LDAKM-EIoT: Lightweight Device Authentication and Key Management Mechanism for Edge-Based IoT Deployment

Mohammad Wazid

Ashok Kumar Das

Sachin Shetty
Old Dominion University, sshetty@odu.edu

Joel J. P. C. Rodrigues

Youngho Park

Follow this and additional works at: https://digitalcommons.odu.edu/vmasc_pubs

Part of the Digital Communications and Networking Commons, Information Security Commons, and the Systems and Communications Commons

Original Publication Citation

This Article is brought to you for free and open access by the Virginia Modeling, Analysis & Simulation Center at ODU Digital Commons. It has been accepted for inclusion in VMASC Publications by an authorized administrator of ODU Digital Commons. For more information, please contact digitalcommons@odu.edu.
Article

LDAKM-EIoT: Lightweight Device Authentication and Key Management Mechanism for Edge-Based IoT Deployment

Mohammad Wazid 1, Ashok Kumar Das 2, Sachin Shetty 3, Joel J. P. C. Rodrigues 4,5, and Youngho Park 6, 6,∗, †

1 Department of Computer Science and Engineering, Graphic Era Deemed to be University, Dehradun 248 002, India; wazidkec2005@gmail.com
2 Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad 500 032, India; ashok.das@iiit.ac.in
3 Virginia Modeling, Analysis and Simulation Center, Center for Cybersecurity Education and Research, Department of Computational Modeling and Simulation Engineering, Old Dominion University, Suffolk, VA 23435, USA; sshetty@odu.edu
4 Federal University of Piauí (UFPI), 64049-550 Teresina-Pi, Brazil; joeljr@ieee.org
5 Instituto de Telecomunicações, 1049-001 Lisbon, Portugal
6 School of Electronics Engineering, Kyungpook National University, Daegu 41566, Korea
∗ Correspondence: parkyh@knu.ac.kr; Tel.: +82-53-950-5114
† Current address: School of Electronics Engineering, Kyungpook National University, 80 Daehak-ro, Sangyeok-dong, Buk-gu, Daegu 41566, Korea.

Received: 19 November 2019; Accepted: 11 December 2019; Published: 14 December 2019

Abstract: In recent years, edge computing has emerged as a new concept in the computing paradigm that empowers several future technologies, such as 5G, vehicle-to-vehicle communications, and the Internet of Things (IoT), by providing cloud computing facilities, as well as services to the end users. However, open communication among the entities in an edge based IoT environment makes it vulnerable to various potential attacks that are executed by an adversary. Device authentication is one of the prominent techniques in security that permits an IoT device to authenticate mutually with a cloud server with the help of an edge node. If authentication is successful, they establish a session key between them for secure communication. To achieve this goal, a novel device authentication and key management mechanism for the edge based IoT environment, called the lightweight authentication and key management scheme for the edge based IoT environment (LDAKM-EIoT), was designed. The detailed security analysis and formal security verification conducted by the widely used “Automated Validation of Internet Security Protocols and Applications (AVISPA)” tool prove that the proposed LDAKM-EIoT is secure against several attack vectors that exist in the infrastructure of the edge based IoT environment. The elaborated comparative analysis of the proposed LDAKM-EIoT and different closely related schemes provides evidence that LDAKM-EIoT is more secure with less communication and computation costs. Finally, the network performance parameters are calculated and analyzed using the NS2 simulation to demonstrate the practical facets of the proposed LDAKM-EIoT.

Keywords: Internet of Things (IoT); edge computing; authentication; key management; security; AVISPA; NS2 simulation

1. Introduction

The Internet of Things (IoT) is a network of physical objects (for example, smart devices) such as smart vehicles, smart industrial monitoring machines, smart home appliances, and many more. Such
objects are connected together to gather, process, rectify, and exchange relevant data over the Internet [1–6]. Furthermore, smart devices along with application programming interfaces (APIs) are used to connect and exchange data over the Internet. Each physical object (i.e., smart lighting system) has a provided IP address, which makes it capable to communicate (i.e., sending and receiving) over the network without human involvement. The shortage of IPv4 addresses resulted in designing the “IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN)”, which has significantly changed the IoT in order to increase the use of IPv6 by smart, as well as small scale objects [7]. The IoT communication environment is used in various types of applications, such as “smart health care”, “smart traffic monitoring”, and “smart homes”, to name a few. Edge computing introduced a new concept of computing. It has already become popular in industry, as well as academic research communities. It empowers many future technologies (for example, 5G, vehicle-to-vehicle and vehicle-to-cloud communications, augmented reality) by putting the connection in between end users and the cloud computing model and services [8–10]. Edge computing brings the utilities and services of cloud computing, which results in the faster processing and quicker response time of applications. Edge computing facilities the produced data of smart devices in the IoT environment to be processed nearby the location where it was generated in place of sending it across long routes to the “cloud” or “data centers” [10–12]. Though the edge based IoT environment provides many advantages over the traditional computing environment, at the same time, it has also the following security and privacy issues:

- The exchanged messages among different communicating parties (i.e., “IoT device”, “edge node”, and “cloud server”) should be protected against several known attacks (for example, “replay”, “man-in-the-middle”, “impersonation”, “offline or online password guessing”, and some other kinds of “bypassing attacks”).
- The edge node receives and processes the data, which are sent by the IoT devices. After the required processing, the edge node sends the data to the cloud server(s) for further processing and storage. Sometimes, the sent data are very critical and important, and any kind of disclosure of the data creates big trouble. Therefore, we need strong secure privacy preservation techniques to protect the data in the edge based IoT environment.
- As existing authentication protocols have security flaws that make them vulnerable to some known attacks (for example, “privileged insider attack”, “online and offline password guessing”, etc.), consequently, it becomes important to enhance the security of the authentication protocols, for instance in the case of “new device addition or revocation”, other communicating parties of the network should also be informed by the concerned (trusted) authority so that they can become aware about this activity and can update their memories accordingly [1,4]. Hence, it is an exigent task to provide a design for such a kind of protocol that supports “dynamic installation/update” without compromising the security of the system [4].
- In the edge based IoT environment, there is a possibility that some IoT devices may be “physically stolen/captured” by the adversary (A). After the physical capturing, A can use a “power analysis attack” [13] to obtain the data from the memory of the captured smart IoT devices. Later on, this drawn out information is used for other malicious works, such as deriving the “session key” among an IoT device and cloud server. A can replace the physical captured device with his/her own malicious device(s) that he/she has cloned in the laboratory. We should be more careful while going for the design of authentication and key management techniques in case some IoT devices are physically compromised so that there should not be any affect on the security of communication happening in the rest of the network [1,4].

As discussed earlier, the edge based IoT environment has several issues related to security and privacy. The existing authentication schemes have various “security flaws” that make them vulnerable to different known and unknown attacks. Some of them are not efficient from the computation and communication cost point of few. Hence, there is an essential requirement for a new “lightweight authentication and key management” scheme for the edge based IoT environment. Consequently, we design a new “lightweight authentication and key management” scheme for such an environment.
1.1. Contributions of LDAKM-EIoT

The contributions of this paper are many-fold in the following contexts:

- We propose a new “lightweight authentication and key management” scheme for edge based IoT environment (LDAKM-EIoT). In LDAKM-EIoT, we use various efficient operations such as bitwise “exclusive-OR (XOR)” and “one way collision resistant cryptographic hash functions”.
- LDAKM-EIoT is secure against different kinds of attacks by the help of “formal security verification” using the widely used AVISPA tool and also through other mathematical security analysis.
- The detailed comparative investigation among related existing schemes and LDAKM-EIoT evidences that LDAKM-EIoT achieves more security and additional functionality features and LDAKM-EIoT has also less communication and computation costs as compared to the other related schemes.
- The practical simulation study of LDAKM-EIoT is also executed with the help of the broadly used NS2 tool.

1.2. Paper Structure

The information about the “network model” and “threat model” of LDAKM-EIoT is discussed in Section 2. The brief discussion on the related existing authentication techniques is provided in Section 3. Different phases of LDAKM-EIoT are explained in Section 4. The security analysis of LDAKM-EIoT is conducted in Section 5. The formal security verification by the help of the AVISPA tool is done and explained in Section 6. The performance comparison among LDAKM-EIoT and other related existing authentication techniques is explained in Section 7. The impact of LDAKM-EIoT and other related schemes on network performance parameters is measured, analyzed, and then compared in Section 8 using the NS2 simulation. At last, the paper is concluded in Section 9.

2. Related Work

Wolf and Serpanos [14] explained different security issues of cyber-physical systems (CPS) and IoT systems. They discussed a security (safety) threat model for CPS and IoT systems. Ni et al. [15] explained the role of fog nodes in various IoT applications. After that, they examined several promising IoT applications as per these different roles.

Yeh et al. [16] demonstrated an “elliptic curve cryptography (ECC)” based user authentication mechanism for wireless sensor networks (WSNs). Their scheme did not achieve the mutual authentication property. To fix the problem of their scheme, Shi and Gong [17] proposed another improved “ECC based user authentication” scheme in WSNs. After that, Turkanovic et al. [18] came up with another user “authentication and key establishment” protocol for heterogeneous WSNs. Later on, their scheme was discovered to be insecure against “offline password guessing”, “offline identity guessing”, “smart card stolen”, “sensor node impersonation”, and “user impersonation” attacks. Moreover, their scheme did not support one of the essential properties, named as “mutual authentication” [19].

Khalil et al. [20] shed some light on the integration of WSNs in IoT. In the IoT environment, smart devices have limited computing and storage resources, like WSNs. It is also emphasized that some of the existing authentication techniques had serious security flaws as they were vulnerable to “impersonation”, “sensing node physical capture”, “replay”, “man-in-the-middle”, and “privileged insider” attacks [3,21].

Farash et al. [21] demonstrated a technique for “user authentication and key establishment” for heterogeneous WSNs, which can be applicable for IoT communication. Later on, Amin et al. [22] did cryptanalysis on the scheme of Farash et al. [21] and discovered that it was not secure against possible attacks such as “offline password guessing” by using the lost/stolen smart card, “session-specific temporary information leakage”, and “user impersonation” attacks. Furthermore, Amin et al. [22] proposed an improved version of their scheme to mitigate the security flaws of Farash et al.’s scheme [21]. Srinivas et al. [23] demonstrated that the scheme of Amin et al. [22] was
insecure against “user impersonation”, “leakage of different keys”, “stolen smart card”, and “server spoofing” attacks. Srinivas et al. further presented an improved and enhanced version of their scheme for “user authentication” in multi-gateway WSNs applicable to the IoT environment. Jiang et al. [24] also discovered the security flaws in the scheme of Amin et al. [22] as it was not able to mitigate some attacks. To overcome these vulnerabilities in the scheme of Amin et al., Jiang et al. presented an improved scheme.

Hsieh and Leu [25] proposed a new technique to resolve the security weaknesses of the other schemes [26–28]. Later on, Wu et al. [29] performed crypto-analysis of Hsieh–Leu’s scheme and identified the vulnerabilities of their scheme as it was not secure against different attacks such as “offline guessing”, “user forgery”, “insider attack”, and “sensor node physical capturing”. In addition, their scheme was not secure against the session key security, and also, it did not provide the mutual authentication property. To resolve these problems, Wu et al. presented a new “user authentication” protocol for WSN, which was also applicable to IoT communication.

Li et al. [30] presented a device-to-device authentication protocol in the IoT environment. Their approach relied on public key techniques that needed public key encryption and decryption by the resource limited IoT smart devices. Though their approach maintained security, it demanded high communication and computation overheads from the IoT devices’ point of view.

Santos et al. [31] designed a “federated identity management (FIdM)” system in order to assist in improving privacy and user authentication in the IoT deployment, where an IoT device accesses services from a service provider. Their approach was lightweight as it required low computational cost due to symmetric cryptographic operations. They applied their proposal to a cashless toll system environment.

Gope and Sikdar [32] designed a “lightweight and privacy preserving two factor authentication” scheme in which physical security was included. In their proposal, an IoT device and a server mutually authenticated each other for accessing services in the IoT environment. However, their scheme needed more computational cost due to fuzzy extractor generation and reproduction functions [33].

In order to design an access control and security policy, Han and Kim [34] designed a mutual authentication scheme between IoT devices. Though their scheme was lightweight due to symmetric cryptographic operations, it did not preserve device anonymity and untraceability properties.

Group key management (GKM) is considered as another important security aspect in the IoT environment, which helps with assigning IoT devices into predefined groups in the network. After that, key management is essential within each group and also among various groups in order to improve efficiency and security in the IoT environment. For this issue, Kung and Hsiao [35] designed an efficient GKM policy in an IoT deployment. Their approach also permitted joining and leaving the devices in a group dynamically in the IoT environment. Their approach was lightweight as it relied on the “cryptographic hash function” and symmetric cryptographic operations.

Raza and Magnusson [36] pointed out that “unanimous consensus” is extremely essential in the IoT environment for cyber security purposes. They presented a lightweight adaptation of the “Internet Key Exchange Version 2 (IKEv2)” for the IoT deployment, called TinyIKE. With the help of TinyIKE, they were able to solve the key management issue for several IoT protocols by applying a single IKEv2 based approach. TinyIKE relies on certificates, elliptic curve cryptographic techniques, as well as symmetric cryptographic techniques.

Challa et al. [3] recommended an ECC based user authentication protocol for IoT applications that can be used in the coming future. However, Jia et al. [37] highlighted that Challa et al.’s method did not protect against impersonation attack and also it did not preserve the untraceability property. Moreover, Challa et al.’s scheme [3] was expensive in computation and communication. Recently, Malani et al. [38] presented a “certificate based device access control” scheme applied to IoT communication to mitigate the security problems and limitations of the existing authentication and access control protocols. Their scheme the preserved anonymity property.

Sharma and Kalra [39] proposed a lightweight user authentication protocol that can be applied to healthcare services in the cloud-IoT environment. However, their scheme was insecure against
privileged-insider attack during the registration of medical professional, where the password of the medical professional was easily guessed by an attacker with the help of stolen smart card attack and registered information supplied by the medical professional to the gateway node. Moreover, their scheme did not provide the sensor node anonymity property and session key security under the Canetti and Krawczyk (CK) adversary model [40] (discussed in the threat model in Section 3.2). Zhou et al. [41] designed an authentication protocol that utilized IoT based architectures combined with cloud servers. Though their scheme was lightweight in nature, Martinez-Pelaez et al. [42] pointed out that Zhou et al.’s scheme [41] was vulnerable to various attacks, such as privileged-insider attack, man-in-the-middle attack, replay attack, and user impersonation attack. Furthermore, Martinez-Pelaez et al. also showed that Zhou et al.’s scheme failed to provide mutual authentication and secret key protection. To mitigate the security limitations mentioned in Zhou et al.’s scheme, Martinez-Pelaez et al. proposed an efficient authentication scheme.

As discussed above, most of the available methods for “authentication and key agreement” for communication in IoT and WSNs are insecure against different types of attacks. In addition, some of these techniques use heavy weight operations as they require more communication and computing resources. As a result, these existing techniques demonstrated for the “IoT environment” may not be much suitable for the authentication procedure in the edge based IoT environment. Consequently, we perceive that there is a requirement for a secure “authentication and key management” mechanism for the edge based IoT environment that should be lightweight. To fulfill this goal, we propose a new “lightweight authentication and key management” scheme for the edge based IoT environment that utilizes only an efficient “one way cryptographic hash” function and “bitwise XOR” operation.

3. System Models

The following two models are utilized to describe the working and usability of LDAKM-EIoT.

3.1. Network Model

The network model of LDAKM-EIoT is presented in Figure 1. In this figure, we have IoT devices, an edge node, which is a gateway to the Internet, the cloud server(s), and a trusted authority (TA). IoT devices can be deployed as per their applications (for example, smart health care, smart manufacturing, smart cities, and smart homes). The task of each IoT device is to collect and process information about some activity (i.e., level of fumes in a industrial plant) and then to send the data to the cloud server(s) through the edge node. At the cloud server, this will be stored and utilized for other processing and decision making tasks. In this model, IoT devices are resource constrained, whereas the edge node and cloud server are resource rich as they are able to do intensive computations and have more storage capacity and battery backup. The TA is responsible for the registration of different devices (i.e., IoT devices, edge nodes, and cloud servers). In such a kind of communication environment, we have to secure the communication between the IoT device and edge node and the edge node and cloud server. Since most of IoT devices are resource constrained, we preferred to use lightweight cryptographic operations (i.e., hash and XOR operations) in different exchanged messages’ computation. Consequently, to secure such kinds of communication, we urge a new “authentication and key management” protocol with lightweight operations. Using this scheme, the communicating entities can securely access the resources of other entities and also communicate securely.
3.2. Threat Model

The widely accepted “Dolev–Yao (DY) threat model” [43] was followed in the designing of LDAKM-EIoT. As per the guidelines of the DY model, two communicating entities communicate over an open (insecure) channel. The end-point entities such as “IoT devices” are not fully trusted in general. The existing network adversary (A) can eavesdrop, update, or delete the communicated messages as the channel is insecure. We also followed Canetti and Krawczyk’s adversary model, also named as the “CK adversary model” [40], which is the current de facto standard model in the designing and modeling of an “authenticated key agreement” security scheme. In the CK adversary model, A can have all the capabilities like the DY model. In addition to that, A can also compromise the secret credentials along with the “session states and session keys” during a session. Moreover, A can physical capture some IoT devices and extract the stored information from the IoT devices by the application of a sophisticated power analysis attack [13]. The obtained credentials can be further used to perform other kinds of malicious activities, such as the computation of “secret credentials” and the “session key”, and launch some attacks, such as “IoT device impersonation”, “replay”, “privileged-insider”, and “man-in-the-middle”. As per the information available in [44], we also assumed that the edge node (EN) was fully trusted and could not be compromised by A. Otherwise, if the EN was compromised, the entire network would be compromised. For such a purpose, we followed the method discussed by Bertino et al.’s protocol [45]. We assumed that the EN was equipped with a tamper resistant hardware device so that all the sensitive confidential information (for example, stored cryptographic keys) should be protected from A. Therefore, the application of a tamper resistant EN attained strong enough security in LDAKM-EIoT. Though it is also true that the attacks are possible on tamper resistant devices, A needs special equipment (device) to perform such an attack to acquire the stored information. This is because it is less expensive to install the EN than to use special equipment because A does not have any economic benefits to perform such an attack on a security scheme [45]. Moreover, we could secure the EN by putting it inside a physical locking system so that the physical capture of the EN will be difficult for A as compared to that for the unattended IoT smart devices. In addition, the “trusted authority (TA)” was also assumed to be a “fully trusted” entity of the network and that it would not be compromised, although the “cloud servers” were considered to be “semi-trusted” entities.
4. The Proposed Scheme: LDAKM-EIoT

In this section, we talk about the precise workings of the proposed scheme, called the “lightweight authenticated key management mechanism for the edge-based IoT environment (LDAKM-EIoT)”. The distinct phases of LDAKM-EIoT are provided in the upcoming part of the paper. It was also assumed that the different network entities (i.e., IoT device, edge node, cloud server) were synchronized with their clocks. It is mandatory to have this assumption while we go for the designing of an authentication mechanism for different networks [1–3,46–48]. We needed different notations for the designing of LDAKM-EIoT, which we summarize in Table 1 along with their significance.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_i$, $ID_{D_i}$</td>
<td>$i$th IoT device and its identity, respectively</td>
</tr>
<tr>
<td>$CS_j$, $ID_{CS_j}$</td>
<td>$j$th cloud server and its identity, respectively</td>
</tr>
<tr>
<td>$EN$, $ID_{EN}$</td>
<td>Edge node and its identity, respectively</td>
</tr>
<tr>
<td>$TA$, $ID_{TA}$</td>
<td>Trusted authority and its identity, respectively</td>
</tr>
<tr>
<td>$RID_{D_i}$, $RID_{CS_j}$</td>
<td>Pseudo identities of $D_i$ and $CS_j$, respectively</td>
</tr>
<tr>
<td>$TID_{D_i}$</td>
<td>Temporary identity of $D_i$</td>
</tr>
<tr>
<td>$K_{EN-D_i}$</td>
<td>1024 bit shared secret key of the IoT device and edge node generated by $TA$</td>
</tr>
<tr>
<td>$K_{EN-CS_j}$</td>
<td>1024 bit shared secret key of the edge node and cloud server generated by $TA$</td>
</tr>
<tr>
<td>$n$</td>
<td>160 bit secret key of $TA$, which is only known to $TA$.</td>
</tr>
<tr>
<td>$RTS_{D_i}$</td>
<td>Registration timestamp of $D_i$</td>
</tr>
<tr>
<td>$n_d, n_c$</td>
<td>Number of IoT devices and cloud servers deployed initially, respectively</td>
</tr>
<tr>
<td>$r_{D_i}, r_{CS_j}, r_{TA}, r_{EN}$</td>
<td>128 bit random numbers of $D_i$, $EN$, and $CS_j$, respectively</td>
</tr>
<tr>
<td>$T_1, T_2, T_3, T_4$</td>
<td>Current timestamps generated by different entities</td>
</tr>
<tr>
<td>$\Delta T$</td>
<td>“Maximum transmission delay” associated with a message</td>
</tr>
<tr>
<td>$h(\cdot)$</td>
<td>“Collision resistant cryptographic one way hash function”</td>
</tr>
<tr>
<td>$SK_{D_i-CS_j}$</td>
<td>Session key between $D_i$ and $CS_j$</td>
</tr>
<tr>
<td>$|$, $\oplus$</td>
<td>Concatenation and bitwise XOR operations, respectively</td>
</tr>
</tbody>
</table>

4.1. Pre-Deployment Phase

This phase permitted the trusted authority (TA), a fully trusted entity in the network, to perform the registration of IoT devices, edge nodes, and cloud servers before they were installed in the deployment area. We assumed that $n_d$ number of IoT devices $D_i$ were associated with a particular edge node $EN$, and the real-time information could be accessed from the IoT devices to $n_c$ number of cloud servers $CS_j$, provided that a mutual authentication and key establishment happened successfully among the IoT devices and cloud servers with the presence of the edge node $EN$.

4.1.1. Registration of IoT Devices

The TA firsts picked a distinct unique identity $ID_{D_i}$ for every IoT device $D_i$ and computed its corresponding pseudo identity as $RID_{D_i} = h(n||ID_{D_i})$, temporal credentials $TC_{D_i} = h(RID_{D_i}||ID_{EN}||ID_{TA}||n||RTS_{D_i})$, $AD_i = h(K_{EN-D_i}||ID_{D_i}||ID_{EN}||RTS_{D_i})$ where $RTS_{D_i}$ is the registration timestamp for $D_i$, $K_{EN-D_i}$ is a 1024 bit shared secret key between each IoT device $D_i$ and edge nodes generated by the $TA$, where $i = 1, 2, \ldots, n_d$, and $n_d$ is the total count of IoT devices. Moreover, $ID_{EN}$ is the identity of an edge node $EN$ where the IoT devices need to authenticate a cloud server $CS_j$ with an IoT device $D_i$. $ID_{TA}$ is the identity of $TA$, and $n$ is a 160 bit secret number of $TA$ that is only disclosed to the $TA$. After that, $TA$ generates a temporary identity $TID_{D_i}$ for each $D_i$. Note that temporal credentials are different for each IoT device $D_i$, which protects against various attacks including impersonation attack, in case any IoT device is physically compromised by an attacker. Lastly, the trusted authority stores the credentials $\{RID_{D_i}, TID_{D_i}, TC_{D_i}, AD_i, h(\cdot)\}$ in the memory of $D_i$ prior to its placement in the IoT network.
4.1.2. Registration of Cloud Servers

The TA picks a distinct unique identity $ID_{CS_j}$ for each cloud server $CS_j$ and proceeds to calculate its pseudo identity as $RID_{CS_j} = h(n || ID_{CS_j})$ and $ACS_j = h(K_{EN-CS_j} || ID_{CS_j} || ID_{EN} || RTS_{CS_j})$ where $RTS_{CS_j}$ is the registration timestamp for $CS_j$ and $j = 1, 2, \cdots n_c$ are the number of cloud servers placed initially in the IoT network, while $K_{EN-CS_j}$ is a 1024 bit shared secret key of the edge node and cloud server $CS_j$ generated by the TA. Next, the TA stores the credentials $\{RID_{CS_j}, ACS_j, ID_{CS_j}, h(\cdot)\}$ in the database of $CS_j$ before its placement in the network.

4.1.3. Registration of Edge Node

The TA picks a distinct unique identity $ID_{EN}$ for edge node $EN$ and stores previously computed information, as well as this information, that is $\{(TID_{D_i}, RID_{D_i}, ID_{D_i}, TC_{D_i}, AD_i) | 1 \leq i \leq n_d\}$, $\{(ID_{CS_j}, RID_{CS_j}, ACS_j) | 1 \leq j \leq n_c\}, h(\cdot))$, in the database of $EN$ before its placement in the network. The registration process is summarized in Figure 2.

<table>
<thead>
<tr>
<th>IoT device ($D_i$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>${RID_{D_i}, TID_{D_i}, TC_{D_i}, AD_i, h(\cdot)}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cloud server ($CS_j$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>${RID_{CS_j}, ACS_j, ID_{CS_j}, h(\cdot)}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Edge node ($EN$)</th>
</tr>
</thead>
</table>
| $\langle\{(TID_{D_i}, RID_{D_i}, ID_{D_i}, TC_{D_i}, AD_i) | 1 \leq i \leq n_d\}$, $
\{(ID_{CS_j}, RID_{CS_j}, ACS_j) | 1 \leq j \leq n_c\}, h(\cdot)\rangle$ |

**Figure 2.** Credentials stored in $D_i$, $CS_j$, and $EN$ during registration processes.

4.2. Authentication and Key Agreement Phase

The key management procedure helps to secure the authentication and key management among IoT devices and cloud servers by the involvement of a trusted edge node. The upcoming steps are essential to achieve this goal.

**A1.** When an IoT device, say $D_i$, wants to initiate secure data transmission to a cloud server, first of all, $D_i$ needs to compute the following parameters. $D_i$ picks a random nonce $r_d$, and current timestamp $T_1$ and calculates $M_1 = h(AD_i || T_1) \oplus r_d$ and $M_2 = h(AD_i || T_1 || TID_{D_i} || RID_{D_i} || TC_{D_i} || r_d)$. Next, $D_i$ sends the message $Msg_1 = \{TID_{D_i}, M_1, M_2, T_1\}$ to the edge node $EN$ via an open channel.

**A2.** After the arrival of message $Msg_1$ at time $T_1^*$, the $EN$ checks the timeliness of $T_1$ with the verifying condition $|T_1 - T_1^*| \leq \Delta T$. If it is valid, the $EN$ fetches $RID_{D_i}$, $TC_{D_i}$, $AD_i$, and $ID_{D_i}$ corresponding to received $TID_{D_i}$, and computes $r_d = M_1 \oplus h(AD_i || T_1)$, $M_2' = h(AD_i || T_1 || TID_{D_i} || RID_{D_i} || TC_{D_i} || r_d)$ and checks whether $M_2' = M_2$. If it is valid, $D_i$ is authenticated by the $EN$ and can access its resources to get access to a cloud server $CS_j$ selected by the $EN$. Next, the $EN$ chooses a random nonce $r_{en}$ and current timestamp $T_2$, picks $ACS_j$ from its database corresponding to $ID_{CS_j}$, and computes $TID_{D_i} = TID_{D_i} \oplus h(ACS_j || T_2)$, $M_3 = h(ACS_j || RID_{CS_j}) \oplus h(r_{en} || T_1 || AD_i || ID_{D_i} || r_d || TC_{D_i})$ and $M_4 = h(TID_{D_i} || T_2 || RID_{CS_j} || ACS_j || h(r_{en} || T_1 || AD_i || ID_{D_i} || r_d || TC_{D_i}))$. After that, the $EN$ sends the message $Msg_2 = \{TID_{D_i}, M_3, M_4, T_2\}$ to $CS_j$ via an open channel.
A3. After the arrival of message \( \text{Msg}_2 \) at time \( T_2 \), \( C_S \) verifies the timeliness of \( T_2 \) by checking if \( |T_2 - T_3| \leq \Delta T \) is valid. After successful verification of \( T_2 \), \( C_S \) computes \( TID_{D_1} = TID_{D_1} \oplus h(ACS_j || T_2) \oplus h(r_{en_1} || T_1 || AD_i || ID_{D_1} || r_{d_1} || TC_{D_1}) = M_3 \oplus h(ACS_j || RID_{CS_j}) \) and \( M'_i = h(TID_{D_1} || T_2 || RID_{CS_j} || ACS_j || h(r_{en_1} || T_1 || AD_i || ID_{D_1} || r_{d_1} || TC_{D_1})). \) After that, \( C_S \) checks whether \( M'_i = M_4 \) is satisfied. If it is valid, the \( E_N \) is authenticated by \( C_S \). Otherwise, \( C_S \) halts the session with the \( E_N \) immediately. Furthermore, \( C_S \) picks up a random nonce \( r_{en_2} \) and current timestamp \( T_3 \) and computes \( M_5 = r_{en_2} \oplus h(T_2 || ACS_j || RID_{CS_j}). \) Then, \( C_S \) computes the session key shared with \( D_i \) as \( SK_{S_i-D_i} = h(TID_{D_1} || T_3) \oplus h(r_{en_1} || T_1 || AD_i || ID_{D_1} || r_{d_1} || TC_{D_1}). \) After computing these values, \( C_S \) sends the message \( \text{Msg}_3 = \{M_5, M_6, M_7, T_3\} \) to the \( E_N \) via an open channel.

A4. After arrival of message \( \text{Msg}_3 \) at time \( T_3 \), the \( E_N \) verifies the timeliness of \( T_3 \) by checking if \( |T_3 - T_3| \leq \Delta T \) is valid. Upon successful validation of \( T_3 \), the \( E_N \) calculates \( r_{en_3} = M_5 \oplus h(T_2 || ACS_j || RID_{CS_j}) \) and \( M'_i = h(r_{en_3} || T_2 || T_3 || ACS_j || RID_{CS_j}). \) Next, the \( E_N \) checks whether \( M'_i = M_7 \), and if it holds, \( C_S \) is authenticated by the \( E_N \). Otherwise, the \( E_N \) halts the session with \( C_S \) instantly. Furthermore, the \( E_N \) selects a random nonce \( r_{en_2} \) along with current timestamp \( T_4 \) and computes \( M_8 = h(AD_i || TC_{D_1}) \oplus h(r_{en_2} || T_4), a = h(RID_{D_1} || TC_{D_1} || T_4) \oplus h(r_{en_1} \oplus T_1 || AD_i || ID_{D_1} || r_{d_1} || TC_{D_1}), \) \( \beta = h(r_{en_3} || ACS_j || ID_{CS_j} || T_3) \oplus h(r_{en_3} || T_4)), \) \( |AD_i||TC_{D_1} \) and \( M_9 = M_3 \oplus h(r_{en_3} || T_4 || T_3 || T_4). \) After computing these values, the \( E_N \) picks a new temporary identity \( TID_{D_1}^{new} \) for \( D_i \) and calculates \( M_{10} = h(M_9 || T_3 || T_4 || AD_i || TC_{D_1} || TID_{D_1}^{new} || TID_{D_1}). \) The \( E_N \) also replaces \( TID_{D_1} \) with \( TID_{D_1}^{new} \) in its database. Finally, the \( E_N \) sends the message \( \text{Msg}_4 = \{M_8, a, \beta, M_9, M_{10}, T_3, T_4\} \) to \( E_N \) via an open channel.

A5. After arrival of message \( \text{Msg}_4 \) at time \( T_4 \), \( D_i \) verifies the timeliness of \( T_4 \) by checking if \( |T_4 - T_4| \leq \Delta T \) is valid. After successful validation of \( T_4 \), \( D_i \) calculates \( h(r_{en_2} || T_4) = M_8 \oplus h(AD_i || TC_{D_1}), h(r_{en_1} || T_1 || AD_i || ID_{D_1} || r_{d_1} || TC_{D_1}) = a \oplus h(RID_{D_1} || TC_{D_1} || T_4), h(r_{en_2} || ACS_j || ID_{CS_j} || T_3) = \beta \oplus h(a || AD_i || TC_{D_1}), \) the shared session key with \( C_S \) as \( SK_{D_i-C_S} = h(TID_{D_1} || T_3) || h(r_{en_1} || T_1 || AD_i || ID_{D_1} || r_{d_1} || TC_{D_1}) || h(r_{en_2} || ACS_j || ID_{CS_j} || T_3). \) In addition, \( D_i \) also calculates \( M'_i = h(SK_{D_i-C_S} || T_3) \) and \( M'_i = h(M'_i \oplus M_8 || h(r_{en_2} || T_4 || T_3 || T_4)). \) Then, \( D_i \) checks whether \( M'_i = M_9 \), and if it is legitimate, the \( E_N \), as well as \( C_S \), are authenticated by \( D_i \), and the computed session key is treated as the valid one. Thus, both \( D_i \) and \( C_S \) will maintain the same computed shared session key \( SK_{D_i-C_S} = SK_{D_i-C_S} \) for secret communications. Moreover, \( D_i \) calculates the new temporary identity as \( TID_{D_i}^{new} = M_{10} \oplus h(M_9 || T_3 || T_4 || AD_i || TC_{D_1} || TID_{D_1}) \) and will use this new identity in all its future communications with the \( E_N \) and \( C_S \). It is worth noticing that the new temporary identity generation mechanism makes the proposed scheme prevent the traceability attack.

Finally, the above phase is abridged in Figure 3.

**Remark 1** (Protection for synchronization attack). We may consider a situation where the message \( \text{Msg}_4 \) has been tampered with or there is a communication error that occurs so that an IoT device cannot receive the parameter \( M_{10} \), as well as the new temporary identity \( TID_{D_i}^{new} \). In order to resolve this problem, we contemplated a possible solution suggested in [49], where an IoT device needs to maintain a set of \( l \) shadow identities \( SID = \{s_{id_1}, s_{id_2}, \ldots, s_{id_1}\} \). In this context, when an IoT device, say \( D_i \), could not receive the message \( \text{Msg}_4 \) within a specific time period (maximum round-trip time), it needs to select one of the unused shadow identities, say \( s_{id_l} \in SID \), and then send it within the message \( \text{Msg}_3 \). When the \( E_N \) receives the \( s_{id_l} \), it can generate a new \( TID_{D_i} \) and send it securely to the device \( D_i \) by using this method, we can address the synchronization attacks without compromising the privacy of an IoT device in the proposed scheme (LDAKM-EIoT).
4.3. Dynamic Node Addition Phase

There is the chance that some of the IoT devices fail to work properly or might stop their working completely. Furthermore, there is the possibility that some of the IoT devices can be “physically stolen (capture)” by an adversary A. In order to draw out the useful data from the stolen IoT devices (i.e., session key, temporary and pseudo identities, and other credentials stored in their memory), henceforth, it becomes very important to add new IoT devices in the required area. LDAKM-EIoT provides a facility for the addition of new IoT devices and also cloud servers in the network at any time after initial installment.

4.3.1. Dynamic IoT Device Addition

The upcoming procedure is needed to add a new IoT device, say \( D_{ni} \), in the required part of the IoT environment.

**DD1.** The TA picks a new unique identity \( ID_{D_{ni}} \) for the new IoT device \( D_{ni} \) and computes its corresponding pseudo identity as \( RID_{D_{ni}} = h(n || ID_{D_{ni}}) \), temporal credentials \( TC_{D_{ni}} = h(RID_{D_{ni}} || ID_{EN} || ID_{TA} || h(RTS_{D_{ni}}), AD_{ni} = h(K_{EN-D_{ni}} || ID_{D_{ni}} || ID_{EN} || RTS_{D_{ni}}) \) where RTS_{D_{ni}} is registration credentials.
timestamp generated for \(D_{ni}\) and \(K_{EN-D_{ni}}\) is the shared secret key between IoT device \(D_{ni}\) and the edge node \(EN\) generated by the TA. TA also generates a unique temporary identity \(TID_{D_{ni}}\) for \(D_{ni}\) that is different from the previously deployed IoT devices.

**DD2.** The TA stores the credentials \(\{RID_{D_{ni}}, TID_{D_{ni}}, TC_{D_{ni}}, AD_{ni}, h(\cdot)\}\) in the memory of \(D_{ni}\) before its placement in the deployment area. Furthermore, the TA securely sends the information \(\{RID_{D_{ni}}, TID_{D_{ni}}, TC_{D_{ni}}, AD_{ni}\}\) related to \(D_{ni}\) to \(EN\) so that the \(EN\) can update this information in its database corresponding to the newly added IoT device \(D_{ni}\). Note that for secure communication between TA and EN, a pre-shared symmetric key among them is essential, and it is generated by the TA prior to the placement of the EN.

### 4.3.2. Dynamic Cloud Server Addition

The upcoming procedure is required to add a new cloud server, say \(CS_{nj}\), in the required part of the IoT environment.

**DC1.** The TA chooses a distinct unique identity \(ID_{CS_{nj}}\) for \(CS_{nj}\) and computes its pseudo identity as \(RID_{CS_{nj}} = h(n \mid ID_{CS_{nj}})\) and \(ACS_{nj} = h(K_{EN-CS_{nj}} \mid ID_{CS_{nj}} \mid ID_{EN} \mid RTS_{CS_{nj}})\) where \(RTS_{CS_{nj}}\) is the registration timestamp for \(CS_{nj}\). Moreover, \(K_{EN-CS_{nj}}\) is the shared secret key of the edge node \(EN\) and cloud server \(CS_{nj}\) generated by the TA.

**DC2.** The TA then stores the credentials \(\{RID_{CS_{nj}}, ACS_{nj}, ID_{CS_{nj}}, h(\cdot)\}\) into the database of \(CS_{nj}\) prior to its deployment in the IoT environment.

## 5. Security Analysis of LDAKM-EIoT

In this section, we provide the security analysis of LDAKM-EIoT in Propositions 1–7, which prove its robustness against the following possible attacks.

**Proposition 1.** LDAKM-EIoT is resilient against replay attack.

**Proof.** In LDAKM-EIoT, we utilized various current timestamp values for the communicating entities \((D_{i}, \text{EN}, \text{and} \ CS)\). In the each exchanged message of LDAKM-EIoT, we used the maximum transmission delay \(\Delta T\) factor, which is a small value. Consequently, an adversary \(A\) can not gain any advantage in replaying the old transmitted messages, which are used in the authentication and key management procedure among \(D_{i}\), \(EN\), and \(CS_{j}\) within \(\Delta T\). Thus, LDAKM-EIoT is secure against replay attack.

**Proposition 2.** LDAKM-EIoT is resilient against man-in-the-middle attack (MITM).

**Proof.** Suppose an adversary \(A\) seizes an authentication request message \(\{TID_{D_{j}}, M_{1}, M_{2}, T_{1}\}\) exchanged between \(D_{i}\) and \(EN\) and further tries to update this message in such a way that it looks similar to a genuine authentication message, as \(\{TID'_{D_{j}}, M'_{1}, M'_{2}, T'_{1}\}\) by using the parameters such as \(M'_{1} = AD_{i} \oplus r_{d_{i}}\) and \(M'_{2} = h(AD_{i} \mid \mid T'_{1} \mid \mid TID'_{D_{j}} \mid \mid RID_{D_{j}} \mid \mid TC_{D_{j}} \mid \mid \mid r_{d_{j}})\). The values of the used variables are \(RID_{D_{j}} = h(n \mid ID_{D_{j}})\), \(TC_{D_{j}} = h(\text{ID}\_\text{EN} \mid ID_{TA} \mid \mid n \mid RTS_{D_{j}})\), \(AD_{i} = h(K_{EN-D_{i}} \mid ID_{D_{j}} \mid ID_{EN} \mid RTS_{D_{j}})\) where \(RTS_{D_{j}}\) is the registration timestamp for \(D_{i}\) and \(K_{EN-D_{i}}\) is the shared secret key of IoT device and edge node generated by the TA. To launch this attack, \(A\) can initiate the generation of random nonce \(r_{d_{j}}\) and current timestamp \(T_{1}^{a}\). However, without obtaining the information about "long term secret" \(RID_{D_{j}}, ID_{EN}, ID_{TA}, n,\) and \(K_{EN-D_{i}}\), \(A\) cannot re-create another valid authentication request message \(\{TID'_{D_{j}}, M'_{1}, M'_{2}, T'_{1}\}\). By using a similar approach, we can also explain that other messages cannot be recomputed by \(A\), which are used in the "authentication and key management" phase without the "long term secrets" used by \(D_{i}, EN, \text{and} CS_{j}\). This proves the resilience of LDAKM-EIoT against MITM.

**Proposition 3.** LDAKM-EIoT provides protection against different impersonation attacks.
Proof. Suppose an adversary $A$ attempts to create a valid authentication request message on behalf of a communicating party (for example, $D_i$) after obtaining $D_i$'s authentication request $MSG_3 = \{TID_{D_i}, M_1, M_2, T_1\}$ sent towards the $EN$ by $D_i$, where $M_1 = AD_i \oplus r_d$, and $M_2 = h(AD_i || T_1 || TID_{D_i} || RID_{D_i} || TC_{D_i} || r_d)$. The values of the used variables are $RID_{D_i} = h(n||ID_{D_i})$, $TC_{D_i} = h(RID_{D_i} || ID_{EN} || ID_{TA} || n || RTS_{D_i})$, and $AD_i = h(K_{EN-D_i} || ID_{D_i} || ID_{EN} || RTS_{D_i})$ where $RTS_{D_i}$ is the registration timestamp for $D_i$ and $K_{EN-D_i}$ is the shared secret key of IoT device and edge node generated by the $TA$. Note that $MSG_1$ uses $M_1$ and $M_2$, which are computed using the “long term secrets”($RID_{D_i}, ID_{EN}, ID_{TA}, n, K_{EN-D_i}$) and also the “short term secrets” (random nonce $r_d$). Without knowing these secret values, $A$ is not able to create a valid authentication request message on behalf of the genuine IoT device $D_i$. Hence, we can say that LDAKM-EIoT is resilient against IoT device impersonation attack. In the same way, it can also be proven that LDAKM-EIoT provides protection against edge node and cloud server impersonation attacks as the creation of other messages $MSG_2$, $MSG_3$, and $MSG_4$ employs both “long term” and “short term” secrets. Therefore, LDAKM-EIoT is secure against impersonation attacks. \(\square\)

Proposition 4. LDAKM-EIoT protects Ephemeral secret leakage (ESL) attack.

Proof. In LDAKM-EIoT, the session key is computed by a genuine cloud server $CS_j$ and the accessed $D_i$ during the the authentication and key management phase as $SK_{CS_j-D_i} = h(TID_{D_i} || T_3 || h(r_{en_1} || T_1 || AD_i || ID_{D_i} || r_d || TC_{D_i}) || h(r_{CS_j} || ACS_j || ID_{CS_j} || T_3))$. Here, $AD_i = h(K_{EN-D_i} || ID_{D_i} || ID_{EN} || RTS_{D_i})$ where $K_{EN-D_i}$ is the shared secret key of the IoT device and edge node generated by the $TA$ and $K_{EN-CS_j}$ is the shared secret key of the $EN$ and cloud server generated by the $TA$. Further, the identities of the IoT device, $ID_{D_i}$, the identity of the edge node $ID_{EN}$, and the identity of the cloud server $ID_{CS_j}$ are also utilized. It is understandable that the “session key” is the combination of both the session temporary (ephemeral) information, also called “short term secrets” (different random nonces and timestamps), as well as the “long term secrets” (different secret keys and identities). Thus, the session key can only be disclosed if $A$ compromises both the session temporary and long term secrets. Moreover, as different random nonces and timestamps are used in the computation of the session keys between $D_i$ and $CS_j$ in distinct sessions, even if a session key is disclosed for a specific session, it will not result in computing the session keys of other sessions because of the combination of short and long term secrets. Thus, LDAKM-EIoT provides protection against session temporary information attack, and it also preserves the “perfect forward secrecy” goal. Therefore, LDAKM-EIoT is resilient against “ESL attack”. Consequently, LDAKM-EIoT preserves the session key security under the “CK adversary model” [40]. \(\square\)

Proposition 5. LDAKM-EIoT is resilient against privileged-insider attack.

Proof. The privileged-insider user of a cloud server, say $A$, knows the registration information $\{RID_{CS_j}, ACS_j, ID_{CS_j}, h(\cdot)\}$. However, he/she cannot calculate the “session key” on behalf of the cloud server as the session key utilizes the credentials that are only known to the IoT devices or the edge node. The “session key” calculated by the legitimate cloud server is $SK_{CS_j-D_i} = h(TID_{D_i} || T_3 || h(r_{en_1} || T_1 || AD_i || ID_{D_i} || r_d || TC_{D_i}) || h(r_{CS_j} || ACS_j || ID_{CS_j} || T_3))$. Moreover, $r_{en_1}$ is the random nonce of the $EN$. The identities of the IoT device, $ID_{D_i}$, the identity of the edge node $ID_{EN}$, and identity of the cloud server $ID_{CS_j}$ are also utilized in $SK_{CS_j-D_i}$. The privileged-insider user of the cloud server does not have any information about $ID_{D_i}$, $ID_{EN}$, and $K_{EN-D_i}$. Therefore, $A$ cannot calculate the “session key” on behalf of the cloud server, as well as he/she cannot impersonate the cloud server, as explained in Proposition 3. Hence, LDAKM-EIoT is resilient against “privileged-insider attack”. \(\square\)

Proposition 6. LDAKM-EIoT preserves the anonymity and untraceability properties.
Proof. Let us assume an adversary $A$ intercepts the messages $\text{Msg}_1 = \{\text{TID}_{D_i}, M_1, M_2, T_i\}$, $\text{Msg}_2 = \{\text{TID}_{D_j}, M_3, M_4, T_2\}$, $\text{Msg}_3 = \{M_5, M_6, M_7, T_3\}$, and $\text{Msg}_4 = \{M_8, a, \beta, M_9, M_{10}, T_3, T_4\}$ during the “authentication and key management phase” between $D_i$ and $CS_j$ via $EN$. We used random nonces and current timestamps in the construction of messages, which helped to generate dynamic and unique messages for different sessions. Moreover, the identities $ID_{D_i}$, $ID_{EN}$, and $ID_{CS_j}$ are not transmitted in the plaintext format during transmission. This helps to attain both the anonymity and untraceability objectives of LDAKM-EIoT.

Proposition 7. LDAKM-EIoT provides protection against IoT device physical capture attack.

Proof. Each IoT device stores the credentials $\{\text{RID}_{D_i}, \text{TID}_{D_i}, \text{TC}_{D_i}, \text{AD}_{i}, h(\cdot)\}$, which are used for the “authentication and key management” purposes with other network communicating entities. The protection against IoT device physical capture attack is a very essential security requirement $[50,51]$. Suppose $n_c$ IoT devices are physically captured by an adversary $A$. We assess the IoT device physical capture attack as the fraction of total secure communications that are compromised from the capturing of $n_c$ IoT devices not including the communication in which the “compromised IoT devices” are straightforwardly extended. For example, one can estimate the probability of $A$’s ability to decrypt the secure communication between a cloud server $CS_j$ and a non-compromised IoT device $D_i$ when $n_c$ IoT devices are already physically stolen and compromised. Suppose this probability is represented by $P_i(n_c)$. If $P_i(n_c) = 0$, an “authentication and key management” protocol is termed as “unconditionally secure against IoT device physical capture attack”. From a captured IoT device $D_i$, $A$ will have the extracted credentials $\{\text{RID}_{D_i}, \text{TID}_{D_i}, \text{TC}_{D_i}, \text{AD}_{i}, h(\cdot)\}$ along with the secret pairwise session key $SK_{CS_j-D_i}$ shared between $D_i$ and $CS_j$ from its memory. However, it is very important to notice that all $\text{RID}_{D_i}$, $\text{TID}_{D_i}$, $\text{TC}_{D_i}$, $\text{AD}_{i}$, $RTS_{D_i}$, and $K_{EN-D_i}$ are distinct for all installed IoT devices. Thus, the stolen $D_i$ by $A$ can only help to find the secret keys between that $D_i$ and $CS_j$. Furthermore, this results in all other secret keys between that $CS_j$ and other non-compromised IoT devices $D_i$ to be not still revealed. Therefore, compromising an IoT device will not cause the compromise of the entire communication, and we get the secure communications among the cloud server and other non-compromised IoT devices. This means that LDAKM-EIoT is unconditionally secure against IoT device physical capture attack.

6. Formal Security Verification Using AVISPA

This section shows the resilience of the proposed scheme (LDAKM-EIoT) against replay and man-in-the-middle attacks based on the formal security verification. We applied the broadly-accepted “Automated Validation of Internet Security Protocols and Applications (AVISPA)” tool $[52]$ to achieve this goal. There are four backends, namely “on-the-fly mode-checker (OFMC)”, “constraint-logic based attack searcher (CL-AtSe)”, “SAT (Boolean satisfiability problem) based model checker (SATMC)”, and “tree automata based on automatic approximations for the analysis of security protocols (TA4SP)”, which are integrated with the AVISPA tool. These backends help in automatic execution analysis of the security protocols. The “High-Level Protocol Specification Language (HLPSL)” helps in converting the high level implementation of the the security protocols into the “Intermediate Format (IF)” using the HLPSL2IF translator. HLPSL defines the roles for all the involved participating entities in a protocol, which are termed as the basic roles. The compulsory roles, known as “session” and “goal and environment”, consist of the composition of the sessions along with globally defined constants, respectively. The IF is provided as input to one of the four backends to produce the “Output Format (OF)”, which explicitly explains under what conditions the protocol becomes “safe”, “unsafe”, or “inconclusive”. The precise discussion on AVISPA and its HLPSL is available in $[52]$. The AVISPA tool is one of the widely used formal security verification tools. Researchers working in the authentication domain frequently use this software tool to validate the security of authentication protocols $[1–3,46,48,53]$. 

Sensors 2019, 19, 5539
In our implementation of the proposed scheme (LDAKM-EIoT) under HLPSL, we defined four basic roles: (a) the role for the TA, (b) the role for an IoT device \( D_i \), (c) the role for the edge node \( EN \), and (d) the role for a cloud server \( CS_j \). Apart from these basic roles, we had two compulsory roles (“session” and “goal and environment”). We applied the “SPAN (Security Protocol ANimator for AVISPA)” [54] for simulating LDAKM-EIoT. The backends that we covered were OFMC and CL-AtSe, because they support the bitwise XOR operation. Therefore, we excluded the simulation results under other backends (i.e., SATMC and TA4SP) from the simulation part, because they did not currently support XOR operation, and the results would be under these backends “inconclusive”. Figure 4 shows the simulation results under the OFMC and CL-AtSe backends. The summary of these backends helped us to predict the security of a designed scheme against replay and man-in-the-middle attacks. Under the OFMC backend, the simulation took 1019 milliseconds, while traversing 129 visited nodes at a depth of 20 plies. On the other hand, under the CL-Atse backend, 256 states were analyzed, and out of those states, 124 states were reachable with 0.58 s of translation time and 0.05 s of computation. The provided simulation results assured that LDAKM-EIoT was safe against the replay and man-in-the-middle attacks.

7. Comparative Analysis with Existing Schemes

In this section, the comparative study of LDAKM-EIoT with other related existing schemes, such as the schemes designed by Challa et al. [3], Farash et al. [21], Sharma and Kalra [39], Zhou et al. [41], and Turkanovic et al. [18], is done. It is worth noting that the schemes of Challa et al. [3], Farash et al. [21], Sharma and Kalra [39], Zhou et al. [41], and Turkanovic et al. [18] were designed for the IoT environment.

7.1. Comparative Study of Communication Costs

We took the identity, random nonce, hash digest (if the secure hash algorithm (SHA-1) [55] was utilized), and timestamp as 160, 128, 160, and 32 bits, respectively. It well known that the 1024 bit “RSA public key cryptosystem” maintains the same security level as that for the 160 bit “elliptic curve cryptography (ECC) cryptosystem” [56]. Thus, an elliptic curve point of the form \( P = (P_x, P_y) \) took \((160 + 160) = 320 \text{ bits} \). The communication costs of LDAKM-EIoT and other schemes are provided in Table 2. In LDAKM-EIoT, the messages \( M_{S1} = \{ TID_{D_i}, M_1, M_2, T_1 \}, M_{S2} = \{ TID_{D_i}, M_3, M_4, T_2 \}, M_{S3} = \{ M_5, M_6, M_7, T_3 \}, \) and \( M_{S4} = \{ M_8, \alpha, \beta, M_9, M_{10}, T_3, T_4 \} \) incurred the costs of \((160 + 160 + 160 + 32) = 512 \text{ bits} \), \((160 + 160 + 160 + 32) = 512 \text{ bits} \), \((160 + 160 + 160 + 32) = 512 \text{ bits} \), and \((160 + 160 + 160 + 160 + 160 + 32 + 32) = 864 \text{ bits} \), respectively, and hence, all the messages...
needed a total of \((512 + 512 + 512 + 864) = 2400\) bits. On the other side, the communication costs for the schemes of Challa et al. [3], Farash et al. [21], Sharma and Kalra [39], Zhou et al. [41], and Turkanovic et al. [18] were 2528, 2752, 2912, 3840, and 2720 bits, respectively. It is depicted in Table 2 that LDAKM-EIoT performed better in terms of communication cost as compared to other considered existing schemes of Challa et al. [3], Farash et al. [21], Sharma and Kalra [39], Zhou et al. [41], and Turkanovic et al. [18].

Table 2. Communication costs’ comparison. LDAKM-EIoT, the lightweight authentication and key management scheme for the edge based IoT environment.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>No. of Messages</th>
<th>No. of Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDAKM-EIoT</td>
<td>4</td>
<td>2400</td>
</tr>
<tr>
<td>Challa et al. [3]</td>
<td>3</td>
<td>2528</td>
</tr>
<tr>
<td>Farash et al. [21]</td>
<td>4</td>
<td>2752</td>
</tr>
<tr>
<td>Sharma and Kalra [39]</td>
<td>4</td>
<td>2912</td>
</tr>
<tr>
<td>Zhou et al. [41]</td>
<td>4</td>
<td>3840</td>
</tr>
<tr>
<td>Turkanovic et al. [18]</td>
<td>4</td>
<td>2720</td>
</tr>
</tbody>
</table>

7.2. Comparative Study of Computation Costs

The notations \(T_{ecm}\), \(T_{fe}\), and \(T_h\) are used to represent the time required for an “ECC point multiplication”, a “fuzzy extractor function” for biometric verification in the case of the scheme designed by Challa et al. [3], and a “one way hash function”, respectively. Based on the experimental results available in [57], we took \(T_h \approx 0.00032\) s, \(T_{ecm} \approx 0.0171\) s, and \(T_{fe} \approx T_{ecm}\), that is \(T_{fe} \approx 0.0171\) s. The computation costs’ comparison among LDAKM-EIoT and other schemes is depicted in Table 3. In LDAKM-EIoT, during the authentication and key agreement procedure, \(D_i\), \(E_N\), and \(CS_j\) incurred \(9T_h \approx 2.88\) ms, \(15T_h \approx 4.8\) ms, and \(8T_h \approx 2.56\) ms, respectively. The comparative results demonstrated that LDAKM-EIoT performed better than Challa et al.’s scheme [3] and Zhou et al.’s scheme [41]. Moreover, LDAKM-EIoT needed the same computation cost as compared to that for Farash et al.’s scheme [21]. However, LDAKM-EIoT required more computation cost as compared to that for Sharma and Kalra’s scheme [39] and Turkanovic et al.’s scheme [18]. However, this was accepted as LDAKM-EIoT provided extra security and functionality features as compared to those for the other compared schemes of Challa et al. [3], Farash et al. [21], Sharma and Kalra [39], Zhou et al. [41], and Turkanovic et al. [18] (see Table 4).

Table 3. Comparison of computation costs.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>User/Smart IoT Device</th>
<th>Gateway Node/Edge Node</th>
<th>Sensing Device/Cloud Server</th>
<th>Total Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDAKM-EIoT</td>
<td>(9T_h) \approx 2.88) ms</td>
<td>(15T_h) \approx 4.8) ms</td>
<td>(8T_h) \approx 2.56) ms</td>
<td>(32T_h) \approx 10.24) ms</td>
</tr>
<tr>
<td>Challa et al. [3]</td>
<td>(17T_{fe} + 5T_{ecm} + 5T_h) \approx 104.20) ms</td>
<td>(5T_{ecm} + 4T_h) \approx 86.78) ms</td>
<td>(4T_{ecm} + 3T_h) \approx 69.36) ms</td>
<td>(17T_{fe} + 14T_{ecm} + 12T_h) \approx 260.34) ms</td>
</tr>
<tr>
<td>Farash et al. [21]</td>
<td>(11T_h) \approx 3.52) ms</td>
<td>(14T_h) \approx 4.8) ms</td>
<td>(7T_h) \approx 2.24) ms</td>
<td>(32T_h) \approx 10.24) ms</td>
</tr>
<tr>
<td>Sharma and Kalra [39]</td>
<td>(11T_h) \approx 3.52) ms</td>
<td>(7T_h) \approx 2.24) ms</td>
<td>(5T_h) \approx 1.6) ms</td>
<td>(23T_h) \approx 7.36) ms</td>
</tr>
<tr>
<td>Zhou et al. [41]</td>
<td>(10T_h) \approx 3.2) ms</td>
<td>(7T_h) \approx 2.24) ms</td>
<td>(19T_h) \approx 6.08) ms</td>
<td>(36T_h) \approx 11.52) ms</td>
</tr>
<tr>
<td>Turkanovic et al. [18]</td>
<td>(7T_h) \approx 2.24) ms</td>
<td>(5T_h) \approx 1.6) ms</td>
<td>(7T_h) \approx 2.24) ms</td>
<td>(19T_h) \approx 6.08) ms</td>
</tr>
</tbody>
</table>
### Table 4. Comparison of functionality and security features.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>FSF1</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FSF2</td>
<td>×</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>FSF3</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FSF4</td>
<td>×</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FSF5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FSF6</td>
<td>×</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
</tr>
<tr>
<td>FSF7</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
<td>✓</td>
</tr>
<tr>
<td>FSF8</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
<td>✓</td>
</tr>
<tr>
<td>FSF9</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
<td>✓</td>
</tr>
<tr>
<td>FSF10</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FSF11</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FSF12</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
</tr>
<tr>
<td>FSF13</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
</tr>
<tr>
<td>FSF14</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
</tr>
<tr>
<td>FSF15</td>
<td>NA</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>NA</td>
</tr>
<tr>
<td>FSF16</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FSF17</td>
<td>×</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>NA</td>
</tr>
<tr>
<td>FSF18</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Note:** FSF1: the property to make the user or smart IoT device anonymous; FSF2: protection for privileged-insider attack; FSF3: protection for off-line password guessing attack; FSF4: protection for stolen smart card or mobile device attack; FSF5: protection for denial-of-service attack; FSF6: protection for user impersonation attack; FSF7: protection for man-in-the-middle attack; FSF8: achieves mutual authentication; FSF10: achieves session key agreement; FSF11: property to make messages untraceable; FSF12: protection for sensor node/sensor/smart IoT device physical capture attack; FSF13: presence of server independent password update phase; FSF14: protection for sensor node/sensing device/smart IoT device impersonation attack; FSF15: support biometric update phase; FSF16: provide formal security verification using the automated software verification tool; FSF17: presence of the smart card revocation phase; FSF18: protection for session-key security under the CK adversary model. ×: insecure against a “specific attack” or a “particular feature” is not there; ✓: secure against a “specific attack” or a “particular feature” is present; NA: not applicable.

### 7.3. Comparative Study of Security and Functionality Attributes

The comparison of the security and functionality attributes for the proposed LDAKM-EIoT and other related schemes is shown in Table 4. Various features (FSF1-5, FSF10, FSF13, and FSF18) were not supported/available in the scheme of Farash et al. [21], whereas Challa et al.’s scheme [3] did not support the features FSF11 and FSF14, which were shown by Jia et al. [37]. Moreover, other schemes, such as the schemes of Sharma and Kalra [39], Zhou et al. [41], and Turkanovic et al. [18] did not provide the required functionality and security features. On the other hand, LDAKM-EIoT provided better security and functionality features as compared to other related schemes.

### 8. NS2 Simulation Study

In this section, we conduct the simulation study on LDAKM-EIoT and other existing related schemes of Challa et al. [3], Farash et al. [21], Sharma and Kalra [39], Zhou et al. [41], and Turkanovic et al. [18] using the NS2 simulator. The impact of LDAKM-EIoT and other related schemes is studied and compared on important network performance parameters (for example, throughput (available in bits per second (bps)) and end-to-end delay (available in seconds)).

**8.1. Simulation Parameters**

The specifics of the network parameters used in the NS2 simulation are given in Table 5. The simulation time was 30 min (1800 s). $S_j$ and $C_S$ respectively represent the $j$th sensor node and cloud server, whereas $U_i/D_i$ represents the $i$th user/IoT smart device respectively in LDAKM-EIoT and other schemes [3,21]. Moreover, we took one gateway node (GW) / edge node (EN) in all the compared schemes. The communication range of smart device or sensor was taken as 100 m. The messages
exchanged among different communicating entities along with their communication costs (in bits) for different schemes are depicted in Table 6, which were used in the simulation for exchanging the packets (messages) among the entities. We applied the ad hoc on-demand distance vector (AODV) [58] as the routing protocol. All other parameters used in the simulation were taken as standard ones.

**Table 5.** Different parameters used in the simulation. AODV, ad hoc on-demand distance vector.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform</td>
<td>Ubuntu 14.04 LTS</td>
</tr>
<tr>
<td>Tool used</td>
<td>NS2 2.35</td>
</tr>
<tr>
<td>Number of gateway nodes/edge nodes (whenever applicable)</td>
<td>1</td>
</tr>
<tr>
<td>Number of users or IoT device (whenever applicable)</td>
<td>9</td>
</tr>
<tr>
<td>Number of sensors or cloud server (whenever applicable)</td>
<td>10</td>
</tr>
<tr>
<td>Simulation time</td>
<td>1800 s</td>
</tr>
<tr>
<td>Communication range of sensors/IoT devices</td>
<td>100 m</td>
</tr>
<tr>
<td>Routing protocol</td>
<td>AODV [58]</td>
</tr>
</tbody>
</table>

**Table 6.** Different messages exchanged among entities used in the simulation.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$U_i/D_i \rightarrow GW/EN$</td>
<td>992 bits</td>
<td>512 bits</td>
<td>672 bits</td>
<td>672 bits</td>
<td>800 bits</td>
<td>512 bits</td>
</tr>
<tr>
<td>$GW/EN \rightarrow SD_j/S_j/CS_j$</td>
<td>1024 bits</td>
<td>1024 bits</td>
<td>1024 bits</td>
<td>1024 bits</td>
<td>1600 bits</td>
<td>512 bits</td>
</tr>
<tr>
<td>$SD_j/S_j/CS_j \rightarrow GW/EN$</td>
<td>–</td>
<td>672 bits</td>
<td>576 bits</td>
<td>672 bits</td>
<td>960 bits</td>
<td>512 bits</td>
</tr>
<tr>
<td>$GW/EN \rightarrow U_i/D_i$</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>$SD_j/S_j/CS_j \rightarrow U_i/D_i$</td>
<td>512 bits</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

8.2. Comparative Analysis of the Simulation Results

The comparative analysis of the network performance parameters for LDAKM-EIoT, Challa et al. [3], and Farash et al. [21] is discussed in the following.

8.2.1. Comparative Analysis of the Network Throughput

The network throughput can be calculated as “the number of bits transmitted per unit time, and it can be mathematically expressed as $(\nu_r \times |\rho|)/T_d$, where $T_d$ is the total time (in seconds), $|\rho|$ the size of a packet, and $\nu_r$ the number of received packets”. The simulation time was 1800 s, which was the actual total considered time. In Figure 5a, the throughput values for the schemes of Challa et al. [3], Farash et al. [21], Turkanovic et al. [18], Sharma and Kalra [39], Zhou et al. [41], and the proposed LDAKM-EIoT are 173.49 bps, 164.34 bps, 160.41 bps, 169.85 bps, 217.16 bps, and 148.95 bps, respectively. It was observed that the throughput of LDAKM-EIoT was less than the other schemes [3, 18, 21, 39, 41]. This was primarily due to the reason that LDAKM-EIoT applied the small sized messages, which caused less communication overhead for the authentication procedure as compared to the other schemes (see Table 6).
8.2.2. Comparative Analysis of the End-to-End Delay

The end-to-end delay (EED) is measured as “the average time taken by the data packets to arrive at a receiving node from a sender node, and it is mathematically expressed in the form \(\frac{\sum_{i=1}^{v_p} (T_{\text{rec}_i} - T_{\text{send}_i})}{v_p}\), where \(T_{\text{rec}_i}\) and \(T_{\text{send}_i}\) are the receiving and sending time of a packet \(i\), respectively, and \(v_p\) the total number of packets”. From the simulation results provided in Figure 5b, it was observed that EED values for the schemes of Challa et al. [3], Farash et al. [21], Turkanovic et al. [18], Sharma and Kalra [39], Zhou et al. [41], and LDAKM-EIoT were 0.17230 s, 0.15151 s, 0.12478 s, 0.12496 s, 0.13707 s, and 0.11640 s, respectively. The EED value of LDAKM-EIoT was less than the schemes provided in [3,18,21,39,41]. This was again due to the reason that LDAKM-EIoT applied small sized messages for the authentication procedure, which caused less end-to-end delay as compared to other schemes.

9. Conclusions

The edge based IoT environment suffers from serious security and privacy issues. It was observed that the majority of the existing schemes for authentication and key management have limitations as they were vulnerable to various attacks. Some of them were not even efficient from the computation and communication cost point of view. Consequently, a new authentication and key management scheme with lightweight cryptographic operations was designed for the edge based IoT environment (LDAKM-EIoT). The rigorous security analysis of LDAKM-EIoT using formal security verification using AVISPA tool and also other security analysis evidenced that LDAKM-EIoT could prevent different possible well known attacks. LDAKM-EIoT also supported new smart IoT device deployment in the network after the initial deployment. Moreover, LDAKM-EIoT preserved the anonymity and untraceability properties. LDAKM-EIoT was also compared with the closely related existing schemes in the IoT environment. The conducted comparative performance analysis and NS2 based simulation study on the network performance parameters evidenced that LDAKM-EIoT was much better in security and functionality features, communication, and computation as compared to other existing schemes.

In the future, we would like to implement the proposed LDAKM-EIoT in a testbed environment. Moreover, we would also like to include more functionality features in the proposed LDAKM-EIoT while maintaining less communication and computational overheads without degrading the security.


Funding: This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning (2017R1A2B1002147); in part by Fundação para a Ciência e a Tecnologia through the UID/EEA/50008/2019 Project; and in part by Brazilian National Council for Scientific and Technological Development (CNPq) via Grant No. 309335/2017-5.
This work was also supported by the Office of the Assistant Secretary of Defense for Research and Engineering (OASD (R&E)) agreement FA8750-15-2-0120.

Acknowledgments: We thank the anonymous reviewers and the Editor for their valuable comments, which helped us to improve the quality and presentation of the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References
7. Glissa, G.; Meddeb, A. 6LowPSec: An end-to-end security protocol for 6LoWPAN. Ad Hoc Netw. 2019, 82, 100–112. [CrossRef]


32. Gope, P.; Sikdar, B. Lightweight and Privacy-Preserving Two-Factor Authentication Scheme for IoT Devices. *IEEE Internet Things J.* 2019, 6, 359–370. [CrossRef]


© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).