Octopus: An Edge-Fog Mutual Authentication Scheme

Maged Hamada Ibrahim
(Corresponding author: Maged Hamada Ibrahim)

Department of Electronics, Communications and Computers Engineering, Helwan University
1, Sherif St., Helwan, P.O. Box 11792, Cairo, Egypt
(Email: mhii72@gmail.com)

(Received Sept. 1, 2015; revised and accepted Dec. 7 & Dec. 15, 2015)

Abstract

Authentication is an important and challenging issue for the security of Fog computing since, services are offered to massive-scale end users (Fog users or Edge) by front Fog servers (or Fog nodes). In this paper, we propose a secure and efficient mutual authentication scheme for the Edge-Fog-Cloud network architecture, to mutually authenticate Fog users at the Edge of the network, with the Fog servers at the Fog layer. Our scheme requires a user – roaming randomly in the network – to hold only one long-lived master secret key (with long enough bit-length) allowing him to communicate with any of the Fog servers in the network, in a fully authenticated way. The Fog users are able to mutually authenticate with new Fog servers joining the network, without the need to re-register and without any extra overheads. Moreover, the servers in the Fog are required to store only one secret key for each Fog user. On the other hand, the Fog users are totally unrelated to any public-key infrastructure. The scheme requires the Fog user to perform very few hash invocations and symmetric encryptions/decryptions. Therefore, the scheme is suitable to be efficiently implemented on the Fog user's smart card/device.

Keywords: Cloud computing, Edge layer, Fog layer, mutual authentication, rogue nodes, smart cards

1 Introduction

Through the last decade, Cloud computing has provided many opportunities for enterprises, by offering their customers a range of computing services. "Pay-as-you-go" Cloud computing model becomes an efficient alternative to owning and managing private data centers for customers facing web applications and batch processing [1, 33]. Cloud computing frees the enterprises and their end users from the specification of many details, such as storage resources, computation limitation and network communication cost. However, this felicity becomes a problem for latency-sensitive applications, which require nodes in the vicinity to meet their delay requirements [3]. When techniques and devices of IoT are getting more involved in people's life, with millions of such devices acquiring services, current Cloud computing architecture can hardly satisfy their requirements of mobility support, location awareness and low latency [33].

The Fog is a layer intermediate between the end users (Edge of the Network) and the Cloud, to bring far in cyberspace Cloud services to close proximity to the Edge and on a wider range. In Fog computing, services can be hosted at end devices such as, access points as illustrated in Figure 1. The infrastructure of this new multi-layered distributed computing allows applications to run as close as possible to sensed actionable and massive data coming out of people, processes and thing. Both Cloud and Fog provide data, computation, storage and application services to the Edge. However, Fog can be distinguished from Cloud by its proximity to end-users, the dense geographical distribution and its support for mobility. There exists wide range of applications for the Fog services, some of which are described next:

- *Malls*: Assuming that a number of Fog servers are deployed inside a multi-floor shopping center, which collectively form an integrated localized information system. The Fog servers at different floors can pre-cache floor-related contents, such as the layout, offers, ads, goods prices, etc. of stores on a particular floor. The Fog servers can deliver engaged services including indoor navigation, ads distribution and feedback collections to mobile users through, for example, WiFi.

- *Airports/Park zones*: The Fog computing system can be deployed in the parkland/zones to provide localized travel services. For instance, Fog servers can be deployed at the entrance and other important locations of the park. The Fog server at the entrance can pre-cache information including park map, park free slots, travel guide and local accommodations; other

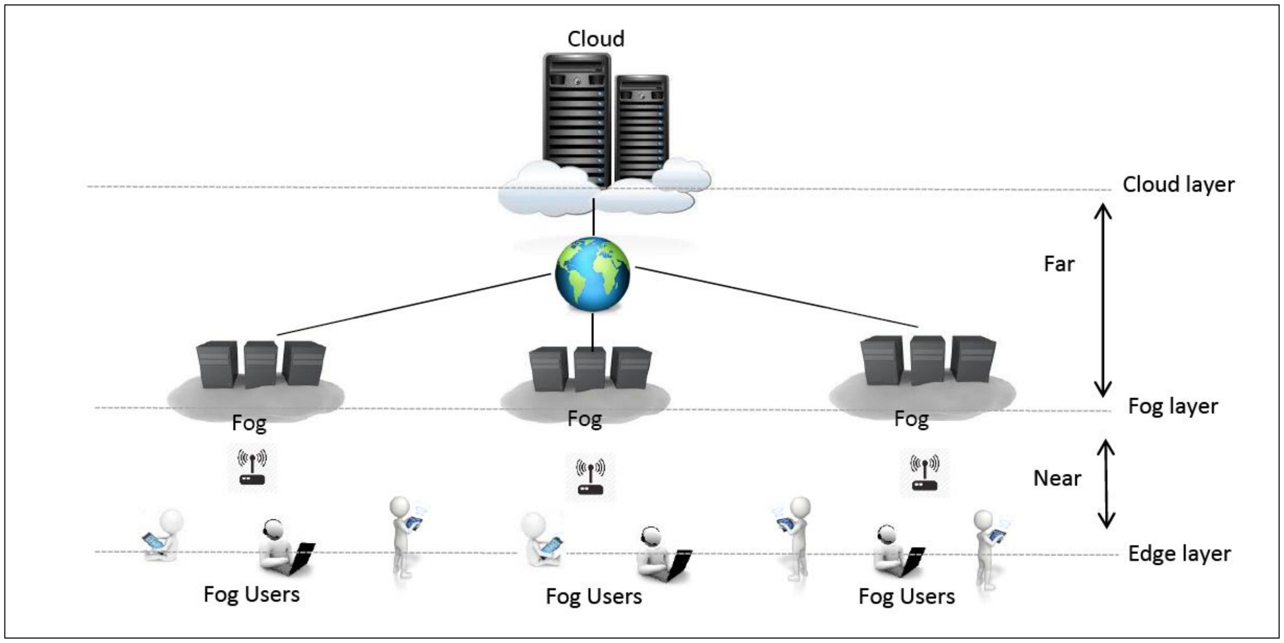

Figure 1: Edge-Fog-Cloud Architecture

Fog servers at different locations inside the park can be incorporated with sensor networks for environment monitoring and provide navigation to travelers. By connecting the Fog servers to the park administration office and Cloud, the Fog servers can be used as an information gateway to send timely alerts, notifications and information to travelers. Such services apply also to airports such as flight dates, delays, airport shops products and offers, medical services, etc.

- *Smart grid*: Energy load balancing applications may run on network Edge devices, such as smart meters and micro-grids [39]. Based on energy demand, availability and the lowest price, these devices automatically switch to alternative energies like solar and wind. Fog collectors at the Edge process the data generated by grid sensors and devices, and issue control commands to the actuators [3]. They also filter the data to be consumed locally, and send the rest to the higher tiers for visualization, real-time reports and transactional analytics. Fog supports ephemeral storage at the lowest tier to semi-permanent storage at the highest tier. Global coverage is provided by the Cloud with business intelligence analytics.

- *Transportation*: A Fog server can be deployed inside the bus and provides onboard video streaming, gaming and social networking services to travelers using WiFi. The on-board Fog server connects to the Cloud through cellular networks to refresh the pre-catched contents and update application services. Using its computing facility, the Fog server can also collect and process user's data, such as number of travelers and their feedbacks, and reports to Cloud [3, 33].

In Cloud computing deployment, data centers are usually owned by Cloud service providers. However, Fog service providers can be different parties, due to different deployment choices: Internet service providers or wireless carriers (e.g. GSM), who have control of home gateways or cellular base stations, may build Fog with their existing infrastructures. Cloud service providers, who want to expand their Cloud services to the Edge of the network, may also build Fog infrastructures. End users, who own a local private Cloud and want to reduce the cost of ownership, would like to turn the local private Cloud into Fog and lease spare resources on the local private Cloud. This flexibility complicates the trust situation of Fog and makes it different from other network architectures.

As a consequence of the absence of authentication services, a rogue Fog node/server would be a Fog device or Fog instance that pretends to be legitimate and masquerade Fog nodes for Edge users to connect to it. For example, in an insider attack, a Fog administrator may be authorized to manage Fog instances, but may instantiate a rogue Fog instance rather than a legitimate one. The discussion in [33] has demonstrated the feasibility of man-in-the-middle attack in Fog computing, before which the gateway should be either compromised or replaced by a Fake one. Once connected, the adversary can manipulate the incoming and outgoing requests from end users or Cloud, collect or tamper user data stealthily, and easily launch further attacks. The existence of fake Fog node or server is a serious threat to user data security and privacy.

2 Related Work

Security and privacy issues were not studied to directly hit the requirements of Fog computing. Some studies were in the context of smart grids [38] and machine-to-machine communications [23]. There are security solutions for Cloud computing. However, they are not suitable for Fog computing because Fog devices work at the Edge of networks on a larger and wider scale. The environment of Fog devices is faced with many threats which do not exist in well managed Cloud. We next discuss the contributions we found closely related to Fog computing.

Reputation based trust model [16] has been applied successfully in e-Commerce, peer-to-peer, user reviews and online social networks. The work in [7] proposed a robust reputation system for resource selection in peer-to-peer networks using a distributed polling algorithm to assess the reliability of a resource before downloading. In designing a Fog computing still some problems are not solved; *How to achieve persistent, unique, and distinct identity? How to treat intentional and accidental misbehavior? How to conduct punishment and redemption of reputation?* There are also trusting models based on special hardware such as Secure Element (SE), Trusted Execution Environment (TEE), or Trusted Platform Module (TPM), which can provide trust utility in Fog computing applications.

Authentication is an important issue for the security of Fog computing since services are offered to massive-scale end users by front Fog nodes/servers. In [33] the authors have considered the main security issue of Fog computing as the authentication at different levels of Fog nodes. *Traditional PKI-based authentication is not efficient and has poor scalability for Fog users at the Edge of the network.*

The work in [2] proposed a cheap, secure and user-friendly solution to the authentication problem in local ad-hoc wireless network, relying on a physical contact for pre-authentication in a location-limited channel. Similarly, NFC can also be used to simplify the authentication procedure in the case of Cloudlet [4]. As the emergence of biometric authentication in mobile computing and Cloud computing, such as fingerprint authentication, face authentication, touch-based or keystroke-based authentication. However, such techniques take relatively long execution time and their security level is always constrained by time complexity, specially when high security level is needed.

Intrusion detection techniques can also be applied in Fog computing [24]. Intrusion in smart grids can be detected using either a signature-based method in which the patterns of behavior are observed and checked against an already existing database of possible misbehaviors. Intrusion can also be captured by using an anomaly-based method in which an observed behavior is compared with expected behavior to check if there is a deviation. The work in [35] develops an algorithm that monitors power flow results and detects anomalies in the input values that could have been modified by attacks. The algorithm detects intrusion by using principal component analysis to separate power flow variability into regular and irregular subspaces.

Password-based authentication techniques [17, 18, 22, 34] have many applications in the Cloud, however, they are not a good idea when it comes to Fog computing due to several reasons: (i) Passwords are characterized by their low entropy, and in order to amplify this entropy to establish session keys, extensive modular arithmetic computations are needed. (ii) Fog users at the Edge of the network communicate with many Fog servers in different Fogs. It is inadequate to keep a password with each server. Moreover, it is not a good idea to keep one common password for all servers. Also the Fog users may communicate in the future with newly joined servers that they never contacted before. (iii) Passwords in general are a weak link in Cloud computing [19] due to its vulnerability to off-line dictionary attacks[1].

Another closely research area is the Secure Wireless Roaming of mobile nodes [9, 36, 37, 41]. However, these protocols are realized by the session keys distributed using public key cryptosystems. The work in [28] presented the lightweight secure structure SPINS and the broadcast authentication protocol μ-TESLA. The μ-TESLA used a reverse hash chain to replace the public-key cryptosystem heavy algorithms. Other protocols are found in [10, 11, 20, 29]. Recently, the work in [40] exploited the advantages of Cloud-assisted BSNs based on MWN model, and designed an efficient, secure and composable protocol for the mobile nodes roaming randomly in the networks, still the protocol requires public key encryption. These protocols are not lightweight enough to be adequate for the massive scalability of Fog computing and the dynamic structure of the network.

Up to our knowledge and until the time this paper was written, the scheme in this paper is the first to directly target mutual authentication in the Edge-Fog layer of the Edge-Fog-Cloud architecture.

3 Motivations and Contributions

In this section, we discuss our motivations and the contribution of our work.

3.1 Motivations

Authentication is an important issue for the security of Fog computing since, services are offered to massive-scale end users by front Fog nodes. Fog users usually have smart devices that are limited in resources and hence, cannot perform extensive traditional digital signatures and public key encryptions. In addition, PKI is impractical to be implemented at the massive scale of the Edge. Password-based authentication strategies are not suitable for communication with large number of servers. Biometric based solutions also have the problem of very long execution time and their security level is always constrained by time complexity, specially when high security level

[1]http://searchcloudsecurity.techtarget.com/tip/password-basedauthentication- a-weak-link-in-cloud-authentication

is needed. Also authentication techniques using Diffie-Hellman key exchange [8], based on the DH problem, use extensive modulo computations which are slow and not suitable for smart devices/cards.

3.2 Our Contribution

We propose an efficient and secure Edge-Fog mutual authentication scheme, to allow any Fog user and any Fog server to mutually authenticate each other, without relying on any PKI. The Fog user is required to store only one long-lived master secret key, and this key allows him to roam randomly in the network and mutually authenticate with any Fog server in any Fog under the authority of a particular CSP. Also, the Fog user must be able to authenticate with new servers joining the Fog without the need to re-register and without any extra overheads. Our scheme uses elementary cryptographic tools (hash functions and symmetric encryption). The computations required by the Fog user are only few hash invocations and symmetric encryptions/decryptions. This makes our scheme very efficient for implementation on smart cards and devices with very limited resources, such as, sensor nodes and smart phones.

4 Model and Assumptions

In this section we describe the network model and assumptions of our scheme.

4.1 Network Model

The Cloud-based Internet is extended by introducing an intermediate layer between Edge users' (Fog users') devices and Cloud, aiming at the smooth, low-latency service delivery from the Cloud to Fog users. This accordingly leads to a three hierarchy Edge-Fog-Cloud architecture. Given a Cloud service provider (CSP), among his Cloud servers in the Cloud layer, there is a special server called the Registration Authority RA of the Cloud, which is responsible for registering Cloud users to the Cloud, as well as Fog users to the Fogs managed by this particular Cloud. Therefore, as shown in Figure 2, under the authority of the RA, there are several locations where for each location (or a Fog F), there is a set of Fog servers/nodes, $\mathcal{FS} = \{FS_1, ..., FS_n\}$. $\mathcal{FS}$ directly communicate with the Fog users, $\mathcal{FU} = \{FU_1, FU_2, ...\}$, in its location through single-hop wireless connections using the off-the-shelf wireless interfaces, such as WiFi, Zigbee or even Bluetooth. With the on-board compute facility and pre-cached contents, they can independently provide pre-defined service applications to mobile users without assistances from Cloud or Internet. On the other hand, the Fog servers, $\mathcal{FS}$, of a Fog F can be connected to the Cloud so as to leverage the rich functions and application tools of the Cloud.

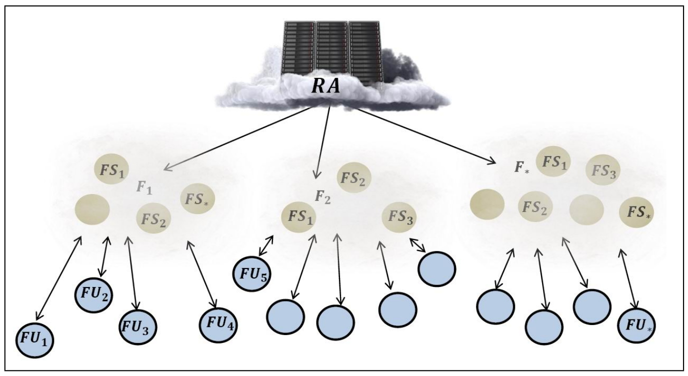

Figure 2: The network model

Dynamic join and leave of Fog Servers. Unlike Cloud servers, the Fog servers/nodes are dynamic in joining and leaving the Fogs. Servers in different places (e.g. Shops, Groceries, Parklands, Bus stops, etc.) may be added to (removed from) a Fog at any time and this may happen frequently. Still services provided by these newly added servers must be available to the registered users. This property of Fog servers must be dealt with in an efficient way from the Fog users' perspective. We argue that this dynamic change of the Fogs must be transparent to the Fog users. I.e. Fog users must still be able to mutually authenticate themselves to the newly joined Fog servers, without the need to re-register any parameters, and without any increase in the complexity at the Fog users' side.

4.2 Assumptions

We assume that the registration authority RA communicates with all managed Fog servers through private and authenticated channels, that could be realized by establishing a public key infrastructure (PKI). In fact, we assume a PKI as a folklore realization of such channels, between the RA and her servers since we focus on the authentication in the Edge-Fog layer. There are many other ways to realize such channels, e.g. if the RA shares a master secret key with each of her servers, the private and authenticated channels are realized without a PKI. Also, PKI may be avoided if the CSP installs her public key pk_{RA} on the servers and the server's private key sk_{FS} on each server FS. The Fog servers in our protocol are not needed to communicate with each other by any means, they communicate only with the RA and with the Edge users when requested. The Fog users at the Edge of the network are completely unrelated to the established PKI. We assume that the RA as a service provider is trusted, however, none of the servers in the Fog are assumed trusted, they are vulnerable to corruption by a corruptive adversary.

5 Our Proposed Scheme

The notations used in our protocol is given in Table 1. Let (E, D) be the encryption/decryption function of a

Table 1: Notations used in our scheme

Notation	**Meaning**
RA	Registration Authority
FU	Fog User
FS	Fog Server/Node
F	Fog/location/zone/area
ID_{FU}	Identity of Fog user
ID_{FS}	Identity of Fog server
ID_F	Fog/zone/area identity
k_{FU}	Fog user master secret key
$k_{FS}^{(FU)}$	Secret key shared between FS and FU
k_s	Session key
$H(x)$	A hash invocation on input x
r_{FU}/r_{FS}	Random nonce picked by FU/FS
(pk_{RA}, sk_{RA})	Public/Private key pair of RA
(pk_{FS}, sk_{FS})	Public/Private key pair of FS
$E(k, x)$	Symmetric key encryption of x using key k
$D(k, x)$	Symmetric key decryption x using key k
$E_{pk}(x)$	public key encryption of x using key pk
$X \to Y$	X computes and sends to Y

strong symmetric encryption scheme (eg. AES), while H is a strong hash function (e.g. SHA-1,SHA-256, etc.). For simplicity we drop the subscript indexes since they are understood. The protocol consists of three phases: (i) Initialization phase, (ii) Registration phase and (iii) Authentication phase.

5.1 Initialization Phase

RA has her own public/private key pair (pk_{RA}, sk_{RA}), where pk_{RA} is known to all servers. Each server has his own public/private key pair (pk_{FS}, sk_{FS}) where RA stores pk_{FS} of each server FS. For each server, FS, in every Fog F, under the authority of RA, RA picks a unique identity ID_{FS} and sends it to FS signed with RA's signature key sk_{RA}. Notice that ID_{FS} is not secret.

5.2 Registration Phase

The registration phase is illustrated in Figure 3. A Fog user FU of identity ID_{FU} approaches the registration authority RA to register. The Fog network F has an identity ID_F and a set of Fog servers $\mathcal{FS}$ with each Fog server FS has an identity ID_{FS}. The registration is as follows:

- FU shows his identity ID_{FU} to the RA.
- RA picks a long-lived random master secret key (with long enough bit-length) k_{FU} for FU.
- FU stores $\langle ID_{FU}, k_{FU}\rangle$ on his smart device/card.
- For each $FS \in \mathcal{FS}$ in each F, RA computes the FS's secret key for FU as $k_{FS}^{(FU)} = H(ID_F, ID_{FS}, k_{FU})$ as shown in Figure 4 [2].
- For each server $FS \in \mathcal{FS}$, RA sends ID_{FU} and $k_{FS}^{(FU)}$ to FS encrypted under FS's public key, pk_{FS}. I.e. ID_{FU} and $E_{pk_{FS}}(k_{FS}^{(FU)})$. All signed with RA's signature key, sk_{RA} for authenticity.
- Finally, each server FS decrypts and stores the tuple $\langle ID_{FU}, k_{FS}^{(FU)}\rangle$ for each FU.

Remark (Joining of a new Fog server). We remark that, whenever a new Fog server (FS) joins a Fog, the RA runs the initialization phase for this server to setup a new identity ID_{FS} for this server and then computes the secret keys $k_{FS}^{(FU_j)} = H(ID_F, ID_{FS}, k_{FU_j})$ for all j of Fog users. We emphasize that this is done without the incorporation of the Fog users FU_j's who are already registered and without any increased overheads on the Fog users' side [3].

5.3 Authentication Phase

The authentication phase is illustrated in Figure 5. When a registered Fog user FU is in the location of the Fog F and needs to authenticate with a server FS, they proceed as follows (notice that, initially, FU does not know the identity ID_{FS} of any server. He just requests a Fog service) :

[2]Figure 4, shows why we named our scheme "Octopus": The master key k_{FU_j} represents the head of an octopus while the generated secret keys $k_{FS_i}^{(FU_j)}$ represent its arms. A user with the same head can later authenticate with any of the arms.

[3]It takes 0.006 ms for one hash invocation on Intel(R) Xeon(R) CPU E5520 @ 2.27G. Therefore, it takes less than a minute to generate FS-FU secret keys for 10 million Fog users.

FU	RA	FS
$\langle ID_{FU} \rangle$	$\langle ID_{FS}, pk_{FS}, sk_{RA} \rangle$	$\langle pk_{RA}, sk_{FS} \rangle$
$\xrightarrow{ID_{FU}}$		
	Picks k_{FU}, Stores $\langle ID_{FU}, k_{FU} \rangle$. $\xleftarrow{k_{FU}}$	
Stores $\langle ID_{FU}, k_{FU} \rangle$		
	Computes $k_{FS}^{(FU)} = H(ID_F, ID_{FS}, k_{FU})$, Computes $E_{pk_{FS}}(k_{FS}^{(FU)})$. $\xrightarrow{\langle ID_{FU}, E_{pk_{FS}}(k_{FS}^{(FU)}) \rangle}$ Signed with sk_{RA}	
		Verifies RA signature using pk_{RA}, Aborts if the verification fails, else, Decrypts using sk_{FS}, Stores $\langle ID_{FU}, k_{FS}^{(FU)} \rangle$ for this FU.

Figure 3: The registration phase of our scheme

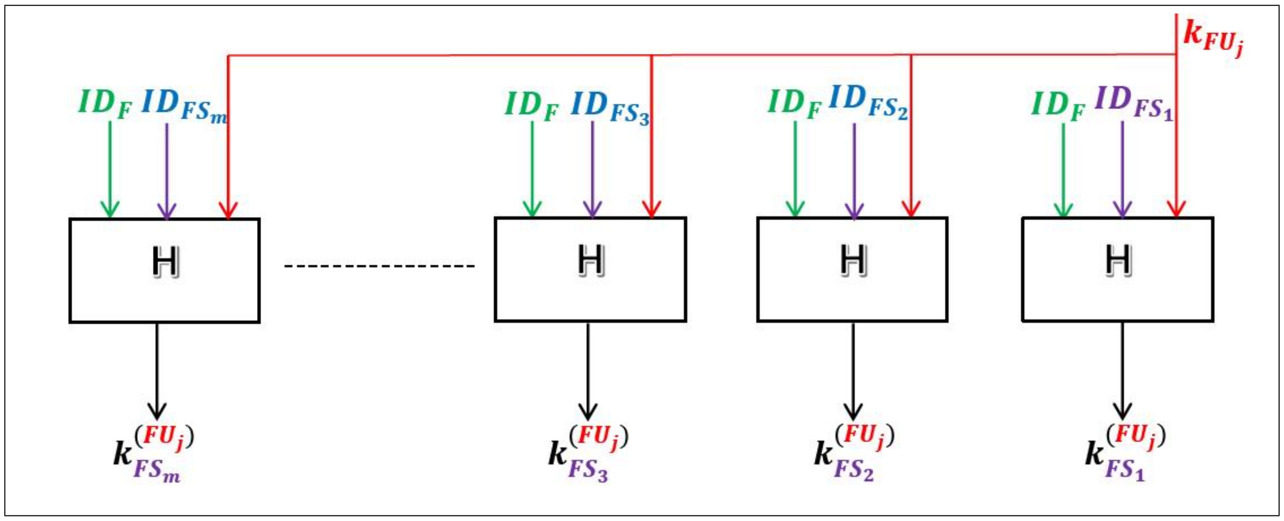

Figure 4: Generation of the FS-FU_j secret keys for user FU_j

$FU \rightarrow F$:

- Picks a random nonce r_{FU}.
- Broadcasts the tuple, $\langle HelloFog, ID_{FU}, r_{FU} \rangle$.

$FS \rightarrow FU$: An in-range server $FS \in \mathcal{FS}$:

- Checks that ID_{FU} is registered, else abort.
- Fetches $k_{FS}^{(FU)}$ for this ID_{FU}.
- Picks a random nonce r_{FS}.
- Prepares the encryption $E(k_{FS}^{(FU)}, (r_{FU}, r_{FS}))$, where $E(K, X)$ is a symmetric encryption of X using secret key K.
- Replies with the tuple, $\langle ID_{FS}, ID_F, ID_{FU}, E(k_{FS}^{(FU)}, (r_{FU}, r_{FS})) \rangle$.

$FU \rightarrow FS$:

- Using the received ID_{FS} and the stored k_{FU}, computes $k_{FS}^{(FU)} = H(ID_F, ID_{FS}, k_{FU})$.
- Decrypts and checks equality of r_{FU} with the received one. If the check fails then abort, otherwise,
- Picks a session key k_s, computes $E(k_{FS}^{(FU)}, (r_{FS}, k_s))$.
- Replies with the tuple, $\langle ID_{FS}, ID_F, ID_{FU}, E(k_{FS}^{(FU)}, (r_{FS}, k_s)) \rangle$.

FS:

- Using $k_{FS}^{(FU)}$, decrypts for (r_{FS}, k_s).

- Checks equality of r_{FS} with the received one, if the check fails then abort, otherwise accepts k_s as a session key.

6 Security Analysis

In this section we analyze the security of our scheme. First we show that the basic security requirements are satisfied, then we proceed to discuss the resistance of our scheme to common adversarial attacks. Finally we provide a formal security proof.

6.1 Basic Security Requirements

Mutual authentication. Mutual authentication between FU and FS is achieved, because both are able to deduce FU-FS secret key $k_{FS}^{(FU)} = H(ID_F, ID_{FS}, k_{FU})$, which is used to encrypt/decrypt for the session key k_s, $E(k_{FS}^{(FU)}, (r_{FS}, k_s))$ by FU and $D(k_{FS}^{(FU)}, (E(k_{FS}^{(FU)}, (r_{FS}, k_s)))$ by FS. The session key k_s will not be common to FU and FS unless the encryption and decryption are performed using the same secret key $k_{FS}^{(FU)}$. The Fog user FU generates the secret key $k_{FS}^{(FU)}$ locally, using his master secret key k_{FU} and the claimed server identity ID_{FS}. On the other hand, the RA has generated $k_{FS}^{(FU)}$ in the same way and delivered it secretly to the server. Hence, if a server identity ID_{FS} is claimed without knowing $k_{FS}^{(FU)}$, the server will not be authenticated by a legitimate user. On the other hand, a Fog user that does not hold the correct k_{FU} matching his identity ID_{FU} stored on the server, will not be authenticated by the server.

Protection of $k_{FS}^{(FU)}$. By inspecting our authentication protocol, the shared key $k_{FS}^{(FU)}$ is never used to encrypt a plaintext known to an eavesdropper, it is used to encrypt messages with a fresh random nonce r_{FS} as part of the plaintext. This random nonce is long, temporary, unknown to an eavesdropper and never placed on the channel in the clear.

Confidential communication session. The session key k_s is shared by both participants before performing their subsequent communication. The FU-FS secret key $k_{FS}^{(FU)}$ is known only to FU and FS, and is used to encrypt/decrypt for k_s. Therefore, the proposed scheme provides confidential communication.

Low computation and storage costs. There is no exponential computation or public key computation required on both sides during the authentication phase in the proposed scheme. Only a few hashes and symmetric encryptions/decryptions. Also, the scheme requires the user to store one master secret key k_{FU}, beside few short identities. Hence, the proposed scheme is efficient and easy to implement on smart cards. Therefore, the proposed scheme provides low computation complexity and storage complexity.

Simple key management. In the proposed scheme, the key management is very simple since, only the long-term secret key k_{FU} is maintained at the RA and on the FU's smart device. There is no PKI required at the FU. Furthermore, FS stores only one secret key for each registered user beside his short identity.

Session independence. The fresh session key k_s is not deduced from previous session keys, and there is no relationship among the session keys. Each session key is chosen as a fresh random string. Hence, a compromise of one session key does not affect other past/future sessions.

6.2 Adversarial Attacks

Fog server compromise (Rogue node). When an adversary corrupts/compromises a Fog server FS_i, then she knows all the FU-FS_i secret keys $\{k_{FS_i}^{(FU_1)}, ..., k_{FS_i}^{(FU_n)}\}$ of the users on this server. We emphasize the following:

- *This compromise does not threaten the security of the master secret key k_{FU} of any user FU given that the used hash function is a strong one-way function and the master secret key k_{FU} as an input to the hash function is long enough to withstand brute force given any compromised $k_{FS}^{(FU)}$.*

- *Compromising the server FS_i does not allow the adversary to deduce any other FU-FS secret keys on any other server on this Fog or any other server on any other Fog. This follows from the fact that, the FU-FS_i are generated independently by applying a one-way hash function on the master secret key k_{FU} of FU and the server's identity ID_{FS_i} since k_{FU} is not known to the adversary.*

The countermeasure for a server compromise is simple. Simply after the corrupted server is cleaned (rebooted, scanned, etc.) the registration authority RA chooses a new identity ID'_{FS_i} for this particular server, regenerates new set of FU-FS_i secret keys, $\{k'^{(FU_1)}_{FS_i}, ..., k'^{(FU_n)}_{FS_i}\}$ where $k'^{(FU_j)}_{FS_i} = H(ID_F, ID'_{FS_i}, k_{FU_j})$ and sends them to FS_i as in the registration phase. The Fog users are informed with the rogue identity ID_{FS_i} in public. *We emphasize that FU master secret key k_{FU} is safe and that FU is not required to incorporate in any new registration processes.*

Secret key guessing attacks. The only secret on the user's side is the user's master key k_{FU}. The key is a strong secret key with long enough bits (to protect against brute force attacks in case a server is compromised) and protected in a tamper-resistant mechanism, such as a smart card. There is no efficient way to obtain it, but brute-force guessing. Therefore,

FU $\langle ID_{FU}, k_{FU}\rangle$		FS $\langle ID_{FU}, ID_{FS}, ID_F, k_{FS}^{(FU)}\rangle$
Picks r_{FU}		
	$\xrightarrow{\langle HelloFog, ID_{FU}, r_{FU}\rangle}$	
		Checks that ID_{FU} exists Fetches $k_{FS}^{(FU)}$ Picks r_{FS} Computes the encryption, $E(k_{FS}^{(FU)}, (r_{FU}, r_{FS}))$
	$\xleftarrow{\langle ID_{FS}, ID_F, ID_{FU}, E(k_{FS}^{(FU)}, (r_{FU}, r_{FS}))\rangle}$	
Computes, $k_{FS}^{(FU)} = H(ID_F, ID_{FS}, k_{FU})$ Decrypts & checks validity of r_{FU} Aborts if the check fails, otherwise, Picks k_s Computes $E(k_{FS}^{(FU)}, (r_{FS}, k_s))$		
	$\xrightarrow{\langle ID_{FS}, ID_F, ID_{FU}, E(k_{FS}^{(FU)}, (r_{FS}, k_s))\rangle}$	
		Decrypts for r_{FS}, k_s Checks validity of r_{FS} Aborts if the check fails, else, Accepts k_s as a session key

Figure 5: Edge-Fog mutual authentication phase

the proposed scheme is secure against secret key guessing attacks.

Replay/impersonation attacks. Consider the case where an adversary records all data transfered between FU and FS, during the authentication phase and the whole session. Now an adversary may try to replay any message at any round, wishing to succeed in the replay and impersonate either FU or FS. Of course trying to replay a data session, encrypted under an old session key k_s will not succeed, unless the adversary succeeds in the authentication phase. Now, lets see what happens if the adversary replays each round in the authentication phase. Assuming that the adversary is a Fog user FU', that replays $\langle HelloFog, ID_F, ID_{FU}, r_{FU}\rangle$ in Figure 5. The server FS replies with the tuple, $\langle ID_{FS}, ID_F, ID_{FU}, E(k_{FS}^{(FU)}, (r_{FU}, r_{FS}))\rangle$ challenging FU' with the random nonce r_{FS}. FU' does not know $k_{FS}^{(FU)}$ and hence, he performs the encryption using some random k^* as $E(k^*, R)$ for some R. Now, the server FS decrypts using the correct $k_{FS}^{(FU)} \neq k^*$, resulting in some $r_{FS}^* \neq r_{FS}$ and hence, the r_{FS} produced by FS equals the received one only with negligible probability. Hence, FU' will not succeed in the third round. In the same way, replaying the second round by an adversary FS', impersonating FS, will not succeed due to the random fresh challenge r_{FU}. Finally, trying to impersonate FU and replay a previously recorded third round, will not succeed since the included random nonce does not match that of round two by FS.

Man-in-the-middle attack. Consider an adversary that puts herself as an intermediate node between FU and FS. This adversary does not know k_{FU}. Now, this adversary tries to masquerade each party to the other. Since, this adversary does not know k_{FU}, she cannot deduce the secret key $k_{FS}^{(FU)}$, generated locally by FU and stored in FS. The adversary cannot generate any correct encryptions of the random nonce r_{FS} or r_{FU}. Hence, the man-in-the-middle attack fails.

Fog user compromise. If FU device is compromised, then the master secret key k_{FU} falls in the hands of the adversary. *This compromise does not affect any other Fog user.* However, for this compromised user FU, he must report to the RA for revoking his compromised master key and register for a new one.

6.3 Formal Security Proof

In this subsection, we give a formal proof of the security of our scheme.

Theorem 1. *Assuming E and H used in our authentication scheme are secure pseudo-random function families, then our scheme based on E and H is a secure mutual entity authentication and key exchange protocol.*

Proof. Assuming H is a strong hash function and that k_{FU} is a long enough secret key, it is infeasible for an attacker knowing any $k_{FS}^{(FU)} = H(ID_F, ID_{FS}, k_{FU})$ to reach k_{FU}. Now we continue to prove our theorem using BAN logic [6] given that both FU and FS believe in $k_{FS}^{(FU)}$ as follows:

Idealization. By removing all plaintext messages, the idealized messages between FU and FS in our scheme are as follows:

- M1: $FU \to FS$: $-$
- M2: $FS \to FU$: $\{(r_{FU}, r_{FS})\}_{k_{FS}^{(FU)}}$
- M3: $FU \to FS$: $\{r_{FS}, FU \xleftrightarrow{k_s} FS, \#(FU \xleftrightarrow{k_s} FS)\}_{k_{FS}^{(FU)}}$

Assumptions. The assumptions of the protocol are as follows:

- A1: $FU \mid\equiv \#(r_{FU})$
- A2: $FS \mid\equiv \#(r_{FS})$
- A3: $FU \mid\equiv (FU \xleftrightarrow{k_{FS}^{(FU)}} FS, \#(FU \xleftrightarrow{k_s} FS))$
- A4: $FS \mid\equiv FS \xleftrightarrow{k_{FS}^{(FU)}} FU$
- A5: $FS \mid\equiv (FU \Rightarrow FU \xleftrightarrow{k_s} FS, \#(FU \xleftrightarrow{k_s} FS))$
- A6: $FU \mid\equiv (FU \xleftrightarrow{k_s} FS, \#(FU \xleftrightarrow{k_s} FS))$

Main goals.

- G1: $FS \mid\equiv (FU \xleftrightarrow{k_s} FS, \#(FU \xleftrightarrow{k_s} FS))$
- G2: $FS \mid\equiv FU \mid\equiv (FU \xleftrightarrow{k_s} FS, \#(FU \xleftrightarrow{k_s} FS))$.
- G3: $FU \mid\equiv FS \mid\equiv r_{FU}$
- G4: $FS \mid\equiv FU \mid\equiv r_{FS}$

Analysis.

From assumptions A1 and A3 and message M2,

$$\frac{FU \mid\equiv \#(r_{FU})}{FU \mid\equiv \#(r_{FU}, r_{FS})} \text{ (Freshness rule)}$$

$$\frac{FU \mid\equiv FU \xleftrightarrow{k_{FS}^{(FU)}} FS, FU \lhd \{(r_{FU}, r_{FS})\}_{k_{FS}^{(FU)}}}{FU \mid\equiv FS \mid\sim (r_{FU}, r_{FS})}$$

(Message meaning rule)

$$\frac{FU \mid\equiv \#(r_{FU}, r_{FS}), FU \mid\equiv FS \mid\sim (r_{FU}, r_{FS})}{FU \mid\equiv FS \mid\equiv (r_{FU}, r_{FS})}$$

(Nonce verification rule)

$$\frac{FU \mid\equiv FS \mid\equiv (r_{FU}, r_{FS})}{FU \mid\equiv FS \mid\equiv r_{FU}} \text{ (Belief rule)}$$

This satisfies goal G3.

Let $X = (FU \xleftrightarrow{k_s} FS, \#(FU \xleftrightarrow{k_s} FS))$. From assumptions A2 and A4 and message M3, we have,

$$\frac{FS \mid\equiv \#(r_{FS})}{FS \mid\equiv \#(r_{FS}, X)} \text{ (Freshness rule)}$$

$$\frac{FS \mid\equiv FU \xleftrightarrow{k_{FS}^{(FU)}} FS, FS \lhd \{(r_{FS}, X)\}_{k_{FS}^{(FU)}}}{FS \mid\equiv FU \mid\sim (r_{FS}, X)}$$

(Message meaning rule)

$$\frac{FS \mid\equiv \#(r_{FS}, X), FS \mid\equiv FU \mid\sim (r_{FS}, X)}{FS \mid\equiv FU \mid\equiv (r_{FS}, X)}$$

(Nonce verification rule)

$$\frac{FS \mid\equiv FU \mid\equiv (r_{FS}, X)}{FS \mid\equiv FU \mid\equiv X} \text{ (Belief rule)}$$

Thus, goal G2 is reached.

$$\frac{FS \mid\equiv FU \mid\equiv (r_{FS}, X)}{FS \mid\equiv FU \mid\equiv r_{FS}} \text{ (Belief rule)}$$

This satisfies goal G4.

From assumption A5 we have,

$$\frac{FS \mid\equiv FU \Rightarrow X, FS \mid\equiv FU \mid\equiv X}{FS \mid\equiv X} \text{ (jursdiction rule)}$$

Thus, goal G1 is reached and so the proof of the theorem. □

7 Complexity Evaluation

Our protocol uses only two simple cryptographic primitives; several invocations of a strong hash function H and symmetric encryption/decryption E/D (e.g. AES), making the protocol very efficient for smart card implementation. There are many hash functions out there for cryptographic applications such as SHA-1, SHA-2, SHA-224, SHA-256, etc. [27]. For the concrete evaluation of the complexity of our protocol, we assume SHA-1 as the hash function in place. SHA-1 takes an input as an arbitrary length message partitioned in blocks of 512 bits where the last block is padded with zeros to complete the block size. Each 512-bit block produces a SHA-1 output of 160 bits where these 160 bits are re-invoked as input with the next 512-bit message block. The final output of SHA-1 is 160 bits as the hash of the arbitrary length message [27].

We assume the identity ID_{FU} is of size 3 bytes (enough for a huge population, however the size maybe a little longer since the user's identity contains printable characters). The identities ID_{FS} and ID_F are assumed one byte each. These choices may differ according to the population.

We also assume that k_{FU} is of 160 bits just for the purpose of evaluation. The length of k_{FU} may be chosen freely by RA, since it is not incorporated in any symmetric key encryptions. k_{FU} is used only as an input to the hash function to generate the keys $k_{FS}^{(FU)}$, therefore, its effective bit-length is exactly its actual bit-length chosen freely be RA.

7.1 Storage Requirements

The storage requirements of our scheme are given in Table 2 and detailed next.

Fog user FU. The Fog user FU is required to store the tuple $\langle ID_{FU}, k_{FU}\rangle$ which is a random long enough string as his master secret key k_{FU} in addition to

Table 2: Storage and computations requirements of our scheme

	Storage	**Computations**		
		Initialization phase	**Registration phase**	**Authentication phase**
FU	One short ID. One secret key.	–	–	One hash invocation. One symmetric encryption. One symmetric decryption.
FS	Two public keys. One private key. One secret key/FU. One short ID/FU. One short ID_F. One short ID_{FS}.	One signature verification.	One signature verification. One private key decryption.	One symmetric encryption. One symmetric decryption.
RA	One public key. One private key. One public key/FS. One short ID/FS. One short ID/F. One secret key/FU. One short ID/FU.	One signature generation. One hash invocation/FS.	One private key encryption. One hash invocation. One signature generation.	–

a one short string as his identity ID_{FU}.

Fog server FS**.** FS is required to store a short string as his identity ID_{FS} and the tuple $\langle ID_{FU}, k_{FS}^{(FU)}\rangle$ for each FU which consists of a short string as ID_{FU} and the FS-FU secret key $k_{FS}^{(FU)}$. This is in addition to RA's public verification key pk_{RA} and his own public/private key pair (pk_{FS}, sk_{FS}) of the public key cryptosystem in place.

Registration authority RA**.** RA stores the master secret keys of all registered Fog users in addition to the public keys of all Fog servers and her own public/private key pair (pk_{RA}, sk_{RA}) of the public key cryptosystem in place. These are in addition to the users and servers short identities.

7.2 Computation Complexity

The computation complexity of our scheme is given in Table 2.

Fog user FU**.** By inspecting our protocol, FU does not perform any computations in the registration phase, he just receives k_{FU}. In the authentication phase, FU performs only one hash invocation, one symmetric encryption and one symmetric decryption.

Fog server FS**.** In the registration phase, FS performs one signature verification for his identity ID_{FS} and one signature verification for each registered user. In the authentication phase, performs one symmetric encryption and one symmetric decryption.

Registration authority RA**.** In the Initialization phase, performs one hash invocation for the generation of ID_{FS} and one digital signature on ID_{FS} for each FS. In the registration phase, for each registered user FU and each Fog server FS, performs one hash invocation, one digital signature and one public key encryption.

7.3 Computation Time

A simulation hardware environment is setup to measure the computation time of the cryptographic primitives. The simulation environment is a 32-bit Cortex-M3 microcontroller with 72 MHz ARM MCU and 512 KB memory [21]. A secret key encryption of an AES-128 block cipher takes 0.919 ms, while the decryption takes 1.074 ms. A one invocation of hash function SHA-1 takes 0.06 ms .

It takes FU about 0.06 ms (one SHA-1 invocations) plus 0.919 ms (one AES encryption) plus 1.074 ms (One AES decryption) totaling 2.053 ms on FU side.

On the FS side, it takes about 0.919 ms (One AES encryption) plus 1.074 ms (One AES decryption) totaling 1.993 ms. Computation time is summarized in Table 3.

7.4 Energy Consumption

In this part, the energy consumption consumed by cryptographic operations is used to evaluate the schemes. This time, we use a low-processor and 64 MB memory running Windows Mobile 5.0 for pocket pc. According to PXA270, the typical power consumption of PXA270 in active is 570 mW [4]. Therefore, using the computation time in the previous subsection, we can calculate the corresponding energy consumption. For example, if it takes 0.919 ms to complete a AES-128, the energy consumption is approximately 0.919∗(570/1000) = 0.523 mJ. So the energy consumed by FU is 1.17 mJ while the energy

[4]http://pdf.dzsc.com/CXX/NHPXA270Cxxx.pdf

consumed by FS is 1.14 mJ. The energy consumed by our scheme is summarized in Table 3.

7.5 Comparison with Closely Related Work

Although, up to the time this paper was written, and up to our knowledge, there is no contribution that directly targets the mutual authentication in the Edge-Fog-Cloud architecture, we discuss other close contributions targeting wireless sensor networks. The roaming protocols of [9, 29] used the identity-based cryptography and group signature to realize the local authentication of the roaming protocol. The communication times of the mobile node in their protocols do not contain the transmission of the authentication materials. The communication times of [10] are equal or greater than by four times, because of the re-authentication process after every moving. The protocol stores all the authentication materials into the neighboring nodes through broadcast, and the broadcast communication computes at least once communication.

These protocols and the recent in [40] employ public-key cryptosystems and bilinear pairings as essential requirements which dramatically increase the computations complexities specially for smart cards. Our protocol does not require the engagement of public key cryptosystems.

Table 3: Computation time and energy consumption

	Computation time		**Energy consumed**
	Round 1	**Round 2**	
FU	–	2.053 ms	1.17 mJ
FS	0.919 ms	1.074 ms	1.14 mJ

8 Discussions

Some applications, such as vehicle-to-vehicle communications in VANETS [32, 30, 31], requires that the Fog users (vehicles) interact with each other within a certain Fog. Given that, each Fog user FU_j shares a secret key with a Fog server FS as $k_{FS}^{(FU_j)}$, there are many protocols that allow this server to establish a common session key for these users allowing them to communicate in a private way. For example, one may consider the Wide-Mouth-Frog protocol [6]. Another protocol is the Needham-Schroeder Symmetric Key Protocol based on a symmetric encryption algorithm, which forms the basis for the Kerberos protocol [25]. Many other server-based key distribution exist [5].

Anonymity is one of the important services that must be available to users in the digital world as long as they behave honestly. Users' communication must be kept authenticated and anonymous unless malicious behaviors are detected. In this case the accused user's clear identity must be traced and revealed by the system to solve accusations [14, 13, 15, 12]. In the Edge-Fog-Cloud model, Edge users have the right to keep their identities anonymous as long as they are honest, while on the other hand, the CSP has the right to be able to trace any user to his clear identity once he/she misbehaves. So, it would be a nice open problem to find a way to add this service to our scheme, or to devise a new authentication scheme that provides this service to the Edge users.

An important requirement of anonymity by many applications is unlinkability of virtual identities, i.e. an adversary $\mathcal{A}$ must not be able to link activities (e.g. transactions) to the same person/entity although his clear identity is blinded from $\mathcal{A}$. There exist schemes based on what is known as "temporary identities" (e.g. [26, 42]). In such schemes, the user shares his identity with the server and this identity is updated to a new fresh string after each session in a way unpredictable to the attacker. Such schemes are computationally efficient and secure. However, the problem with such schemes is that, both the user and the server must be in synchronism with the current temporary identity. At a certain round of the protocol, the adversary may disrupt the communication (e.g. through jamming) resulting in a loss of synchronism between the user and the server. The consequences of such attack is the DoS of the current session and all future sessions. These schemes may be suitable for small area networks where it is easy to reset and reinitialize the system when such attack is detected. However, for large scale networks such as the Edge-Fog-Cloud architecture, such schemes are impractical.

9 Conclusions

Services of Fog computing are offered to massive-scale end users where it is hard to realize PKI on such a large scale at the Edge of the network. We proposed a secure and efficient scheme to allow any Fog user to mutually authenticate with any Fog server in any Fog under the authority of a Cloud service provider. Our Scheme does not require a Fog user to be incorporated in any PKI. The Fog user is required to store one master secret key in the registration phase only once. Using this master key the Fog user is able to mutually authenticate with any Fog server managed by the Cloud service provider. On the other hand, Our scheme provides a simple countermeasures if one or more Fog servers are compromised and fully corrupted by an adversary. Even if all the Fog servers are corrupted, the master secret key of the user with long enough bit-length remains secure against brute force and hence, the Fog user does not need to be incorporated in any re-initialization or re-registration of a new master key. Also, the Fog user is able to mutually authenticate with any Fog server that joins a Fog after Fog user registration without the need for the user to re-register and without any extra overheads on the user's side.

Our Scheme is computationally efficient, even in the existence of huge population. It requires the Fog users and the Fog servers to perform very few hash invocations and symmetric key encryptions/decryptions and hence, it is suitable for implementation on smart cards and devices with limited resources.

References

[1] M. Armbrust, A. Fox, R. Griffith, et al., "A view of cloud computing," *Communications of the ACM*, vol. 53, pp. 50–58, Apr. 2010.

[2] D. Balfanz, D. K Smetters, P. Stewart and H. C. Wong, "Talking to strangers: Authentication in adhoc wireless networks," in *Symposium on Network and Distributed Systems Security*, 2002.

[3] F. Bonomi, R. Milito, J. Zhu and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the ACM Workshop on Mobile Cloud Computing*, pp. 13–16, New York, USA, 2012.

[4] S. Bouzefrane, B. Mostefa, F. Amira, F. Houacine and H. Cagnon, "Cloudlets authentication in NFC-based mobile computing," in *2nd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering*, pp. 267–272, 2014.

[5] C. Boyd and A. Mathuria, *Protocols for Authentication and Key Establishment*, Springer Science & Business Media, 2013.

[6] M. Burrows, M. Abadi and R. M. Needham, "A logic of authentication," in *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 426, pp. 233–271, The Royal Society, 1989.

[7] E. Damiani, D. Capitani D. Vimercati, S. Paraboschi, P. Samarati and F. Violante, "A reputation-based approach for choosing reliable resources in peer-to-peer networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pp. 207–216, 2002.

[8] Z. M. Fadlullah, M. M. Fouda, N. Kato, A. Takeuchi, N. Iwasaki and Y. Nozaki, "Toward intelligent machine-to-machine communications in smart grid," *IEEE Communications Magazine*, vol. 49, no. 4, pp. 60–65, 2011.

[9] D. He, J. Bu, S. Chan, C. Chen and M. Yin, "Privacy-preserving universal authentication protocol for wireless communications," *IEEE Transactions on Wireless Communications*, vol. 10, no. 2, pp. 431–436, 2011.

[10] D. He, C. Chen, S. Chan and J. Bu, "Strong roaming authentication technique for wireless and mobile networks," *International Journal of Communication Systems*, vol. 26, no. 8, pp. 1028–1037, 2013.

[11] Y. M. Huang, M. Y. Hsieh, H. C. Chao, S. H. Hung and J. H. Park, "Pervasive, secure access to a hierarchical sensor-based healthcare monitoring architecture in wireless heterogeneous networks," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 4, pp. 400–411, 2009.

[12] M. H. Ibrahim, "AATCT: Anonymously authenticated transmission on the cloud with traceability," *International Journal of Advanced Computer Science & Applications*, vol. 6, no. 9, pp. 251–259, 2015

[13] M. H. Ibrahim, "Resisting traitors in linkable democratic group signatures," *International Journal of Network Security*, vol. 9, no. 1, pp. 51–60, 2009.

[14] M. H. Ibrahim, "Noninteractive, anonymously authenticated, and traceable message transmission for VANETs," *International Journal of Vehicular Technology*, 2009.

[15] M. H. Ibrahim, "Secure anonymously authenticated and traceable enterprise DRM system," *International Journal of Computer Applications*, vol. 126, no. 3, September 2015.

[16] A. Jøsang, R. Ismail and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618–644, 2007.

[17] M. Kumar, "An enhanced remote user authentication scheme with smart card," *International Journal of Network Security*, vol. 10, no. 3, pp. 175–184, 2010.

[18] M. Kumar, "A new secure remote user authentication scheme with smart cards," *International Journal of Network Security*, vol. 11, no. 2, pp. 88–93, 2010.

[19] C. C. Lee, C. H. Liu and M. S. Hwang, "Guessing attacks on strong-password authentication protocol," *International Journal of Network Security*, vol. 15, no. 1, pp. 64–67, 2013.

[20] M. Li, W. Lou and K. Ren, "Data security and privacy in wireless body area networks," *IEEE Wireless Communications*, vol. 17, no. 1, pp. 51–58, 2010.

[21] J. Liu, Q. Li, R. Yan and R. Sun, "Efficient authenticated key exchange protocols for wireless body area networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2015, no. 1, pp. 1–11, 2015.

[22] R. Lu, Z. Cao, Z. Chai and X. Liang, "A simple user authentication scheme for grid computing.," *International Journal of Network Security*, vol. 7, no. 2, pp. 202–206, 2008.

[23] R. Lu, X. Li, X. Liang, X. S. Shen and X. Lin, "Grs: The green, reliability, and security of emerging machine to machine communications," *IEEE Communications Magazine*, vol. 49, no. 4, pp. 28–35, 2011.

[24] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel and M. Rajarajan, "A survey of intrusion detection techniques in cloud," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 42–57, 2013.

[25] R. M. Needham and M. D. Schroeder, "Using encryption for authentication in large networks of computers," *Communications of the ACM*, vol. 21, no. 12, pp. 993–999, 1978.

[26] C. Noureddine, F. Cherif, P. L. Cayrel and B. Mohamed, "Improved rfid authentication protocol based on randomized mceliece cryptosystem," *International Journal of Network Security*, vol. 17, no. 4, pp. 413–422, 2015.

[27] National Institute of Standards and Technology (NIST), "Fips 180-4, secure hash standard," *Federal Information Processing Standards Publication.*

[28] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen and D. E. Culler, "SPINS: Security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521–534, 2002.

[29] Y. Qiu, J. Zhou, J. Baek and J. Lopez, "Authentication and key establishment in dynamic wireless sensor networks," *Sensors*, vol. 10, no. 4, pp. 3718–3731, 2010.

[30] F. M. Salem, M. H. Ibrahim and I. I. Ibrahim, "Non-interactive authentication scheme providing privacy among drivers in vehicle-to-vehicle networks," in *IEEE Sixth International Conference on Networking and Services*, pp. 156–161, 2010.

[31] F. M. Salem, M. H. Ibrahim and I. I. Ibrahim, "Non-interactive secure and privacy preserving protocol for inter-vehicle communication networks," in *IEEE Seventh International Conference on Information Technology: New Generations*, pp. 108–113, 2010.

[32] F. M. Salem, M. H. Ibrahim and I. I. Ibrahim, "Efficient noninteractive secure protocol enforcing privacy in vehicle-to-roadside communication networks," *International Journal of Vehicular Technology*, vol. 2012, 2012.

[33] I. Stojmenovic and S. Wen, "The fog computing paradigm: Scenarios and security issues," in *IEEE Federated Conference on Computer Science and Information Systems*, pp. 1–8, 2014.

[34] J. L. Tsai, "Efficient nonce-based authentication scheme for session initiation protocol," *International Journal of Network Security*, vol. 9, no. 1, pp. 12–16, 2009.

[35] J. Valenzuela, J. Wang and N. Bissinger, "Real-time intrusion detection in power system operations," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1052–1062, 2013.

[36] Z. Wan, K. Ren and B. Preneel, "A secure privacy-preserving roaming protocol based on hierarchical identity-based encryption for mobile networks," in *Proceedings of the First ACM conference on Wireless Network Security*, pp. 62–67, 2008.

[37] J. Wang, Y. Yanshuo and K. Zhou, "A regular expression matching approach to distributed wireless network security system," *International Journal of Network Security*, vol. 16, no. 5, pp. 382–388, 2014.

[38] W. Wang and Z. Lu, "Survey cyber security in the smart grid: Survey and challenges," *Computer Networks*, vol. 57, pp. 1344–1371, April 2013.

[39] C. Wei, Z. M. Fadlullah, N. Kato and I. Stojmenovic, "On optimally reducing power loss in micro-grids with power storage devices," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 7, pp. 1361–1370, 2014.

[40] Q. Q. Xie, S. R. Jiang, L. M. Wang and C. C. Chang, "Composable secure roaming authentication protocol for cloud-assisted body sensor networks," *International Journal of Network Security*, vol. 18, no. 5, pp. 816–831, 2016.

[41] G. Yang, Q. Huang, D. S. Wong and X. Deng, "Universal authentication protocols for anonymous wireless communications," *IEEE Transactions on Wireless Communications*, vol. 9, no. 1, pp. 168–174, 2010.

[42] E. J. Yoon, K. Y. Yoo, J. W. Hong, S. Y. Yoon, D. I. Park and M. J. Choi, "An efficient and secure anonymous authentication scheme for mobile satellite communication systems," *EURASIP Journal on Wireless Communications and Networking*, vol. 2011, no. 1, pp. 1–10, 2011.

Maged Hamada Ibrahim. B.Sc. in Communications and Computers Engineering from Helwan University with Distinction and Honor's Degree in 1995. He also obtained his M.Sc. from the same University in 2001. Then his Ph.D. from Helwan University in 2005. He is now an Associate Professor at Helwan University. He is joining several network security projects in Egypt. His main interest is engineering cryptography and communications security. More specifically, working on the design of efficient and secure cryptographic algorithms and protocols, in particular, secure distributed multiparty computations, public key infrastructures, digital signatures, digital rights management protocols and non-cryptographic solutions to telecommunication security problems. Other things that interest him are number theory and the inspection of mathematics for designing secure and efficient cryptographic schemes.