LAAP: Lightweight Anonymous Authentication Protocol for D2D-Aided Fog Computing Paradigm

Prosanta Gope

Department of Computer Science

University of Sheffield, Regent Court, 211 Portobello Sheffield, S1 4DP, UK
Email: prosanta.nitdgp@gmail.com/p.gope@sheffield.ac.uk

Abstract

Fog computing is a new paradigm that extends cloud computing and services to the edge of the network. Although it has several distinct characteristics, however, the conventional fog computing model does not support some of the imperative features such as D2D communications, which can be useful for several critical IoT applications and services. Besides, fog computing faces numerous new security and privacy challenges apart from those inherited from cloud computing, however, security issues in fog computing have not been addressed properly. In this article, first we introduce a new privacy-preserving security architecture for fog computing model with the cooperative D2D communication support, which can be useful for various IoT applications. Subsequently, based on the underlying foundation of our proposed security architecture we design three lightweight anonymous authentication protocols (LAAPs) to support three distinct circumstances in D2D-Aided fog computing. In this regard, we utilize the lightweight cryptographic primitives like one-way function and EXCLUSIVE-OR operations, which will cause limited computational overhead for the resource limited edge devices.

Keywords: Fog computing, D2D communication, D2D-Aided fog computing, Anonymity, Lightweight authentication.

1 Introduction

Emergence of fifth generation wireless communication (5G) will bring about revolution in the communications technology. It is envisioned to have a huge numbers of wireless devices (e.g. smart meters, cars, sensors, etc.) in 5G, where it will provide more than 10Gb/s speed, with 1000 times higher wireless capacity and save up to 90 percent of energy consumption compared with the current 3GPP LTE-A system [1]. 5G can also support ultra-high definition visual communications, multimedia interactive, mobile industry automation, vehicle connectivity and other applications and thus achieve a real "*Internet of Everything*". To support better connectivity and communication link quality in 5G, small cell concept has been introduced by deploying more small cell base stations (BSs). Furthermore, device-to-device (D2D) communication has also been considered as a way to enhance the network performance by allowing UEs to communicate directly with their corresponding destinations instead of using a BS or access point (AP) [2]. The D2D communication also make devices in the same proximity help each other for the better performance of services.

Now, IoT devices, especially tiny sensors, usually face challenges rooted from limited resources (e.g., computation power, storage, etc.), which may often impair quality of services (QoS) and user experience in IoT. To resolve the resource limitation issue at IoT devices, the concept of *cloud computing* was introduced as a promising computing paradigm, which can offer services to the end users with elastic resources at low cost [3]. However, the cloud computing has drawbacks as well. A primary limitation is delay-the lag between client request and server response, which is regarded as a problem for latency-sensitive applications that require nodes in the vicinity to meet their delay requirements [4]. Besides, while techniques and devices of IoT are getting more and more involved in people's life, current cloud computing paradigm can barely support their needs such as low latency, mobility support, etc

To fulfill the above requirements (such as massive connectivity and low latency) of modern applications, the concept of a new paradigm was introduced, which is known as the *fog computing*, [5] (shown in Fig 1). The fog computing model is designed primarily to minimize delay by exploiting the fog layer, which is the middle layer between the core network and edge devices. Fog computing has its advantages due to its edge location, and therefore is able to support applications (e.g. gaming, augmented reality, real time video stream processing) with low latency requirements. This edge location can also provide rich network context information, such as local network condition, traffic statistics and client status information, which can be used by fog applications to offer context-aware optimization. Another interesting characteristic is the location-awareness; not only can the geo-distributed fog node infer its own location but also the fog node can track end user devices to support mobility, which may be a game changing factor for location-based services and applications. Furthermore, the interplays between fog and cloud become important since fog can easily gets local overview while the global coverage can only be achieved at a higher layer. In general, a fog computing model consists of three major components:(i) edge device (ED), (ii) network access device (NAD), i.e., fog node, in the proximity of an NAD, and (iii) centralized cloud server (CCS) [3]. EDs can be various sensors or tiny-powered smart devices collecting information from the certain area, where they are deployed. NAD, which equips with more powerful computation ability and has sustainable energy supply, can be considered as access point (AP) in WLAN, base station (BS) in mobile networks, and road side unit (RSU) in vehicular ad-hoc networks. CCS can be considered as a server of any specific purpose, e.g., authentication server for security, database server for data storage, application server for membership and service management, etc. In different application scenario, the characteristics of fog components will be different due to the application requirements. For example, depending on the nature of the IoT applications, EDs can be treated as either the movable or fixed objects. In an organization that offers Internet access through Wi-Fi, one may roam around the whole campus with his/her gadgets (mobile device, tablet, laptop, etc) and get support from several NADs. In that case, EDs can be considered as movable objects. On the other hand, in smart-grid, smart meters (EDs) are implanted to collect usage information and send those information to a nearest NAD/substation [20]. Therefore, in that case, EDs are the fixed objects. Generally, if EDs always needs to interact with CCS then that will make the CCS busy. In fact, that may incurs the performance of the whole system. The concept of fog computing can easily resolve this issue. In that case, EDs are allowed to interact with a nearby NAD, which has more resources such as memory, storage, and computing power. The concept of fog computing can be useful for several IoT applications, such as IoT-based health-care system [23]. In that case, a group of sensors (EDs in fog computing) can send their reading to the CCS through an NAD support, where EDs are connected to the NAD through a short-range communications such as Wi-Fi, Zig-bee, and Blue-tooth. The concept of fog computing can be useful for various mission critical applications that require real-time data processing. For example, in a cloud robotic system, the motion control of a robot depends on the data collected by the sensors and feedback of the control system. The control system running on the centralized cloud may cause the sense process-actuate loop slow or even unavailable as a results of communication failure. This is where the concept of fog computing can be used by performing the required processing for the control system near to the robot. The key difference between fog and cloud computing is the fog's proximity to the underlying accessing nodes. The fog computing is localized, while the cloud computing is centralized.

1.1 Problem Statement and Motivation

As discussed, the concept of fog computing offers several notable features. However, we should say that it is still in its infant stage, and there are many issues which have not been considered yet including security concerns. First of all, the number of authentication requests will increase since the handover chance between the fog nodes will increase at EDs. However, the computational requirements

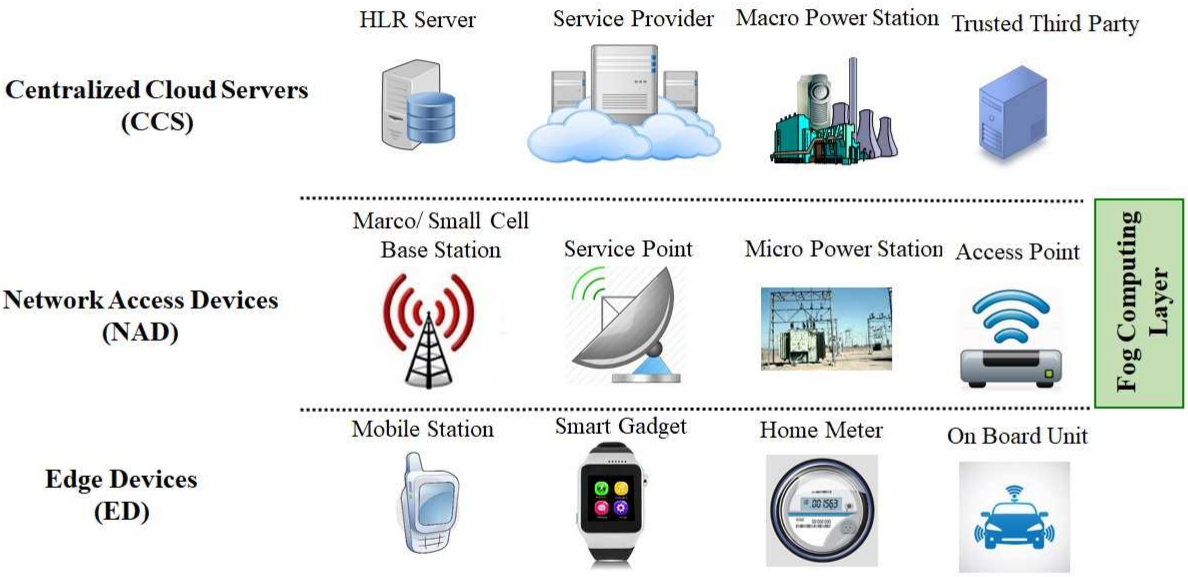


Figure 1: Fog Computing Paradigm

of authentication requests in conventional fog modeling has not been well addressed, even the fog devices, i.e., macro or small cell base stations, may need to more helps, compared to the conventional security protection in 3/4G, where, each authentication need to involve HSS/AuC located in the core network and the authentication information, including session keys, is computed by it. Although the authentication works have been offloaded to fog nodes, the performance bottleneck on authentication may not be alleviated due to the increasing number of requests. Hence, for the aforesaid reasons, the new type of security architecture is required to satisfy. Secondly, in fog computing, there are many privacy issues as fog nodes will collect and process various sensitive information, therefore security and privacy of these information are highly desirable. Besides, the mobility of EDs will require interaction with several fog nodes, and this may cause privacy issues as well. Therefore, although inclusion of fog node layer will reduce the load of the centralized cloud server, keeping the footprint of identity information secretly to the fog nodes is essential for privacy protection.

Our Contributions. The contributions of this article are threefold. First, we introduce a novel privacy-preserving security architecture for D2D-Aided fog computing model, which can provide verification of the end-user devices without involving centralized server. Second, based on the underlying foundation of our proposed security architecture we design three lightweight anonymous authentication protocols (LAAPs) to support three different circumstances in D2D-Aided fog computing. In order to do that, we utilize the lightweight cryptographic primitives [6], such as one-way hash function and exclusive-or operations to support the security even for resource-limited IoT devices [21]. Third, we provide the comprehensive performance evaluation of the proposed authentication protocols for fog computing in terms of privacy, computation cost, and communication cost, and verify the feasibility of the protocols for ensuring the key security features such as anonymity and mutual authentication, secure key exchange, etc. It can be argued that, our proposed scheme can be useful for various critical real-time application scenarios. For instance, in order to access Internet through Wi-Fi Protected Access (WPA), end user devices need to be authenticated through a server. Now, in case if the server is broken or under maintenance then that may cause interruption in Internet access services. Conversely, in our proposed security architecture for fog computing if the centralized server is broken then also we can ensure services without any interruption.

Paper organization. The remainder of this article is organized as follows. In Section 2, we first present a new security architecture for D2D-Aided fog computing paradigm (shown in Fig. 2), subsequently we define the security requirements in designing authentication protocol for D2D-Aided fog computing paradigm. Subsequently, in Section 3, we introduce three authentication protocols which support three different scenarios in the proposed security architecture of fog computing. In Section 4, we informally analyze the security of the proposed protocols and then in Section 5 we formally analyze the security of the LAAP protocols. In Section 6, we provide the performance evaluation of the authentication protocols in terms of privacy, computation cost, and communication cost, and verify the feasibility of the protocols for ensuring the key security features for privacy preserving IoT such as anonymity and identity verifications. Finally, a concluding remark is given in Section 7

2 System and Attacker Models

In this section, we first illustrate a new security architecture for D2D-Aided fog computing paradigm (shown in Fig. 2), subsequently we define the security requirements in designing authentication protocol for D2D-Aided fog computing paradigm.

2.1 System Model

In conventional fog computing model, when an ED and NAD need to interact securely, then for authenticating each others, they may need the support of CCS, and that impairs the performance of the system in terms of higher latency. Besides, in many application scenarios, such as secure handover in 5G, we need the support of low latency, and for that it's imperative to have D2D communication technology. Furthermore, due to the dramatic growth of the number of devices, some applications, an NAD can be fully overloaded. In order to resolve this issue, NADs may need to interact each other to offload some of computation overhead. On the other hand, for roaming services in mobile communication an NAD can cooperate another NAD to verify the legitimacy of a mobile station (ED) through the expeditious authentication process. To support all these imperative features in fog computing, we need fog computing model with cooperative D2D communications, that permits EDs to interact with each other and even help each other to validate without involving CCS. In this way, the fog computing layer is expanded from NAD layer only to NAD and ED layers in the generic model.

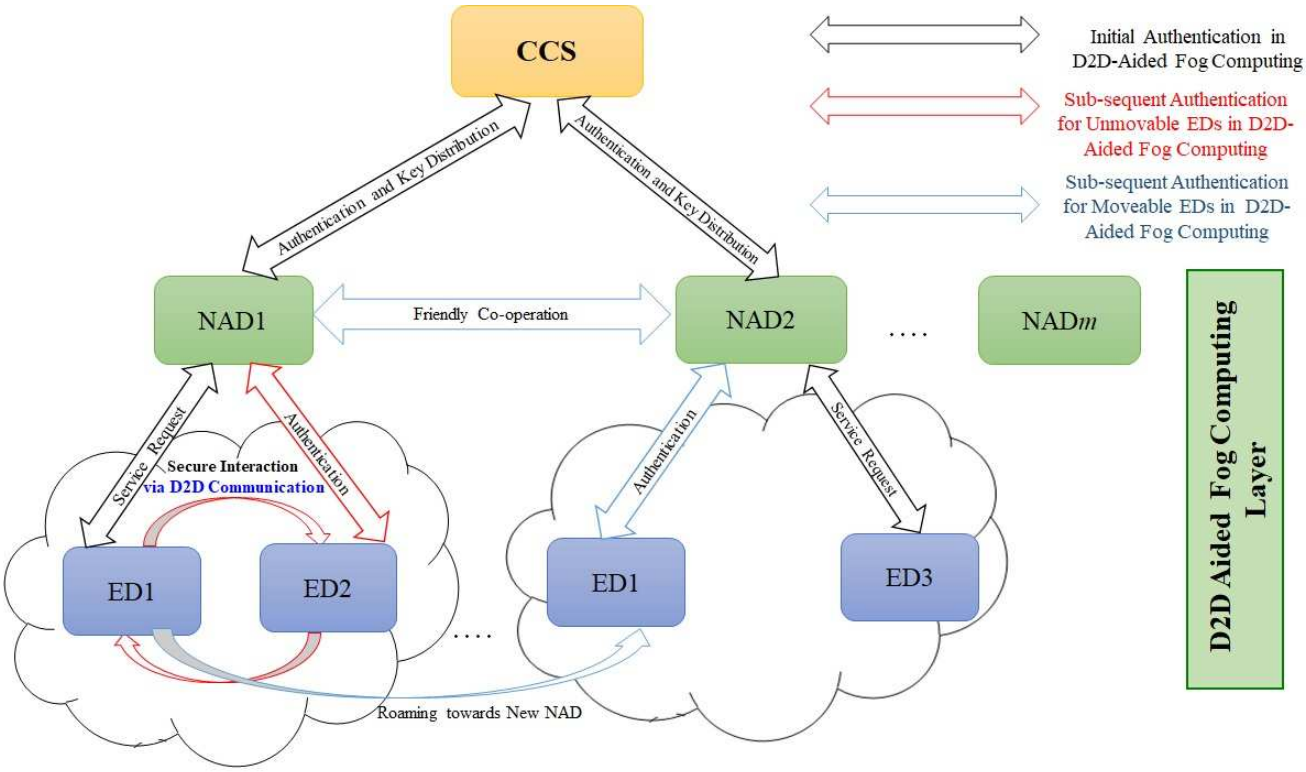

CCS
Initial Authentication in D2D-Aided Fog Computing
Sub-sequent Authentication for Unmovable EDs in D2D-Aided Fog Computing
Sub-sequent Authentication for Moveable EDs in D2D-Aided Fog Computing
Authentication and Key Distribution
Authentication and Key Distribution
NAD1
Friendly Co-operation
NAD2
....
NAD*m*
D2D Aided Fog Computing Layer
Service Request
Authentication
Secure Interaction via D2D Communication
Authentication
Service Request
ED1
ED2
....
ED1
ED3
Roaming towards New NAD

Figure 2: Security Architecture for the D2D-Aided Fog Computing Model

The extended fog computing layers will be able to significantly reduce the load at CCS.

Fig. 2 shows a new security architecture for D2D-Aided fog computing paradigm. In this architecture, we support the following three authentication scenarios by considering both the fixed and movable EDs: (i) Initially, for secure interaction, a legitimate ED in a group under the proximity of an NAD needs to be authenticated with help of CCS. (ii)When the next ED of the same group wants to interact with NAD, then the most recent ED, who already authenticated by NAD, will help the next one to be authenticated. (iii) In case of movable ED, when an ED moves to the proximity of a new NAD then for authenticating ED, the most recently visited NAD will help the new one. Now, as discussed, our proposed security architecture can even work if the CCS is broken or under maintenance. In that case, through the subsequent authentication process of (ii) and (iii), the legitimacy an ED can be verified by NAD. This feature of the proposed scheme, can be useful for various critical applications such as Internet access through WPA.

2.2 Attacker Model and Security Requirements

An emerging wave of Internet deployments, most notably the Internet of Things (IoTs), requires the support of mobility, geographical distribution, in addition to location awareness and low latency. The concept fog computing is proposed in the context of IoT which can be regarded as a useful paradigm to meet these requirements. However, in order to ensure security and privacy in several critical IoT services [26] and applications (such as VANETs, Smart Grid, M2M, etc.), there is a requirement of privacy preserving secure fog computing model. In this regard, the following security issues should be addressed properly.

Mutual Authentication: It is an important issue for the security of fog computing since services are offered to massive-scale end users by front fog nodes. Traditional PKI-based authentication protocol is not efficient for the resourced-limited EDs and has poor scalability.

Secure Key Exchange: For ensuring communication security, ED needs to share the same secret communication key with NAD for a particular session. After establishing secret communication key through secure key-exchange mechanism, both ED and NAD can use this key for ensuring communication security.

Anonymity: In general, anonymity can be divided into two types: weak anonymity and strong anonymity [22-23]. The first one can be achieved by hiding the real-identity of the entity using any encryption or encoding method, but in that case, because of the same cipher or encoded output, an entity can easily be traced. In case of strong anonymity, the adversary (e.g eavesdropper) cannot trace the entity (e.g. EDs) by using interactions with them. If the transmitted information cannot satisfy that property, then an attacker can continuously trace the activity of the entity. This feature can also help to support location privacy, where EDs offload their tasks to the nearest fog nodes, to whom the tasks are offloaded, can infer that fog node is nearby and farther from other nodes. Furthermore, if a fog edge device utilizes multiple fog services at multiple locations, it may disclose its path trajectory to other fog nodes, assuming the fog nodes collude. As long as such an ED is attached on a person or an object, the location privacy of the person or the object will be at risk. If an ED always strictly selects its nearest fog node for services, the fog node can surely identify the edge device utilizing its computing resources is nearby.

3 Lightweight Anonymous Authentication Protocol (LAAP) for D2D-Aided Fog Computing Paradigm

When an ED wants to send it's reading data then ED needs to interact with the nearest NAD. To ensure privacy preserving issues, it's essential that the interaction should be anonymous. Moreover, for the the stronger sense of security, it is also important that during anonymous interaction, the entities should be able to authenticate each other and also can check the freshness of the interacted messages. Besides, in many IoT applications, EDs are tiny powered, therefore, the security solution for offering secure anonymous interaction should be lightweight. In this section, we present three anonymous authentication protocols for three different situations in D2D-aided fog computing. In this regard, the first authentication protocol is designed for initial authentication in presence of CCS. Other two protocols are designed for the subsequent authentications in D2D-aided fog computing by considering the fixed and movable EDs, respectively. The proposed protocols are designed based on the lightweight cryptographic primitives such as hash function, EXCLUSIVE-OR, where cryptographic hash function causes less computational overhead as compared to any symmetric/asymmetric encryption schemes [16]. The notations used in the proposed scheme are summarized below in Table 1.

Table 1: Notions and Cryptographic Functions

Symbol	**Definition**
AID	One-time Alias Identity
$\{pid_1,pid_2,...,pid_n\}$	Set of Unlink-able Pseudo IDs
$Tseq$	Transaction Sequence Number
K_{ec}	Shared Secret Key between the ED and CCS
GK	Shared Group Key
CK	Communication Key
tk	Temporary Key
$h(.)$	Secure One-way Hash Function
$\oplus$	EXCLUSIVE-OR
$\vert\vert$	Concatenation Operation

3.1 Registration Phase

Conceive, there is a group of edge devices have been deployed in a particular area, where they need to register into the CCS. Our registration process consists of the following steps:

(1) The i-th ED ED_i requests to be registered into CCS through a secure channel.

(2) CCS maintains a global counter ($gcount$) of 64-bit. For any request this counter is incremented by one. After receiving request from ED_i, the CCS increments the value of $gcount$ by one and subsequently, generates a transaction sequence number $Tseq = gcount$, along with a secret shared key K_{ec} and a set of un-linkable pseudo IDs PID =$\{pid_1$, pid_2,,...,pid_n,$\}$ for ED_i, and then provides these parameters along with the group key GK to the ED_i through the secure channel, by maintaining a copy of these parameters in its database.

3.2 Initial Authentication Protocol for D2D-Aided Fog Computing (LAAP1)

This Phase of the proposed scheme will be executed when the first ED ED_i, in a group wants to send its data reading or the field information to the CCS with the help of a nearest NAD. Besides, we recommend this Phase to be executed when any uncanny situation arises. For example, NAD cannot validate the legitimacy of an ED with the help of other ED, then NAD needs to interact with CCS through this Phase. This Phase consists of the following steps:

Step 1: ED_i→NAD: $M_{A_1}: \{AID,\ N_x,\ Tseq\}$.

ED_i generates a random number N_e and computes $N_x = N_e \oplus\ K_{ec}$, a one-time alias identity$AID = h(ID_{\mathrm{ED}_i}||K_{ec}||Tseq)$, and then sends AID along the transaction sequence number $Tseq$ to a nearest NAD, where ID_{ED_i}, K_{ec}, $h(.)$ represent the identity of the edge device, shared secret key, the random number generated by the edge device, and one-way non-collision hash unction, respectively. Note that, in case of loss of synchronization ED_i needs to select one of the unused pair of $(pid_j,\ k_{em_j})$ and assign $AID = pid_j$, $k_{em_j} = K_{ec}$. In that case, ED_i need not to send any transaction sequence number $Tseq$ in M_{A_1}.

Step 2: NAD→ CCS : M_{A_2}:{Fwd. M_{A_1}}.

Since, NAD has no information about the ED_i, hence it forwards the request message $M_{A_1}$to the CCS.

Step 3: CCS→ NAD : M_{A_3}:{$e1$, $e2$, $\mathrm{Res}_{\mathrm{CCS}}$, CK}.

After receiving the message M_{A_2}, CCS at first finds the transaction sequence number $Tseq$ in its database, and then retrieves ID_{ED_i}, K_{ec} from that particular row of the database. Hereafter, CCS computes and validates the request parameters like AID_{ED}. . If the verification is successful then the CCS generates a communication key CK and a new transaction sequence number $Tseq_{new}$. Subsequently, ED_i computes $e1 = h(K_{ec} \mid\mid Tseq) \oplus Tseq_{new}$, $e2 = h(K_{ec} \mid\mid ID_{\mathrm{ED}_i}) \oplus CK$, $\mathrm{Res}_{\mathrm{CCS}} = h(e1 \mid\mid e2 \mid\mid K_{ec})$, and updates $Tseq = Tseq_{new}$. Finally, CCS forms a response message M_{A_3} and sends to the NAD.

Note that, if CCS cannot find the $Tseq$, provided by ED_i in its database, then CCS will try to recognize pid_j in AID. In that case, the system (CCS) can comprehend that there is a loss of synchronization with ED_i. Now, if the system can recognize pid_j then it will proceed for any futher computation and at the end it sends a response message M_{A_3} to NAD.

Step 4: NAD→ED_i:M_{A_4}:{$e1$, $e2$, Res_{CCS}, Res_{NAD}, TN, R_n}

Upon receiving the response message M_{A_3} with the communication key CK , NAD generates a *Track No.* and a random number R_n, and computes $TN = h(CK \ \mid\mid R_n) \oplus$*Track No.*, $Res_{NAD} = h(Track\ No. \mid\mid\ CK \mid\mid R_n)$. Subsequently NAD forms a response message M_{A_4}, then sends M_{A_4} to the edge device ED_i.

Step 5: Verification at ED_i

After receiving M_{A_4}, ED_i first validates the response parameters ($\mathrm{Res}_{\mathrm{CCS}}$, $\mathrm{Res}_{\mathrm{NAD}}$) received from both the NAD and CCS, respectively. If the verification is successful then ED_i first decodes $Tseq_{new}$, CK and updates $Tseq = Tseq_{new}$. Hereafter, ED_i broadcasts the random number R_n to other group members. Finally, ED_i needs to encode it's identity by $\mathrm{ED}_i^* = \mathrm{ED}_i \bigoplus h(R_n \mid\mid GK)$ and then stores it into a common group database (Table II), which consists of two fields i.e. the last edge device communicated with NAD and it's status, which can be either "*Idle*" or "*Busy*" . Note that, when an ED interacts with a NAD while NAD is busy to handle other requests, the NAD will forward the request of the later one to another NAD. In this way, loads at NADs can be balanced, which is one of the important feature of the fog computing model. It should be noted that, our LAAP1 authentication scenario can be observed in the applications, where three different types of entities are involved during authentication process. The detail procedure of this Phase is depicted in Fig. 3.

Run-Out Situation:

Now, in case when all the pseudo IDs are used up then ED_i needs to send a "Run-Out" request i.e. $\mathrm{RO}_1 : \{[\mathrm{ED}_i \mid\mid T_{ED} \mid\mid GK]E_{GK}$, GID, $T_{ED}\}$, where T_{ED}, and GID denote the timestamp and group identity of ED_i, respectively. Upon receiving RO_1, CCS first checks the timestamp T_{ED}. If it is valid then CCS finds the GK based on GID. Subsequently, CCS generates a set of new pseudo IDs PID_{new} =(pid_1, pid_2, ...,pid_n) and forms a response message RO_2:$\{PID_{new}$, $\mathrm{Res}_{\mathrm{CCS}}$, $T_{CSS}\}$and sends RO_2 to ED_i. The detail procedure of this Phase is depicted in Fig. 3.

Table 2: Common Group Database

Last Communicated Edge Device	**Status**
ED_i^*	*Idle/Busy*

3.3 Subsequent Authentication Protocol with the Co-operation of EDs in D2D-Aided Fog Computing (LAAP2)

It should be noted that, in the above fog computing model, CCS needs to involve for every device request, which is a conventional approach. Since the number of IoT devices are increasing everyday. Therefore, dealing with a huge number of requests will increase the load of CCS and that may also impair the performance of the entire system. On the other hand, since the number of IoT devices are increasing everyday, dealing with a huge number of requests will increase the load at CCS, and this may impair the performance of the entire system. On the other hand, in D2D-fog computing model, devices in the same proximity can help each other for the better performance of security services, e.g., authentication. Therefore, we introduce the cooperative D2D communication support, where EDs can

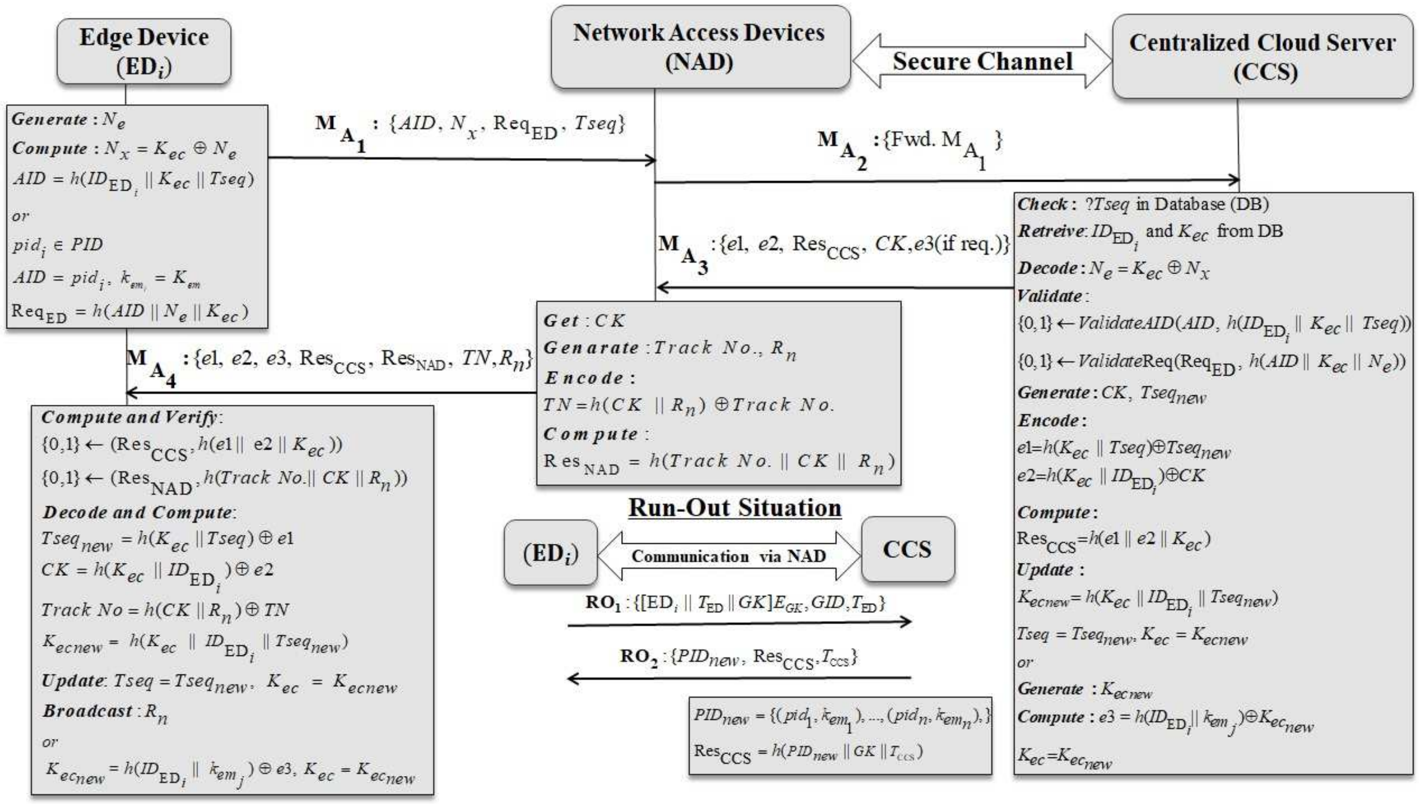

Edge Device (ED_i)
Network Access Devices (NAD)
Secure Channel
Centralized Cloud Server (CCS)
Generate : N_e
Compute : $N_x = K_{ec} \oplus N_e$
$AID = h(ID_{\mathrm{ED}_i} \| K_{ec} \| Tseq)$
or
$pid_i \in PID$
$AID = pid_i,\ k_{em_i} = K_{em}$
$\mathrm{Req}_{\mathrm{ED}} = h(AID \| N_e \| K_{ec})$
M_{A_1} : $\{AID, N_x, \mathrm{Req}_{\mathrm{ED}}, Tseq\}$
M_{A_2} :{Fwd. M_{A_1} }
Check : ?$Tseq$ in Database (DB)
Retreive: ID_{ED_i} and K_{ec} from DB
Decode : $N_e = K_{ec} \oplus N_x$
Validate :
$\{0,1\} \leftarrow ValidateAID(AID, h(ID_{\mathrm{ED}_i} \| K_{ec} \| Tseq))$
$\{0,1\} \leftarrow Validate\mathrm{Req}(\mathrm{Req}_{\mathrm{ED}}, h(AID \| K_{ec} \| N_e))$
Generate : $CK, Tseq_{new}$
Encode :
$e1 = h(K_{ec} \| Tseq) \oplus Tseq_{new}$
$e2 = h(K_{ec} \| ID_{\mathrm{ED}_i}) \oplus CK$
Compute :
$\mathrm{Res}_{\mathrm{CCS}} = h(e1 \| e2 \| K_{ec})$
Update :
$K_{ecnew} = h(K_{ec} \| ID_{\mathrm{ED}_i} \| Tseq_{new})$
$Tseq = Tseq_{new}, K_{ec} = K_{ecnew}$
or
Generate : K_{ecnew}
Compute : $e3 = h(ID_{\mathrm{ED}_i} \| k_{em_j}) \oplus K_{ec_{new}}$
$K_{ec} = K_{ec_{new}}$
M_{A_3} :{$e1$, $e2$, $\mathrm{Res}_{\mathrm{CCS}}$, CK,$e3$(if req.)}
Get : CK
Genarate : $Track\ No., R_n$
Encode :
$TN = h(CK \| R_n) \oplus Track\ No.$
Compute :
$\mathrm{Res}_{\mathrm{NAD}} = h(Track\ No. \| CK \| R_n)$
M_{A_4} :{$e1$, $e2$, $e3$, $\mathrm{Res}_{\mathrm{CCS}}$, $\mathrm{Res}_{\mathrm{NAD}}$, TN,R_n}
Compute and Verify:
$\{0,1\} \leftarrow (\mathrm{Res}_{\mathrm{CCS}}, h(e1 \| e2 \| K_{ec}))$
$\{0,1\} \leftarrow (\mathrm{Res}_{\mathrm{NAD}}, h(Track\ No. \| CK \| R_n))$
Decode and Compute:
$Tseq_{new} = h(K_{ec} \| Tseq) \oplus e1$
$CK = h(K_{ec} \| ID_{\mathrm{ED}_i}) \oplus e2$
$Track\ No = h(CK \| R_n) \oplus TN$
$K_{ecnew} = h(K_{ec} \| ID_{\mathrm{ED}_i} \| Tseq_{new})$
Update: $Tseq = Tseq_{new},\ K_{ec} = K_{ecnew}$
Broadcast : R_n
or
$K_{ec_{new}} = h(ID_{\mathrm{ED}_i} \| k_{em_j}) \oplus e3,\ K_{ec} = K_{ec_{new}}$
Run-Out Situation
(ED_i)
Communication via NAD
CCS
RO_1 :{[$\mathrm{ED}_i \| T_{\mathrm{ED}} \| GK]E_{GK}, GID, T_{\mathrm{ED}}$}
RO_2 :{PID_{new}, $\mathrm{Res}_{\mathrm{CCS}}$,$T_{\mathrm{CCS}}$}
$PID_{new} = \{(pid_1, k_{em_1}), ..., (pid_n, k_{em_n}),\}$
$\mathrm{Res}_{\mathrm{CCS}} = h(PID_{new} \| GK \| T_{\mathrm{CCS}})$

Figure 3: Initial Authentication Protocol in Presence of CCS for D2D-Aided Fog Computing

verify each other by preserving the privacy from any eavesdropper. In this case, we assume that both ED_i and ED_j share a link key K_{ij}, through Blom's scheme [24], which is based on MDS (Maximum Distance Separable) code. For a general introduction to MDS codes see [25]. Blom's scheme uses a public $(\lambda + 1) \times N$ matrix G and a private matrix $N{\times}(\lambda + 1)$ matrix D which is generated over GF(q) and N is the size of the group, where q is a prime power. This solution is λ- secure, meaning that keys are secure if no more than λnodes are compromised. In order to do that, G must have $(\lambda+1)$ linearly independent columns. Then the Key matrix can be defined as a symmetric matrix K$=$ $(D.G)^T.G$. After deployment, all EDs broadcast their column instances of G and row instances of matrix $(D.G)^T$, that allows any pair of EDs i and j to compute the link-key $K_{ij} = row_i{\times}column_j$ and $K_{ji} = row_j{\times}column_i$, respectively. For detailed analysis and proofs the interested reader is referred to [24]. This phase of authentication process consists of the following steps:

Step1: $\mathrm{ED}_j \rightarrow \mathrm{ED}_i$: M_{B_1}: $\{AID, GAuth, Tseq\}$.

When an edge device ED_j wants to interact with it's nearest NAD, then NAD needs to verify the legitimacy of $\mathrm{ED}_j$with the help of the most recently interacted EDs, ED_i. Therefore, it's assumed that the last interaction between the ED_i and the NAD was successful. Now, in order to interact with NAD, ED_j at first encodes it's identity into a one-time alias identity $AID = h(ID_{\mathrm{ED}_j}||GK \ || \ Tseq)$ and generates a group authentication request $GAuth = h(ID_{\mathrm{ED}_j}|| \ R_n|| \ GK \ || \ K_{ij})$, and subsequently it checks the "*Status*" field of the common group database. If the status field is "*Idle*" then ED_j needs to decode ED$_i^*$ by using the group key GK and the random number R_n and then immediately sends a request message $M_{B_1}$to the ED$_i$. Otherwise the edge device ED$_j$ needs to wait for the new entry of "Last Communicated Edge Device" (as shown in Table 2).

Step 2: ED$_i$→NAD: M_{B_2}:{$TrackNo.$ Req$_{\mathrm{ED}_i}$,$Tseq$, $tk*$}.

After receiving the request message M_{B_1}, ED$_i$ decodes ID_{ED_j} from the alias identityAID and subsequently checks the parameter $GAuth$, to verify the legitimacy of ED$_j$and also to find that whether it belongs to the same group or not. If not, then ED$_i$ terminates the interaction. Otherwise, ED$_i$ generates a temporary key tk and encodes tk by using CK. Finally, ED$_i$ forms a request message M_{B_2} based on the Track No. (received from NAD) and communication key CK and then sends $M_{B_2}$to NAD.

Step 3: NAD→ED$_i$:M_{B_3}:{$e1$, $e2$, Res$_{\mathrm{NAD}}$, R_n}.

Upon receiving the response message M_{B_2}, NAD at first validates the Track No. and decodes tk from $tk*$. Hereafter, NAD generates a new communication key and track number (CK_{new},$Track_{new}$) and a random number R_n and then encodes the CK_{new} and $Track_{new}$ with the temporary key tk(received from ED$_i$) and finally sends them along the response parameter Res$_{\mathrm{NAD}}$ and R_n to ED$_i$.

Step 4: ED$_i$→ ED$_j$:M_{B_4}:{$e1$, $e2$, Res$_{\mathrm{ED}_i}$, R_n , $tk^{\#}$}.

After receiving M_{B_3}, ED$_i$ at first validates Res$_{\mathrm{NAD}}$and then encodes the temporary key tk by using the group key GK and link key K_{ij}, i.e.$tk^{\#}$=$h(GK \mid\mid ID_{ED_i}\mid\mid K_{ij}) \oplus tk$. Hereafter, ED$_i$ forms a response message M_{B_4} and then sends it to ED$_j$.

Step 5: Verification at ED$_j$

Upon receiving M_{B_4},ED$_j$ first checks the response parameter Res$_{\mathrm{ED}_i}$ and subsequently decodes tk from $tk^{\#}$, $Track_{new}$ from $e1$, and CK_{new} from $e2$ and then broadcast the random number R_n to other group members. It should be noted that here the random number R_n is used for replay attack protection. The detail procedure of this Phase is depicted in Fig. 4.

It should be noted that, even though our LAAP2 has been designed for the fixed edge devices. However, the protocol can also provide authentication support for the limited range of mobility applications. For example, Internet access through Wi-Fi in a campus. In that case, a set of EDs can form a group and then during roaming in the campus, one ED who is already authenticated can help another one to be authenticated and get Internet services.

Now, to enhance the security level of the proposed scheme it is imperative to update the group key in regular interval and also when one of the following events occurs: (i) a new ED joins the group; (ii) a joined ED leaves the group. In that case, we directly adopt the secure hash based key management protocol of [9].

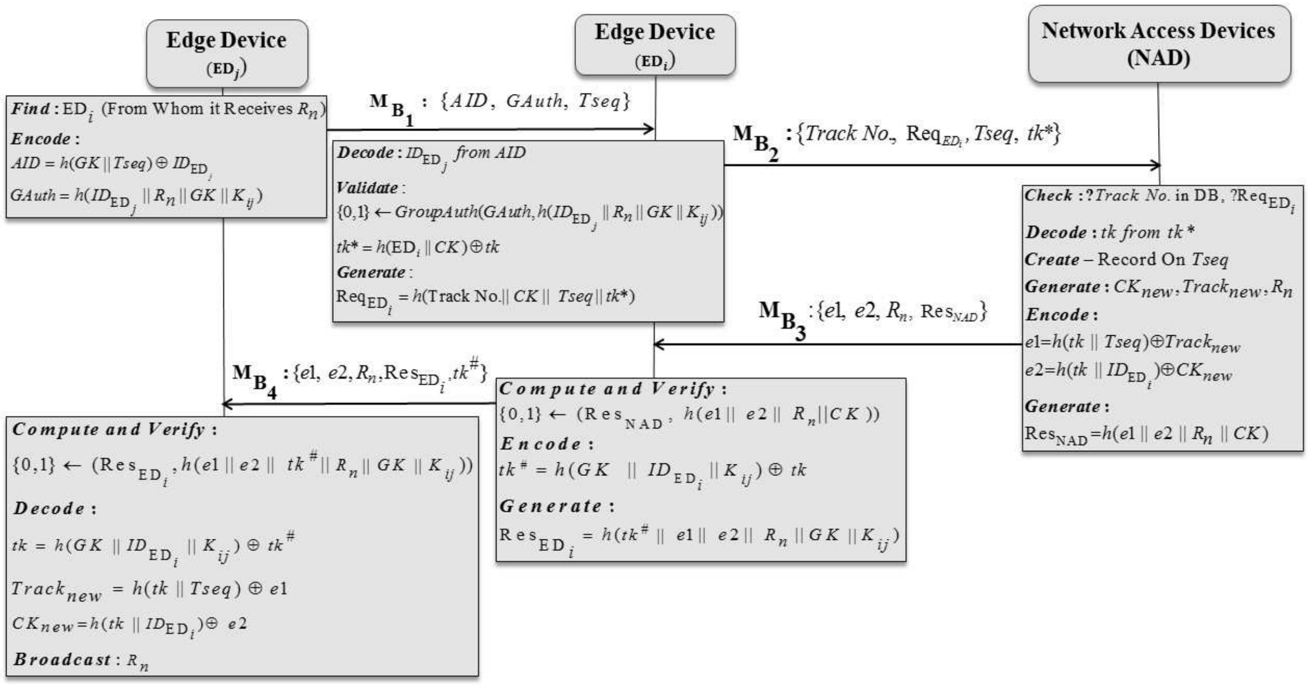

Edge Device (ED$_j$)
Edge Device (ED$_i$)
Network Access Devices (NAD)
Find : ED$_i$ (From Whom it Receives R_n)
Encode :
$AID = h(GK \| Tseq) \oplus ID_{ED_j}$
$GAuth = h(ID_{ED_j} \| R_n \| GK \| K_{ij})$
M_{B_1} : {AID, $GAuth$, $Tseq$}
Decode : ID_{ED_j} from AID
Validate :
$\{0,1\} \leftarrow GroupAuth(GAuth, h(ID_{ED_j} \| R_n \| GK \| K_{ij}))$
$tk^* = h(ED_i \| CK) \oplus tk$
Generate :
$Req_{ED_i} = h(\text{Track No.} \| CK \| Tseq \| tk^*)$
M_{B_2} : {$Track\ No.$, Req_{ED_i}, $Tseq$, tk^*}
Check : ?$Track\ No.$ in DB, ?Req_{ED_i}
Decode : tk from tk^*
Create – Record On $Tseq$
Generate : CK_{new}, $Track_{new}$, R_n
Encode :
$e1 = h(tk \| Tseq) \oplus Track_{new}$
$e2 = h(tk \| ID_{ED_i}) \oplus CK_{new}$
Generate :
$Res_{NAD} = h(e1 \| e2 \| R_n \| CK)$
M_{B_3} : {$e1$, $e2$, R_n, Res_{NAD}}
Compute and Verify :
$\{0,1\} \leftarrow (Res_{NAD}, h(e1 \| e2 \| R_n \| CK))$
Encode :
$tk^{\#} = h(GK \| ID_{ED_i} \| K_{ij}) \oplus tk$
Generate :
$Res_{ED_i} = h(tk^{\#} \| e1 \| e2 \| R_n \| GK \| K_{ij})$
M_{B_4} : {$e1$, $e2$, R_n, Res_{ED_i}, $tk^{\#}$}
Compute and Verify :
$\{0,1\} \leftarrow (Res_{ED_i}, h(e1 \| e2 \| tk^{\#} \| R_n \| GK \| K_{ij}))$
Decode :
$tk = h(GK \| ID_{ED_i} \| K_{ij}) \oplus tk^{\#}$
$Track_{new} = h(tk \| Tseq) \oplus e1$
$CK_{new} = h(tk \| ID_{ED_i}) \oplus e2$
Broadcast : R_n

Figure 4: Subsequent Authentication Protocol with the Co-operation of EDs in D2D-Aided Fog Computing

3.4 Subsequent Authentication Protocol with the Co-operation of NADs in D2D-Aided Fog Computing (LAAP3)

When an ED moves to the proximity of a new NAD (say NAD$_i$) then to authenticate ED, the most recently visited NAD (say NAD$_j$)will help the new one. In that case, we assume that the initial authentication process between ED and NAD$_i$ through LAAP1 was successful, and ED received valid authentication token fromNAD$_j$ i .e. $Token = \{(NAD_j \| T_{nad} \| Track\ No. \| Sign)E_{GK_{nad}}$, where $Sign = h(NAD_j \| GK_{nad} \| T_N \|$ Track No.). Furthermore, it should be noted that, the key idea behind fog computing is to deploy several NADs, to nearby end users. Such that, end users can get better prominence especially in terms of latency and throughput. In that case, we assume that all the NADs belongs to CCS shares a group key GK_{nad}. This group key can be used to validates the legitimacy of the group members. Here, we further assume that, for secure execution of this protocol (LAAP3) NADs need to help each other to check the validity of ED$_i$. However, NADs should not collude each others to collect the footprint of shares a group key GK_{nad}. This group key can be used to validates the legitimacy of the group members. Here, we further assume that, for secure execution of this protocol (LAAP3) NADs need to help each other to check the validity of ED$_i$. This phase of the authentication process consists of the following steps:

Step1: ED$_i$ → NAD$_i$: M_{C_1}: {*Track No.*, Req$_{\text{ED}}$,N_e, Token}.

ED$_i$ generates a nonce N_e and computes Req$_{\text{ED}}$ =*h(Track No.* $\|$ N_e $\|$ *CK).* Finally ED$_i$ constructs a request message M_{C_1} and sends to NAD$_i$.

Step 2: NAD$_i$→NAD$_j$: M_{C_2}:{*TrackNo.*, Req$_{\text{ED}}$,*Tseq*, NAD$_i$ *R*eq$_{\text{NAD}}$}.

Upon receiving the message M_{C_1}, NAD_i first decrypts the Token and gets the information about NAD i.e.NAD$_j$, to whom ED$_i$ has most recently visited. Hereafter,NAD$_i$ checks the time stamp (T_{NAD}) and the signature (*Sign*) inside the token, in order to validate Token. If any of the parameter inside the Token is invalid then NAD$_i$ terminates the execution of this protocol. Otherwise, NAD$_i$generates a nonce N_d and computes Req$_{\text{NAD}}$ =*h(*NAD$_i$ || N_d || GK_{nad}). Finally, NAD_i forms a request message M_{C_2} and sends it to NAD$_j$.

Step3:NAD$_j$→NAD$_i$: M_{C_3}:{*e1*, *e2*, Res$_{\text{ED}}$, Res$_{\text{NAD}}$}.

Upon receiving the message M_{C_2},NAD$_j$ first checks the Track No., Res$_{\text{ED}}$, Res$_{\text{NAD}}$whether they are valid or not. If not, then the system (NAD$_j$) will terminate the execution of this protocol. Otherwise, NAD$_j$ randomly generates a communication key CK_{new} and computes Res$_{\text{NAD}}$ = $h(e1||\ e2\ ||\ GK_{nad})$, Res$_{\text{ED}}$ = $h(e1||$ *CK* || *Track No.*). At the end, NAD$_j$forms a response message M_{C_3} and sends the message to NAD$_i$.

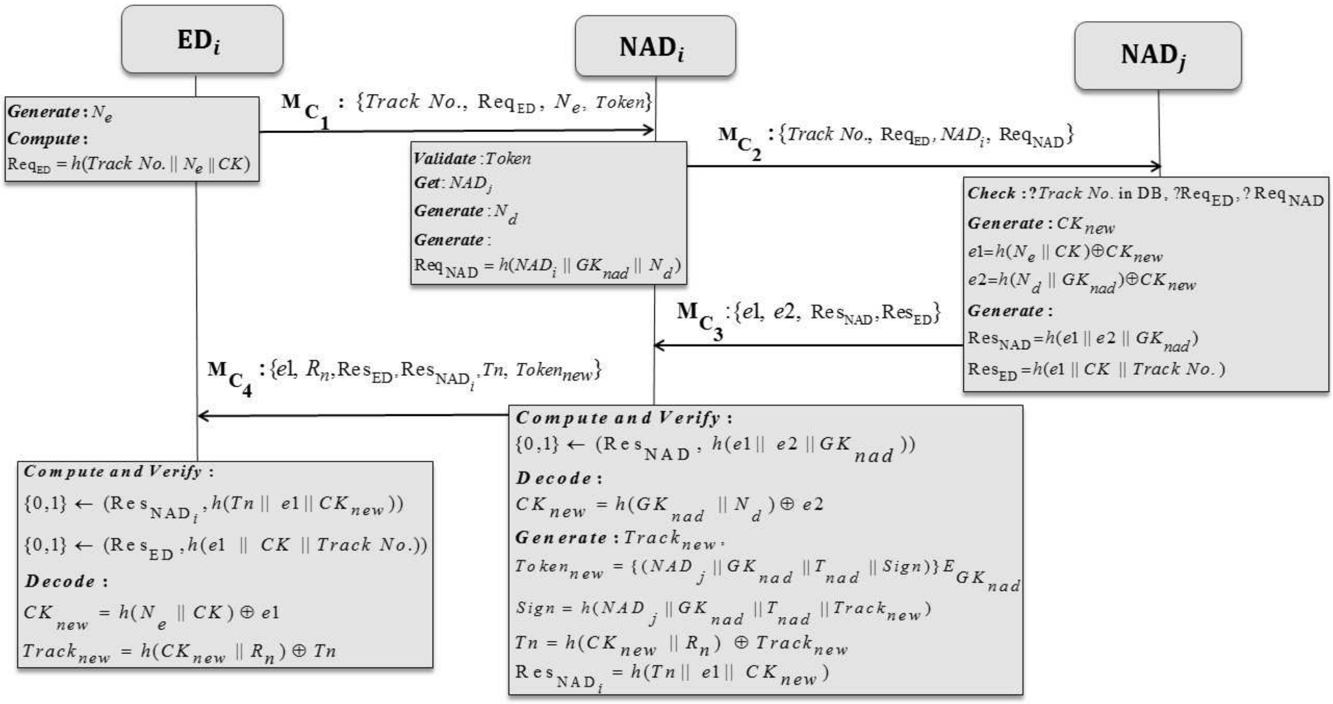
Step4: NAD$_i$→ED$_i$:M_{C_3}:{e1, R_n, Res$_{\text{ED}}$, Res$_{\text{NAD}_\text{i}}$, *Tn*, $Token_{new}$}.

After receiving M_{C_3}, NAD$_i$ first validates the response parameter Res$_{\text{NAD}}$ and subsequently decodes the communication key CK_{new} from *e*2. Hereafter, NAD$_i$ generates a new track number$Track_{new}$, random number R_n, and a token $Token_{new} = \{(NAD_i||\ T_{nad}\ ||\ Track_{new}\ ||\ Sign)E_{GK_{nad}}$, where $Sign = h(NAD_i||\ GK_{nad}||\ T_{nad}||\ Track_{new})$. After that, NAD$_i$ computes $Tn = h(CK_{new}\ ||\ R_n) \oplus Track_{new}$, Res$_{\text{NAD}_i}$ = $h(Track_{new}||\ CK_{new}||\ R_n)$ and forms a response M_{C_4} and sends to ED$_i$. It should be noted that, the parameter $Token_{new}$ will be useful once the ED$_i$moves to a new NAD.

Step5: Verification at ED$_i$.

After receiving M_{C_4}, ED$_i$ first validates the response parameters (Res$_{\text{ED}}$, Res$_{\text{NAD}_i}$) received from both the NAD$_j$and NAD$_i$, respectively. If the verification is successful then ED$_i$first decodes CK_{new}, $Track_{new}$ and keep them for further communication. Otherwise, ED$_i$ aborts the execution of this protocol (LAAP3). The detail procedure of this Phase is depicted in Fig. 4.

It should be noted that, since both LAAP2 and LAAP3 can support secure handover of an UE (user equipment) in mobile communication, they can be alternatively used. The choice among them depends on the performance and network environment. For example, in 5G, the establishment of a secure communication using LAAP3 may provide low latency but it requires an interaction between NADs via the interface X2, which causes higher communication cost as compared to LAAP2 . The secure communication process of LAAP2 could be more feasible when UE can find any authenticated UE to prove its legitimacy. It is argued that the concept of cooperative interactions among the NAD-to-NAD and ED-toED will not only resolve the performance bottleneck issue in conventional fog computing but also support the security requirements raised by the security architecture for D2D-aided fog computing paradigm.



$\mathbf{ED}_i$
$\mathbf{NAD}_i$
$\mathbf{NAD}_j$
Generate: N_e
Compute:
$\text{Req}_{ED} = h(Track\ No. \| N_e \| CK)$
M_{C_1} : {Track No., Req_{ED}, N_e, Token}
Validate: Token
Get: NAD_j
Generate: N_d
Generate:
$\text{Req}_{NAD} = h(NAD_i \| GK_{nad} \| N_d)$
M_{C_2} : {Track No., Req_{ED}, NAD_i, Req_{NAD}}
Check: ?Track No. in DB, ?Req_{ED}, ?Req_{NAD}
Generate: CK_{new}
$e1=h(N_e \| CK)\oplus CK_{new}$
$e2=h(N_d \| GK_{nad})\oplus CK_{new}$
Generate:
$\text{Res}_{NAD}=h(e1 \| e2 \| GK_{nad})$
$\text{Res}_{ED}=h(e1 \| CK \| Track\ No.)$
M_{C_3} : {e1, e2, Res_{NAD}, Res_{ED}}
M_{C_4} : {e1, R_n, Res_{ED}, Res_{NAD_i}, Tn, $Token_{new}$}
Compute and Verify:
$\{0,1\} \leftarrow (\text{Res}_{NAD}, h(e1 \| e2 \| GK_{nad}))$
Decode:
$CK_{new} = h(GK_{nad} \| N_d) \oplus e2$
Generate: $Track_{new}$,
$Token_{new} = \{(NAD_j \| GK_{nad} \| T_{nad} \| Sign)\}E_{GK_{nad}}$
$Sign = h(NAD_j \| GK_{nad} \| T_{nad} \| Track_{new})$
$Tn = h(CK_{new} \| R_n) \oplus Track_{new}$
$\text{Res}_{NAD_i} = h(Tn \| e1 \| CK_{new})$
Compute and Verify:
$\{0,1\} \leftarrow (\text{Res}_{NAD_i}, h(Tn \| e1 \| CK_{new}))$
$\{0,1\} \leftarrow (\text{Res}_{ED}, h(e1 \| CK \| Track\ No.))$
Decode:
$CK_{new} = h(N_e \| CK) \oplus e1$
$Track_{new} = h(CK_{new} \| R_n) \oplus Tn$

Figure 5: Subsequent Authentication Protocol with the Co-operation of NADs in D2D-Aided Fog Computing

4 Informal Security Analysis:

In this section, we will demonstrate that our proposed scheme can ensure various imperative security requirements such as mutual authentication and key exchange, privacy against eavesdropper, etc. which are greatly important in fog computing.

In LAAP 1:

Accomplishment of Mutual Authentication: CCS authenticates ED_i by using the parameters such as $Tseq$ and AID. If any of the parameters is invalid then CCS terminates the authentication process. On the other hand, using the legitimate response parameters Res_{CCS} in M_{A_4}, ED_i authenticates CCS and NAD, respectively. The security of mutual authentication is based on AID and Res_{CC}, where $AID = h(ID_{ED_i}||K_{ec}||Tseq)$ and $Res_{CCS} = h(e1||e2||K_{ec})$, K_{ec} is the long-term secret only known by CCS and ED_i, $e_1 = h(K_{ec}||Tseq) \oplus Tseq_{new}$, and $e_2 = h(K_{ec}||ID_{ED_i}||N_e) \oplus CK$. Since $Tseq$ is only known by ED_i and CCS and changed in every session, and e_2 is computed with a nonce N_e, they can be regarded as challenges of mutual authentication to CCS and ED_i to prevent from replay attacks, respectively. Moreover, we consider $h(\cdot||K_{ec}||\cdot)$ and $h(\cdot|| \cdot ||K_{ec})$ as two pseudo-random functions, where the outputs of pseudo-random functions are indistinguishable from the outputs of random functions. Hence, with the known inputs, ID_{ED_i}, $Tseq$, $e1$, and $e2$, the probability of producing the corresponding outputs without those pseudo-random function is negligible.

Accomplishment of Key Exchange: In our proposed mutual authentication and key exchange scheme LAAP1, CCS generates communication key CK and then distributes $\text{ED}_i$and NAD through the encoded parameter *e2* and secure channel, respectively. As discussed, after secure key-exchange, for communication security, both $\mathrm{ED}_i$and NAD can use *CK* to ensure privacy, integrity, and freshness of the sensitive data.

Accomplishment of Privacy Against Eavesdropper (PAE): The proposed scheme LAAP1 has maintained the one-time-alias feature (using AID), where there is no direct relationship between the aliases. Furthermore, since most of the parameters exchanged between the participants are one-time. This approach of the proposed scheme is quite effective for ensuring PAE support.

In LAAP 2:

Accomplishment of Mutual Authentication: NAD authenticates ED_j through the assistance of the most recently communicated edge deviceED_i. In that case, NAD needs to verify the legitimacy of the request parameter $\mathrm{Req}_{\mathrm{ED}_i}$. ED_i authenticates NAD using the response parameter $\mathrm{Res}_{\mathrm{NAD}}$. On the other hand, ED_i can verify the legitimacy of ED_j, using the request parameter *GAuth,* which can verify whether ED_j belongs to the same group as ED_i. Now, ED_jauthenticates ED_i as well as NAD using the response parameter $\mathrm{Res}_{\mathrm{ED}_i}$. In this way, all three participants in LAAP2 authenticate each other.

Accomplishment of Key Exchange: In LAAP2, NAD generates a new communication key CK_{new} and then encodes that by using the temporary key *tk* , which is generated by ED_i. In order to get *tk*, ED_i needs to use its group key *GK.*

Accomplishment of Privacy Against Eavesdropper (PAE): Similar to LAAP1, LAAP2 ensures PAE using the *AID* approach. Besides, like LAAP1, all the messages exchanged between the participants are random or one-time. Hence, an outsider cannot trace the communication. In this way, we achieve PAE for ED_i.

In LAAP 3:

Accomplishment of Mutual Authentication: We first recall the environment in LAAP3. ED_imoves from NAD_j to NAD_i, then the current NAD NAD_i needs to authenticate $\mathrm{ED}_i$with the help ofNAD_j. In that case, NAD_j authenticates ED_i using the request parameter $\mathrm{Req}_{\mathrm{ED}}$, where the legitimacy of the parameter is validated by using *CK.* NAD_j and NAD_i authenticates each others using the request parameter$\mathrm{Req}_{\mathrm{NAD}}$, and response parameter $\mathrm{Res}_{\mathrm{NAD}}$, respectively, where the legitimacy of these parameters are based on the group key GK_{nad}. Now, ED_i authenticates both NAD_i and NAD_j using the parameters $\mathrm{Res}_{\mathrm{ED}}$, $\mathrm{Res}_{\mathrm{NAD}_i}$, respectively.

Accomplishment of Key Exchange: In LAAP3, NAD_j generates a new communication key CK_{new} and then distributes the encoded CK_{new}among NAD_i and ED_i. In order to decode that, ED_ineeds to use the valid *CK,* and NAD_i needs to use the valid group key GK_{nad}.

Accomplishment of Privacy Against Eavesdropper (PAE): To communicate with NAD_i, ED_i uses the *Track No.*, only NAD_j can recognize that. Besides, all the parameters in $M_{C_1}$are one-time, therefore it will be difficult for an outsider to trace ED_i.

It should be noted that, to support location privacy feature in our proposed security architecture for D2D-aided fog computing, each time ED_i needs to choose LAAP1 to interact with NAD. Since, all the parameters in $M_{A_1}$are one-time. Therefore, even if ED$_i$ has already been interacted with a particular NAD before, but, it cannot comprehend the ED$_i$. However, in this regard, each time CCS needs to be involved. Therefore, it impairs the performance of the system. Furthermore, in LAAP2, since EDs are assumed to be fixed objects. Hence, we need not to consider the location privacy feature in this scheme.

5 Formal Security Analysis of the Proposed Scheme

Bellare and Rogaway [10] first proposed the theoretical security proof for an authentication and key exchange protocol with a symmetric two-party case, which is known as BR93-Model. Now, in our LAAP1 scheme, the communication between each serving NAD and the CCS is assumed to be secure, so that serving NAD and CCS can be regarded as a single network. On the other hand, in LAAP2, although two EDs in a group are involved, however, NAD needs to proof the legitimacy to last communicated ED. In this case, we can regard ED$_i$ and ED$_j$ as single entity. Thus both of our security model will fit the symmetric two-party setting on BR93-Model.

5.1 Complexity Assumptions

The security of our proposed scheme is based on the secure one-way hash function, which can be regarded as a pseudo-random function [10]. Therefore, we first introduce the security definitions of pseudo-random function and show their game environments for the simulations of the security proofs of the proposed protocols. The simulation of the security proofs have to follow the security definitions and the game environments of the pseudo random function.

Definition 1 *Let f be a polynomial-time computable function and $AdvH = |Pr\left[H^f = 1\right] - Pr\left[H^{f'} = 1\right]|$ denote the advantage of that an algorithm H, controlled by a probabilistic polynomial-time adversary $\mathcal{A}$, distinguishes ffrom another functionf'. We say that f is a (t, q, ∈)-secure pseudo-random function if no feasible algorithm H, making at most q oracle queries to f or a truely random function $f'$and running at most t by plying the following game, can distinguish f from f' with the advantage AdvH≥∈, where q is the polynomial number of oracle calls of $\mathcal{A}$.*

Initialization: *A challenger $\mathbb{C}$ interacting with A picks a random bit b∈ {0, 1} to determine the function f_b where f_0 is a pseudo-random function and f_1 is a truely random function [7-8].*

***Training Phase:**$\mathcal{A}$ issues q queries with $x_1,...,x_q$ to $\mathbb{C}$, wher e $x_i \in$ {0,1}*. The challenger responds these queries by sending $f_b(x_i)$ to $\mathcal{A}$ for i=1,...,q. where $f_b(x_i) \in \{0,1\}^l$ andl is a fixed positive integer.*

Guess: *A outputs* $b^{'} \in \{0,1\}$ *as a guess of* b*,* $\mathcal{A}$*wins this game if* b'=b*. We define the advantage of* $\mathcal{A}$*winning the game as* $Adv_{f_0,\mathcal{A}}$= $|Pr\left[b^{'}=b\right]-\frac{1}{2}|$.

According to the pseudo-random function assumption no probabilistic polynomial-time adversary can win the above game with non-negligible advantage.

5.2 Security Model and Notations

Protocol Participants. $\prod_{X,Y}^{s}$ denotes the oracle which plays the role of X to interact with Y in session s, and $\prod_{X,Y}^{t}$ denotes the oracle which plays the role B to interact with A in session t, where $X,Y\in I$, s$\in N$, I being the set of identities of the of the players (such as EDs, NAD, and CCS) who participate in the protocol and N being the set of positive integers.

Protocols. As we mentioned before, although the proposed protocols (LAAP1 and LAAP2) are three-party authentication and key exchange scheme. However, the protocols can be reduced as de-facto two-party setting protocols. Therefore, we define a two-party authentication and key exchange protocol.

Definition 2 A two-party authentication and key exchange protocol P, can be formally specified by an efficiently computable function $\prod$ with the following inputs:

k: Security parameter length used in the protocol.

X: Initiator's identity of P, where $X \in I$.

Y: Intended partner's identity P, where $Y \in I$.

a: Secret information, where $a\in$ {0, 1}*.

$\mathfrak{K}$ The conversation in P so far.

r: The random coin flips of the sender or initiator, where $r \in 0,1^{+}$

The output of $\prod$ (k, X,Y, a, $\mathfrak{K}$, r) = (m, δ, α) can be defined as follows:

m: The subsequent message to be sent, where $m \in$ {0, 1} $\bigcup$ {*}, where * denotes that the initiator sends no message.

δ: The decision, where $\delta\in$ {$\mathbb{A}$, $\mathbb{R}$, *}, where $\mathbb{A}$, $\mathbb{R}$, * denote accept, reject, and no decision, respectively.

α : The private output, where $\alpha \in 0,1$*$\bigcup${*}, * denotes that the initiator does not have any private output.

5.3 Adversary Model

While execution of the protocol P, an adversary $\mathcal{A}$, who is a probabilistic polynomial-time Turing machine, can be able to control the communication channel between X and *Y,* by eavesdropping the messages sent by X and Y and modifying the messages produced by X, and *Y*, and compromise session secrets shared between X, and Y in the real environments. These behaviors can be modeled by the following queries.

Execute($\prod_{X,Y}^{s}$, $\prod_{Y,X}^{t}$): The query models all kinds of passive attaks, where a passive adversary can intercept all the data exchanged between $\prod_{X,Y}^{s}$ and $\prod_{Y,X}^{t}$ in a session of P.

Send($\prod_{X,Y}^{s}$, m): This query models active attacks, where an adversary sends a message m to $\prod_{X,Y}^{s}$ and obtains the response message according to the proposed scheme.

Reveal($\prod_{X,Y}^{s}$): The query models the exposure of session key (known session key attacks) in a particular session s.

Corrupt($\prod_{X,Y}^{s}$): This query models the revelation of the long-term secret key. This query models passive attack.

Test($\prod_{X,Y}^{s}$): When $\prod_{X,Y}^{s}$ accepted and shared a session key, adversary $\mathcal{A}$ can make this query and try to distinguish a real session key from a random string.

5.4 Security Definitions

Mutual Authentication Security: First we briefly review the definition of matching conversation [10-11].

Definition 3(Matching Conversations): *An authenticated key exchange protocol P is a message-driven protocol and the goal of P is to achieve matching conversation. We first define a protocol session within a party X as (X, Y, s,* `role`*) where Y is the identity of X's partner, s is a session id, and* `role` *can be either* `initiator` *or* `responder`*. A P of two protocol sessions within a partyX and a party Y are of the form (X, Y, s,* `initiator`*) and (X, Y, t,* `responder`*), respectively. Two session are said to be matching conversation involving X and Y if their session id's are identical and the initiator and responder are each of them. A protocol P consists of more than two sessions and each pair of sessions in sequence is matching conversation. P is said to be a protocol of matching conversation. The definition of matching conversation is of the same functionality as defined in* [10-11].

Now, we define mutual authentication based on the definition of matching conversation as follows. P is a mutual authentication protocol if for any polynomial time adversary $\mathcal{A}$, 1) matching conversation implies acceptance and 2) acceptance implies matching conversation. The fist condition says that if the sessions of two parties consists of a matching conversation, the parties accept the authentication of each other. The second condition says that if each party accepts the authentication with the other party in a conversation, the probability of no matching conversation is negligible.

Definition 4 *An authentication protocol P is* **MA-Secure** *(i.e. P satisfies* **MA-Security***) if*

(1)Matching conversation implies acceptance:

If oracles $\prod_{X,Y}^{s}$ and $\prod_{X,Y}^{t}$ have matching conversations, both oracles accept and

(2)Acceptance implies matching conversations:

The probability of $No - Matching^{E}(k)$ is negligible, where k is a security parameter and $No -$

$Matching^E(k)$ is the event that their exists i, j, X , Y *such that $\prod_{X,Y}^{i}$ accepted but there is no oracle $\prod_{Y,X}^{j}$ which is engaged in a matching conversation.*

The event $No - Matching^E(k)$ can also be denoted as *$Succ_P^{MA}(\mathcal{X})$* which is the probability of that a polynomial-time adversary $\mathcal{X}$ can successfully impersonate one of the two interactive entities who want to authenticate each other in P.

Authentication Key Exchange (AKE) Security: During the execution of an **MA-Secure** authentication protocol P, a polynomial-time adversary $\mathcal{X}$ interacts with two fresh oracles: $\prod_{X,Y}^{s}$ and its partner$\prod_{Y,X}^{t}$. At the end of the execution, $\mathcal{A}$ issues a *Test* query to one of the two fresh oracles. Then the real session key or a random string is returned to $\mathcal{X}$ according to the value of a random bit b. Finally, $\mathcal{X}$ outputs a bit b' and terminates the game. The **AKE-Advantage** $Adv_P^{AKE}(\mathcal{A})$ is defined as $|\Pr[\mathrm{b} = \mathrm{b}'] - 1/2|$. We give a formal definition of **AKE-Security** below:

Definition 5 *A protocol P is* **AKE-Secure** *if P satisfies the following properties:*

(1)A being adversary engages in the execution of P with $\prod_{X,Y}^{s}$ and its partner $\prod_{Y,X}^{t}$. Then both oracles can accept and share the same session key each other.

(2)P is **MA-Secure.**

(3)For every probabilistic polynomial-time adversary $\mathcal{X}$, $Adv_P^{AKE}(\mathcal{X})$ is negligible.

When a Test query is issued before finishing the execution of the protocol, the game is played as the above definition if the session key is generated by any one of the two fresh parties. Otherwise, the Test query wil be rejected.

5.5 Security Proofs

Our proposed scheme is based on hash function, that we can consider as secure pseudo-random function. In this sub-section, we will show that the proposed protocols are provably secure based on pseudo-random function assumption. As discussed above, even though both of our protocols are three-party authentication and key exchange protocol, however, they can be reduced into a two-party authentication and key exchange protocol. Therefore, in this subsection we prove the security of LAAP1, and in the similar fashion, we can prove the security evidence for the LAAP2.

Lemma 1 If h is a (t_0, q_0, ε_0)-secure pseudo-random function family with negligible ε_0, the proposed LAAP1 is **MA-Secure**.

Proof. Assume that there is a polynomial-time adversary $\mathcal{A}$ who can break MA-Security of the proposed protocol P with non-negligible probability *$Succ_P^{MA}(\mathcal{A})$*. We conduct a polynomial time algorithm $\mathcal{F}$ using $\mathcal{A}$ to show that $\mathcal{F}$ can break the pseudo-random function with non-negligible advantage, too. Besides, $Succ_{\mathcal{A}}$is *$Succ_P^{MA}(\mathcal{A})$* = Pr[$Succ_E$]+ Pr[$Succ_N$]- Pr[$Succ_E, Succ_N$] ≤Pr[$Succ_E$]+Pr[$Succ_N$], where $Succ_E$ and $Succ_N$, respectively, are the events that $\mathcal{A}$ successfully impersonates as a legitimate ED and networks, respectively, to pass authentication. Therefore, we split the proof into two cases.

One is networks impersonation and the other is the ED impersonation.

Case1(Networks Impersonation):

Assume that $\mathcal{A}$ can impersonate as networks with the probability ϵ'. If $\mathcal{A}$ wants to be successfully authenticated by $\prod_{E,N}^{s}$ controlled by $\mathcal{F}$, $\mathcal{A}$ must send correct $Res_{CCS} = h(e1||\ e2\ ||\ K_{ec})$.

In the following game , $\mathcal{F}$ will exploit the ability of $\mathcal{A}$ to break the pseudo-random function assumption with $\epsilon' \leq 4\epsilon_0 + 2^{-k}$, where k is the security parameter. $\mathcal{F}$ plays the game in **Definition 1** with challenger $\mathcal{C}$.

Initialization. Let the size of the long-term secret key K_{ec} , in LAAP1 be k-bit long. $\mathcal{C}$ selects a random bit b $\in$ {0,1} and set up a secure one-way hash function h_b where $h_0 = h_{K_{ec}}$is a pseudo-random function and $h_1$is a random function. If $\mathcal{F}$ simulates the game by using h_1 to interact with $\mathcal{A}$, we denote this game as a *random experiment.* If $\mathcal{F}$ uses h_0 to simulate the game, we call this game as real experiment. The goal of $\mathcal{F}$ is to correctly guess h_b =h_0 or h_1 (i.e. $b\ =\ 0\ or\ 1$).

Training. $\mathcal{F}$ simulates $\prod_{E,N}^{s}$, and $\prod_{N,E}^{t}$ to interact with $\mathcal{A}$ by answering the following queries.

•*Execute* ($\prod_{E,N}^{s}$, $\prod_{N,E}^{t}$): $\mathcal{F}$ uses h_b provided by $\mathcal{C}$ as$h_{K_{ec}}$in the protocol. $\mathcal{F}$ also randomly generates CK and $Tseq_{new}$ and then computes $e1$= $h(K_{ec}||\ Tseq){\oplus}Tseq_{new}$, $e2$= $h(K_{ec}\ ||\ ID_{\mathrm{ED}_i}) \oplus CK$, $Res_{CCS} = h(e1||\ e2\ ||\ K_{ec})$. Subsequently, $\mathcal{F}$ simulates $\prod_{E,N}^{s}$, $\prod_{N,E}^{t}$ with the help of h_b, *e1*, *e2,*and Res_{CCS}.

•*Send*($\prod_{E,N}^{s}$, m): $\prod_{E,N}^{s}$ sends the request message m = {AID, N_x, Req_{ED},$Tseq$}, of the protocol . $\prod_{E,N}^{s}$ first validate Req_{ED} by querying $h_b$and then finds the $Tseq$ in it's database and then checks the correctness of AID, by querying h_b.

•*Send*($\prod_{N,E}^{t}$, m): If m = {AID, N_x, Req_{ED},$Tseq$}, then $\prod_{N,E}^{t}$ computes $e1$= $h(K_{ec}||\ Tseq){\oplus}Tseq_{new}$, $e2$= $h(K_{ec}\ ||\ ID_{\mathrm{ED}_i}) \oplus CK$, $Res_{\mathrm{CCS}} = h(e1||\ e2\ ||\ K_{ec})$, $TN = h(CK\ ||\ R_n){\oplus}$*Track No.*, Res_{NAD} = $h($*Track No. // CK //* $R_n)$ by randomly selecting *Track No.*, R_n and querying h_b. $\prod_{N,E}^{t}$ then responds {$e1$, $e2$, Res_{CCS}, Res_{NAD}, TN, R_n} to $\mathcal{A}$.

Challenge. $\mathcal{A}$ firstly queries Send($\prod_{E,N}^{s}$, m) to trigger the protocol. $\prod_{E,N}^{s}$ then sends m = {AID, N_x, Req_{ED},$Tseq$} to $\mathcal{A}$. Then $\mathcal{A}$ generates the authentication response parameters Res_{CCS}, Res_{NAD} with the success probability $\Pr[Succ_N] = \epsilon'$. Thus $\mathcal{A}$ queries *Send*($\prod_{N,E}^{t}$, m = {$e1$, $e2$, Res_{CCS}, Res_{NAD}, TN, R_n}. After receiving them $\mathcal{F}$ issues two queries $x* = (e1||\ e2\ ||\ K_{ec})$, $y* = ($*Track No.* $\|\ CK\ \|\ R_n)$ to h_b and obtains the outputs $Res^{*}_{\mathrm{CCS}} = h(e1||\ e2\ ||\ K_{ec})$, $Res^{*}_{\mathrm{NAD}} = h($*Track No.* $\|\ CK\ \|\ R_n)$.

Guess. Finally, $\mathcal{F}$ outputs a guessing bit $b'$$\in$ {0,1}. If $Res^{*}_{\mathrm{CCS}} = Res_{\mathrm{CCS}}$, and $Res^{*}_{\mathrm{NAD}} = Res_{\mathrm{NAD}}$, then $\mathcal{F}$ outputs 0; otherwise, $\mathcal{F}$ outputs a random bit 0 or 1.

The probability of that $\mathcal{A}$ can send out the correct authentication message is ϵ' in the real experiment and 2^{-k} in the random experiment. Hence we have the following:

$$\Pr\left[\mathrm{b} = \mathrm{b'}\right] = \Pr\left[b = b', b = 0\right] + Pr\left[b = b', b = 1\right]$$

$$= (\epsilon' + (1-\epsilon')/2)1/2 + ((1-2^{-k})/2)1/2$$

$$= 1/2 + \epsilon'/4 - 2^{-(k+2)}$$

$$\in_0 \geq \left|Pr\left[b = b'\right] - 1/2\right| = \epsilon'/4 - 2^{-(k+2)}$$

$$\epsilon' \leq 4 \in_0 + 2^{-k}$$

The analysis of the probability of that $\mathcal{F}$ successfully distinguishes the given h_b(*i.e.b* = *b'*)can be divided into two cases (*b* = *b'*) under a real experiment i.e. b = 0, and ($b = b'$) under a random experiment, i.e. b =1. In case of real experiment, $\mathcal{A}$ can successfully send a correct authentication information to win the game with the probability ϵ'. Hence, $\mathcal{F}$ will output b' =0 with probability ϵ' when $\mathcal{A}$ sends a correct authentication information under a real experiment. Besides, if $\mathcal{A}$ sends a wrong information , $\mathcal{F}$ can only randomly guess b, i.e. $\mathcal{F}$ will output$b$' = 0 with probability (1-ϵ')/2. Therefore, the probability of (b = b') and (*b=0)* is $(\epsilon' + (1-\epsilon')/2)1/2$. In case of random experiment, $\mathcal{A}$ can only send the correct authentication information by randomly guessing with the probability 2^{-k} and thus $\mathcal{F}$ outputs *b'=1* with probability (1 - 2^{-k})/2. Therefore, the probability of (*b* = *b'*) and (b =1) is (1-2^{-k})/2. Therefore, the probability of (b = b') and (b =1) is ((1 - 2^{-k})/2)1/2.

Case2 (ED Impersonation):

Suppose that $\mathcal{A}$ can impersonate as ED with probability ϵ''. If $\mathcal{A}$ wants to be accepted by $\prod_{N,E}^{t}$, then $\mathcal{A}$ has to send out the correct authentication information. $\mathcal{F}$ plays the game which is the same as Case1 with $\mathcal{C}$.

Initialization. $\mathcal{C}$ selects an hash function$h_b$according to a random bit b∈ {0, 1}for answering the queries from $\mathcal{F}$ where $h_0 = h_{K_{ec}}$is a pseudo-random function and $h_1$is a random function.

Training. $\mathcal{F}$ firstly selects the required N_e, $Tseq$ in the protocol. $\mathcal{F}$ then simulates $\prod_{E,N}^{s}$, and $\prod_{N,E}^{t}$ by answering *Execute*($\prod_{E,N}^{s}$, $\prod_{N,E}^{t}$), *Send*($\prod_{E,N}^{s}$, m). The simulations of these oracles are similar to those in Case 1.

Guess. $\mathcal{F}$ outputs a guess b'∈ {0,1} according to AID and Req_{ED}. If AID = h_k(ID_{ED_i}|| K_{ec}|| *Tseq)* and Req_{ED} = *h*(*AID* || N_e|| K_{ec}), then $\mathcal{F}$ outputs 0, that means $h_b = h_{K_{ec}}$; otherwise it outputs a random bit 0 or 1.

The probability of that $\mathcal{A}$ successfully sends out the correct

AID = h_k(ID_{ED_i}||K_{ec}||*Tseq)* and Req_{ED} = *h*(*AID* || N_e|| K_{ec}) is Pr[$Succ_E$]= 2^{-k} in the random experiment. Hence, we have $\Pr[b = b'] = 1/2 + \epsilon''/4 - 2^{-(k+2)}$, *and* ϵ''/4 - $2^{(k+2)}$, and $\epsilon'' \leq 4\epsilon_0$+ 2^{-k}.

The analysis of Pr$[b = b']$is similar to that in the proof. of **Lemma 1**.

By Case 1 and Case 2,

$$Succ_P^{MA}(\mathcal{A}) \leq Pr\left[Succ_N\right] + Pr\left[Succ_E\right] = \epsilon' + \epsilon'' \leq 8\epsilon_0 + 2^{-(k-1)}$$

From the above, $\in_0$ is non-negligible, which contradicts that $\in_0$ is negligible. It turns out that proposed LAAP scheme is **MA-Secure**.

Lemma 2 If h is a (t_0, q_0, ε_0)-secure pseudo-random function family with negligible ε_0, the proposed LAAP1 is **AKE-Secure**.

Proof. In **Lemma 1** we prove that the proposed protocol P is **MA-Secure**. Conceive, $\mathcal{A}$ is an adversary who can break **AKE-Security** of P with non-negligible $Adv_P^{AKE}(\mathcal{A})$ $=\epsilon$. We construct a simulator $\mathcal{F}$ using the ability of $\mathcal{A}$ to break the pseudo-random function assumption with $\epsilon \leq \epsilon_0$. $\mathcal{F}$ plays the game in **Definition 3** with a challenger $\mathcal{C}$.

Initialization. $\mathcal{C}$ picks a random bit $b\{0, 1\}$ and sets up a secure hash function h_b for answering the queries from $\mathcal{F}$ where $h_0 = h_{K_{ec}}$is a pseudo-random function andh_1 is a random function.

Training. $\mathcal{F}$ selects the required N_e, $Tseq$ in the protocol. $\mathcal{F}$ then simulates $\prod_{E,N}^s$, and $\prod_{N,E}^t$ by answering $Execute(\prod_{E,N}^s, \prod_{N,E}^t)$, $Send(\prod_{E,N}^s,\ m)$. The simulations of these oracles are similar to those in the proof of **Lemma 1.**

•Test($\prod_{E,N}^s$): If CK of $\prod_{E,N}^s$ is generated , $\mathcal{F}$ randomly chooses $c \in$ {0,1}, and returns the real session key CK for $c = 0$ or a random string for $c = 1$. Otherwise, $\mathcal{F}$returns $\perp$denoting meaninglessness.

•Test($\prod_{N,E}^t$): The simulation is the same as the above one.

Challenge. After querying $Execute(\prod_{E,N}^s, \prod_{N,E}^t)$, $\mathcal{A}$ sends a *Test* query to $\mathcal{F}$.

Guess. After querying $Test(\prod_{E,N}^s)$ or Test($\prod_{N,E}^t$), $\mathcal{A}$ outputs a bit c'=0. if $\mathcal{A}$ thinks that the responding string is the real session key; otherwise, c'=1. Finally, $\mathcal{F}$ outputs b' = 0 if c' = c; otherwise $\mathcal{F}$ outputs b' = 1. Thus we have the following:

$$\Pr\left[b = b'\right] = Pr\left[b = b', b = 0\right] + \Pr\left[b = b', b = 1\right]$$

$$= 1/2.\epsilon + 1/4 + 1/4 = 1/2 + \epsilon/2$$

$$\epsilon_0 \geq \Pr\left[b = b'\right] - 1/2 = \epsilon/2$$

The analysis of the probability of (b = b') is also similar to that in the proof of **Lemma 1**. $\mathcal{A}$ can win the game by successfully guessing (c = c') with probability $(\epsilon + 1/2)$ under a real experiment (b = 0). $\mathcal{A}$ can only guess (c = c') randomly 1/2 under a random experiment (b =1). If $\mathcal{A}$ successfully guesses (c = c'), $\mathcal{F}$ will output b' =1. Therefore, the probability of (b = b') and (b = 0) is $(\epsilon+1/2)/2$, and the probability of (b = b') and (b = 1) is 1/4. From the above, $\epsilon_0$is non-negligible, and thus a contradiction occurs. Therefore, $Adv_P^{AKE}(\mathcal{A})$ is negligible for each polynomial-time adversary $\mathcal{A}$ and P is **AKE-Secure**.

Lemma 3 If h is a (t_0, q_0, ε_0)-secure pseudo-random function family with negligible ε_0, the proposed scheme can ensure strong anonymity support.

Proof. In our proposed scheme, both the pseudo identity and one-time-alias identity with transaction sequence number can ensure strong anonymity support, which is the combination of the identity privacy and untraceability. There is not direct relationship between the aliases, where each one-time-alias identity is generated based on a secure hash function *h,* which is a pseudo-random function and in **Lemma 1** and **Lemma 2** it is proved that the ability of $\mathcal{A}$ to break the pseudo-random function is negligible. Besides, it can also be noticed that, during the execution of LAAP1 and LAAP2, none of the parameter in the request message M_{A_1}, and $M_{B_1}$are allowed to to be sent twice. This approach of the proposed scheme is quite effective for privacy against eavesdropper (PAE) to achieve.

6 Performance Evaluation

In this Section, we benchmark the performance of the proposed scheme to show that our privacy preserving approach for fog paradigm is efficient and hence can be useful for various critical IoT applications and services.

As we mentioned before that, in our proposed scheme we achieve strong anonymity by computing one-time alias identity $AID = f(ID_{\mathrm{ED}_i}||K_{ec}||Tseq)$, where $f(.)$ is a secure one-way hash-function. However, there are many ways to accomplish strong anonymity support for an entity, where $f(.)$ can be regarded as one of the following: symmetric key encryption AES-CBC (used as $f(.)$ in [12]), asymmetric key encryption ECIES (used as $f(.)$ in [13]), modular exponential operation (used as $f(.)$ in [14]), Chinese reminder theorem (used as $f(.)$ in [15-16]), and the pairing operation (used as $f(.)$ in [17-18]). In this Section, we demonstrate that our proposed privacy preservation technique is more feasible for the fog computing environment. In order to do that, here we simulate several cryptographic primitives used to achieve anonymity in the proposed scheme and others [12-17], using Java Cryptography Extension (JCE) [19] on a smartphone of HTC One X as a testbed. The smartphone runs Androids 4.1.1 mobile OS and equipped with 1.5 GHz quad-core ARM Cortex-A9 CPU and 1 GB RAM. For readers' reference, Table 3 gives the execution time of all the cryptographic operations. From Table 3, it is clear that secure one-way hash-function will cause less execution time than other cryptographic operations, hence can ensure expeditious validation in our privacy preserving fog computing model.

Table 3: Computational Overhead of the Various Cryptographic Operations

Cryptographic Operation	**Execution Time**
Hash Operation (SHA-256)	0.015 ms
Symmetric Key Encryption (AES-CBC)	0.027ms
Asymmetric Key Encryption (ECIES)	1.387 ms
Modular Exponendital Operation with D-H	0.76 ms
Chinese Reminder Theorem	0.63 ms
Pairing Operation	6.827 ms

6.1 Computation and Communication Cost of the Proposed Protocols

Now, we evaluate the required computational costs for all three anonymous authentication protocols (LAAP1, LAAP2, and LAAP3) for fog computing models. To analyze performance of the LAAP1, LAAP2, and LAAP3 more comprehensively, here we conduct an experiment. Table 4 lists our experimental environment, including hardware specifications, used algorithm, link type, and average transmission time. Table 5 shows that because of the symmetric encryption/decryption during Token generation, the computational complexity of LAAP3 is little-bit higher than both LAAP1 and LAAP2. On the other hand, even though all LAAP1, LAAP2, and LAAP3 require the same number of data flows (four). However, during validation process in LAAP2, and LAAP3 neither EDs nor the NAD needs to communicate with the CCS, which is assumed to be placed far from NAD and edge devices. In this way, we minimize the communication cost [20] of the system in LAAP2, and LAAP3 which can be seen in Table 5. After analyzing both LAAP1, LAAP2 and LAAP3, here we categorize the performances of the proposed scheme into three cases, i.e. the best case, average case and the worst case.

•**Best Case:** It indicates the minimal cost during the execution of the authentication process in the fog computing model. In this case, we consider the successful execution of the initial authentication process through LAAP1 and all the subsequent authentication process by using LAAP2 or LAAP3. Therefore, for authenticating ten EDs in our proposed scheme the overall cost will be as follows: 1×LAAP1 + 9×LAAP2 = 443.19 ms or 1×LAAP1 + 9×LAAP3 = 443.92 ms.

•**Average Case:** It specifies the reasonable cost during the execution of the authentication process in the fog computing model. In this case, we consider that some of the subsequent authentication process through LAAP2 may failure. So, we assume 50% of the authentication process will be carried out through LAAP1 and rest 50% through LAAP2.. Hence, the overall cost in this case can be expressed as follows: 5×LAAP1 + 5×LAAP2 = 510.01 ms or 5×LAAP1 + 5×LAAP3 = 510.1 ms.

•**Worst Case::** It indicates the maximum cost during the execution of the authentication process, which may occur if all the subsequent authentication process by using LAAP2 ot LAAP3 are unsuccessful, therefore to authenticate ED, NADs require the support of CCS . Therefore, the overall cost for authentication in this case can be expressed as follows: 10×LAAP1 = 593.55 ms.

Table 4: The Experimental Environment

Hardware Specification	
ED: HTC One X with 1.5 GHz Max Turbo Frequency	
NAD: ThinkPad E460 with Intel Core i5-5200U and 2.2 GHz Max Turbo Frequency	
CCS: ASUS GR8-R047R with Intel Core i7-4510U and 3.1 GHz Max Turbo Frequency	
Computation Time (SHA-256)	**Computation Time (AES-CBC)**
ED: $T_{h_{\text{ED}}} = 0.015\text{ms}$	-
NAD: $T_{h_{\text{NAD}}} = 0.011\text{ms}$	**NAD:** $T_{Sym_{\text{NAD}}} = 0.023\text{ms}$
CCS: $T_{h_{\text{CCS}}} = 0.0092$ ms	-
Link Type	
ED-NAD: One-hop Wireless (802.11)	
NAD-CCS: Wired (Internet)	
Average Transmission Time	
ED-NAD: 10.62 ms	
NAD-CCS: 18.96 ms	

Table 5: Computation Cost of the Proposed Schemes

For LAAP1	ED$_i$	NAD	CCS
Computation Complexity	$7T_{h_{\text{ED}}}$	$3T_{h_{\text{NAD}}}$	$6T_{h_{\text{CCS}}}$
Computation Time	0.105 ms	0.033 ms	0.0552 ms
Total Computation Time	0.1932 ms		
Average Communication Cost in Time	2×10.62 +2×18.96 = 59.16 ms		
Total Cost for Execution	0.1932 +59.16 = 59.353		

.

.

For LAAP2	ED$_j$	ED$_i$	NAD
Computation Complexity	$5T_{h_{\text{ED}}}$	$6T_{h_{\text{NAD}}}$	$3T_{h_{\text{CCS}}}$
Computation Time	0.075 ms	0.09 ms	0.027 ms
Total Computation Time	0.192 ms		
Average Communication Cost in Time	4×10.62 = 42.48 ms		
Total Cost for Execution	0.192 + 42.48 = 42.67 ms		

.

.

For LAAP3	ED$_i$	NAD$_i$	NAD$_j$
Computation Complexity	$5T_{h_{\text{ED}}}$	$7T_{h_{\text{NAD}}}+2T_{Sym_{\text{NAD}}}$	$6T_{h_{\text{CCS}}}$
Computation Time	0.075 ms	0.123 ms	0.052 ms
Total Computation Time	0.25 ms		
Average Communication Cost in Time	4×10.62 = 42.48 ms		
Total Cost for Execution	0.25 + 42.48 = 42.73 ms		

7 Conclusion

In this article, first we have proposed a new security architecture for fog computing paradigm with some advance features like D2D communication, etc. Besides, this article has also discussed several security issues in the context of fog computing. Subsequently, we have proposed three lightweight anonymous authentication protocols (LAAPs) for conventional fog computing and D2D communication aided fog computing, respectively. Analyses show that our proposed scheme is secure and feasible for resource-limited devices in IoT.

Acknowledgements

The author would like to thank all the fantastic people who made this world a better, joyful and happy place to live in.

References

[1] J. G. Andrews et al., “What Will 5G Be?,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065-1082, June 2014. doi: 10.1109/JSAC.2014.2328098

[2] A. Osseiran, K. Doppler, C. Ribeiro, M. Xiao, M. Skoglund, and J. Manssour, “Advances in device-to-device communications and network coding for IMT-Advanced,” *ICT Mobile Summi*t, 2009.

[3] G. Galante and L. C. E. d. Bona, “A survey on cloud computing elasticity,” *Proc. Int. Conf. Utility Cloud Comput.*, pp. 263-270, 2012,

[4] 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 MCC Workshop on Mobile Cloud Computing*, ser. *MCC'12. ACM*, 2012, pp. 13-16.

[5] S. Yi , Z. Qin, Q. Li, “Security and Privacy Issues of Fog Computing: A Survey,” *10th International Conference, WASA 2015*, Qufu, China, August 10-12, 2015.

[6] A.Kumar, A. Aggarwal, “Lightweight Cryptographic Primitives for Mobile Ad Hoc Networks,” *Recent Trends in Computer Networks and Distributed Systems Security, CCIS,* Vol. 335, pp. 240-251.

[7] T. Hwang, P. Gope, “IAR-CTR and IAR-CFB: Integrity Aware Real-time Based Counter and Cipher Feedback Modes,” *Security and Communication Networks*, DOI: 10.1002/sec.1312, 2015.

[8] P. Rogaway, “Efficient Instantiations of tweakable Blockciphers and Refinements to Modes OCB and PMAC,” In Proceeding of the ASIACRYPT (2004), LNCS, Springer, Heidelberg, vol. 3329, pp. 16-31; 2004.

[9] S.-M. Cheng , W.-R. Lai , P. Lin and K.-C. Chen, "Key management for UMTS MBMS", *IEEE Trans. Wireless Commun.*, vol. 7, no. 9, pp. 3619-3628, 2008.

[10] M. Bellare and P. Rogaway, “Entity Authentication and Key Distribution," in Proc. of Advances in Cryptology - *CRYPTO’93*, 1993.

[11] R. Canetti and H. Krawczyk, “Analysis of Key-exchange Protocols and Their Use of Building Secure Channels," in Proc. of Advances in Cryptology - *EUROCRYPT’01*, 2001.

[12] S. Kumari et al., “User Authentication Schemes for Wireless Sensor Networks: A Review,” *Ad Hoc Networks*, vol. 27, no. 5, pp. 13581-13589, 2016.

[13] H. Mun, K. Han, Y. Lee, C. Yeun, and H. Choi, “Enhanced secure anonymous authentication scheme for roaming service in global mobile network,” *Mathematical and computer Modeling*, vol. 55, pp. 214-222, 2012.

[14] T. Zhou and J. Xu, “Provable secure authentication protocol with anonymity for roaming service in global mobility networks,” *Computer Networks*, vol. 55, pp. 205-213, 2011.

[15] Q. Jiang, J. Ma, G. Li, L. Yang, “An enhanced authentication scheme with privacy preservation for roaming services in global mobility networks,” *Wireless Personal Communications*, vol. 68, pp. 1477-1491, 2013.

[16] P. Gope, T. Hwang, “Lightweight and Energy Efficient Mutual Authentication and Key Agreement Scheme with User Anonymity for Secure Communication in Global Mobility Networks,” *IEEE Systems Journal*, Vol. 10 (4): pp 1370-1379, 2016.

[17] D. He, S. Chan and M. Guizani, “Handover authentication for mobile networks: security and efficiency aspects,” *IEEE Network*, vol. 29, no. 3, pp. 96-103 ,May-June 2015. doi: 10.1109/MNET.2015.7113232.

[18] S. Kumari, "Questioning Key Compromise Attack on Ostad-Sharif et al.’s Authentication and Session key Generation Scheme for Healthcare Applications", *IEEE Access*, DOI: 10.1109/ACCESS.2019.2905731.

[19] Oracle Technology Network, “*Java Cryptography Architechture (JCA)*,” http://docs.oracle.com/javase/6/docs/technotes/guides/security /crypto/CryptoSpec.html.

[20] P. Gope, and B. Sikdar, “Lightweight and Privacy-Friendly Spatial Data Aggregation for Secure Power Supply and Demand Management in Smart Grids,” *IEEE Transactions on Information Forensics and Security* , vol. 14 (6), pp. 1554-1566, 2019.

[21] P. Gope, and B. Sikdar "Lightweight and Privacy-Preserving Two-Factor Authentication Scheme for IoT Devices," *IEEE Internet of Things Journal*, DOI:10.1109/JIOT.2018.2846299, 2018.

[22] P. Gope, and B. Sikdar, "An Efficient Privacy-preserving Authentication Scheme for Energy Internet-based Vehicle-to-Grid Communication" *IEEE Transactions on Smart Grid*, DOI: 10.1109/TSG.2019.2908698, 2019.

[23] P. Gope et al. "Lightweight and Physically Secure Anonymous Mutual Authentication Protocol for Real-Time Data Access in Industrial Wireless Sensor Networks" *IEEE Transactions on Industrial Informatics*, DOI: 10.1109/TII.2019.2895030, 2019.

[24] R. Blom "An optimal class of symmetric key generation systems" *In Proc. of the EUROCRYPT 84 workshop on Advances in cryptology: theory and application of cryptographic techniques* , Springer-Verlag New York, Inc., New York, NY, USA, pp. 335-338.

[25] F.J. MacWilliams and N.J.A. Sloane "The Theory of error correcting codes" *North-Holland, New York, 1977.s* .

[26] G-S. Poh, P. Gope, and J. Ning, "PrivHome: Privacy-Preserving Authenticated Communication in Smart Home Environment" *IEEE Transactions on Dependable and Secure Computing*, DOI: 10.1109/TDSC.2019.2914911, 2019.

Author's biography

Prosanta Gope received the PhD degree in Computer Science and Information Engineering from National Cheng Kung University (NCKU), Tainan, Taiwan, in 2015. He is currently working as a Lecturer (Assistant Professor) in the Department of Computer Science (Cyber Security) at the University of Sheffield, UK. Dr. Gope served as a Research Fellow in the Department of Computer Science at National University of Singapore (NUS). His research interests include lightweight authentication, authenticated encryption, access control system, security in mobile communication and fog computing, lightweight security solutions for smart grid and hardware security of the IoT devices. He has authored over 50 peer-reviewed articles in several reputable international journals and conferences, and has four filed patents. He received the Distinguished Ph.D. Scholar Award in 2014 from the National Cheng Kung University, Tainan, Taiwan. He currently serves as an Associate Editor of the IEEE SENSORS JOURNAL, the SECURITY AND COMMUNICATION NETWORKS and the MOBILE INFORMATION SYSTEMS JOURNAL.