, "Nonmonotonic cryptographic protocols," in *Proc. 7th Comput. Secur. Found. Workshop*, Franconia, NH, USA, Jun. 1994, pp. 100–116.
[55] P. Kumar, A. J. Choudhury, M. Sain, S.-G. Lee, and H.-J. Lee, "RUASN: A robust user authentication framework for wireless sensor networks," *Sensors*, vol. 11, no. 5, pp. 5020–5046, 2011.

[56] A. Braeken, M. Liyanage, P. Kumar, and J. Murphy, "Novel 5G authentication protocol to improve the resistance against active attacks and malicious serving networks," *IEEE Access*, vol. 7, pp. 64040–64052, 2019.

PLACIDE SHABISHA received the M.Sc. degree in communication networks from the University of Sidi Bel Abbes, Algeria, in 2012. He is currently pursuing the Ph.D. degree in engineering science with Vrije Universiteit Brussel, Belgium, with a scholarship from the VLIR-UOS project: IUC 2017 Phase 3 UB. Since 2013, he has been an Assistant Lecturer with the Department of Information and Communications Technology, University of Burundi. His research interests include the Internet of Things (IoT), cloud computing, elliptic curve cryptography, and computer and network security.

AN BRAEKEN received the M.Sc. degree in mathematics from Gent University, in 2002, and the Ph.D. degree in engineering science from the research group Computer Security and Industrial Cryptography (COSIC), KU Leuven, in 2006. She worked for almost two years at the management consulting company Boston Consulting Group (BCG). She became a Professor, in 2007, at the Erasmushogeschool Brussel (since 2013, Vrije Universiteit Brussel), Department of Engineering Technology. She is a (co)author of over 120 publications. Her current interests include security and privacy protocols for the IoT, cloud and fog, blockchain, and 5G security. She has been a member of the program committee for numerous conferences and workshops (IOP2018, EUC 2018, ICNS 2018, and so on) and a member of the Editorial Board for *Security and Communications Magazine*. She has been a member of the Organizing Committee of the IEEE Cloudtech 2018 Conference and the Blockchain in IoT Workshop at Globecom 2018. Since 2015, she has been a 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 an STSM Manager in the COST AAPELE project, from 2014 to 2017, where she is currently on the Management Committee of the COST RECODIS project (2016–2019).

PARDEEP KUMAR (M'13) received the B.E. degree in computer science from Maharshi Dayanand University, Haryana, India, in 2002, the M.Tech. degree in computer science from Chaudhary Devi Lal University, Haryana, in 2006, and the Ph.D. degree in ubiquitous computing from Dongseo University, Busan, South Korea, in 2012. From 2012 to 2018, he held postdoctoral positions at the Department of Computer Science, Oxford University, Oxford, U.K., from August 2016 to September 2018, the Department of Computer Science, The Arctic University of Norway, Tromsø, Norway, from August 2015 to August 2016, and the Centre for Wireless Communications and the Department of Communications Engineering, University of Oulu, Finland, from April 2012 to August 2015. He is currently a Lecturer/Assistant Professor with the Department of Computer Science, Swansea University, Swansea, U.K. His research interests include security in sensor networks, smart environments, cyber physical systems, the Internet of Things, and 5G networks.

KRIS STEENHAUT received the master's degrees in engineering science and applied computer science, and the Ph.D. degree in engineering science from Vrije Universiteit Brussel (VUB), in 1984, 1986, and 1995, respectively. She is currently a Professor with the Department of Electronics and Informatics (ETRO) and the Department of Engineering Technology (INDI), Faculty of Engineering Sciences, VUB. She has cooperated in and coordinated European ITEA Consortia, such as the European Sensor Network Architecture (ESNA) and Interoperable Sensor Networks (ISN) and participated in FP7 under the ICT Policy Support Program on the theme, ICT for a low carbon economy and smart mobility, within the EDISON project on smart lighting. Her research interests include the design, implementation, and evaluation of wireless sensor networks for building automation, environmental monitoring, and smart grids.