Security Enhancement on a Lightweight Authentication Scheme With Anonymity Fog Computing Architecture

LIN WANG\(^1,2\), (Member, IEEE), HAONAN AN\(^1\), AND ZHUO CHANG\(^1,3\)

\(^1\)School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China
\(^2\)Networked Sensing and Big Data Engineering Research Center of Hebei Province, Qinhuangdao 066004, China
\(^3\)School of Cyber Security and Computer, Hebei University, Baoding 071000, China

Corresponding author: Lin Wang (wlin@ysu.edu.cn)

This work was supported in part by NFSC under Grant 61772453, and in part by the Natural Science Foundation of Hebei Province under Grant F2018203444.

ABSTRACT The multi-level, heterogeneous and third-party attributes of fog computing (FC) have caused great concern in the communication security of next-generation IoT systems. This paper proposes a secure authentication scheme for dynamic key generation capability, which can enhance the security of the entire heterogeneous network without the constraints on device types, attributes and communication protocols. The communication devices exchange IDs and random numbers for registration, then generate anonymous information. During the authentication process, the cloud device (e.g., the server) uses the pseudonym information to match the recorded random numbers. Both FC devices utilize the protocol with such random numbers to generate session keys and deliver updated random numbers. Comprehensive security and performance analysis shows that the scheme can meet the security requirements of the FC architecture with lower overhead.

INDEX TERMS Authentication protocol, key generation, key share, IoT security, fog computing.

I. INTRODUCTION

With the popularity of the Internet of things (IoT), more and more information needs to be shared by wireless communication. However, due to the exposed nature of wireless signals, how to authenticate devices and ensure the information security in wireless communication becomes a huge challenge. Especially in the IoT system, the devices are often low computational capacity and low power consumption, which makes it necessary to reduce the cost of authentication and encryption while ensuring information security. Because the transmitted data onto IoT devices often involves certain privacy needs, we also need to avoid the third party knowing which device sends what kind of information. In addition, many wireless communication protocols in IoTs have different transmission speeds and connection modes, such as Z-Wave, Wi-Fi, Zigbee, 5G, etc [1]. Therefore, designing a universal, deployable dynamic key generation and device authentication method for various wireless protocols is essential.

In order to carry out multi-source heterogeneous data onto different devices more easily, avoid a lot of bandwidth load and resource waste, improve real-time service, and ensure security and privacy issues, the fog computing (FC) is proposed. Authentication is an important issue for the security of FC since services are offered to massive-scale end users by front fog nodes [2]. FC usually includes multiple authentication entities, such as edge devices (sensors, mobile phones, etc.), fog devices (gateways, base stations, small servers and other devices with computing, storage, network connectivity capabilities), and cloud data centers. In FC, different devices in different layers need to successively authenticate and transmit information from edge to cloud according to their needs. At the same time, fog layer devices need to authenticate and exchange data with each other and share the stored data and calculation results [3]. Therefore, we need a lightweight, anonymous and secure authentication method for IoT devices, which needs to meet the different needs of FC, e.g., mutual authentication between fog devices,
session key update, group key generation and multi-level fog device extension.

To offer reliable services and protect data privacy, a variety of security protocols should be implemented on fog nodes [4]. Generally speaking, different authentication protocols need to transmit sensitive data by some encryption methods. It is also a challenge to adopt a more effective encryption method and update the key. Traditional encryption methods are generally divided into symmetric encryption and asymmetric encryption. Asymmetric encryption is not suitable for IoT devices with limited capability due to its high complexity, while symmetric encryption is easy to deploy and has high encryption and decryption efficiency. However, in symmetric encryption, both sides of communication share the same pair of keys. Once the key is known by the third party, or the third party just analyzes the network traffic, the communication content will be inferred. Therefore, the long-term use of a fixed session key is insecure among fog devices.

In order to update the key dynamically, the generation of symmetric key by using the physical layer information in wireless communication, i.e. channel state information (CSI) [5]–[8], has been concerned. However, the premise of such works is that CSI or received signal strength (RSS) needs to change constantly. Once the environment is static, the CSI/RSS will remain unchanged for a long time, the key cannot be updated. In addition, this kind of work also needs to assume that the antenna of the eavesdropper is not too close to the legitimate device, because when the antenna of the eavesdropper is connected with the legitimate device, the enemy can also eavesdrop the same CSI/RSS, to obtain the generated key between the legitimate devices. This kind of work can only be used to generate a consistent key after two devices are authenticated successfully, it does not matter how to authenticate each other.

Some other works [9]–[14] use the protocol to authenticate and dynamically update the key. However, these works can only be applied to authenticate or update key between smart device and gateway. The application scenario is single and cannot be extended to the authentication between different devices in FC. Moreover, these works don’t pay attention to anonymity and the authentication cost [9], [12], [13].

In this paper, we propose a lightweight scalable authentication method with anonymity for fog computing architecture. This model will face the following challenges: 1) It can be used for the devices in the entire FC architecture. 2) It is lightweight and has lower energy consumption to fit the characteristics of most IoT nodes. 3) It can ensure the dynamic session key generation and anonymity to provide higher security.

In response to the above challenges, our method takes the advantages of anonymous authentication protocol and physical layer dynamic key generation to meet the authentication and encryption requirements in the FC architecture. This method uses symmetric encryption to encrypt some sensitive data and uses the hash result of the combination of some random numbers and dynamic secret key generated by both sides as our session key. The pseudonym is updated in each time of authentication to ensure the anonymity. The specific contributions can be summarized as follows:

1) We design a lightweight anonymous authentication protocol for multi-level devices under the FC architecture. The protocol can dynamically update the session key adopted by both sides while ensuring anonymous authentication of the devices.

2) We propose a group key management protocol. The server can share the key to the desired communication fog nodes with a specific attribute, and the private key can be generated and updated between the two fog nodes without leaking keys to the servers.

3) We present comprehensive security and performance analysis of the proposed scheme. The analysis results show that the proposed protocols can prevent many attacks and ensure data security between wireless devices in the FC architecture with lower overhead.

The remainder of this paper is organized as follows. We provide some related works in Section II and present the detailed system model in Section III. Section IV and Section V describe the design of anonymous authentication protocol and the management scheme of group secret key respectively. Security analysis and protocol evaluation are illustrated in Section VI and Section VII. We conclude the whole paper in Section VIII.

II. RELATED WORK

A. DEVICES AUTHENTICATION IN IoT

There are some works [15], [16] that use face or other biometric features to ensure device security. However, computing and communication overhead will increase linearly with the expansion of the database. Some works such as [17], [18] consider solving this problem by cloud computing. Reference [19] analyzes and summarizes the security and privacy problems faced by this kind of work under the fog computing model, and proposes authentication, data encryption, and data integrity checking schemes with less communication cost. Such work requires the participation of human.

Some works use smart card to achieve anonymous authentication [20], [21]. However, these works introduce smart card, which limits the flexibility of use.

Reference [22] proposes a new resource-efficient physical unclonable function (PUF) based authentication scheme to protect the security and privacy of the confidential information in edge devices. However, this method needs some modification to the hardware of IoT devices.

The protocol proposed in [23] forces all fog nodes to store certain credential information of all the users in the trust domain, which costs large storage in each fog node. All the real identity of fog users and nodes are publicly transmitted, which makes the system not anonymous.

Subsequently, the elliptic curve cryptography (ECC) and the asymmetric encryption are exploited by an anonymous mutual authentication scheme for fog nodes in [24]. Although
this work claims that confidential communication is guaranteed and no computation is needed in the authentication phase, it does not take message integrity or edge compromise resilience into account [25]. Due to the use of asymmetric encryption, the overhead may be high in some stages [26]. Using public-key infrastructure is not suitable due to the resource constraints of IoT devices [3].

In order to overcome the shortage of using asymmetric encryption method in [24], [27] proposes a method only use symmetric encryption and the hash-based Message Authentication Code (HMAC) to authenticate the devices. However, this work still used a long-term master secret to generate new session keys.

Reference [26] proposes a method uses the polynomial formed by the combination of the identity of fog node and fog user as the key, which exhibits a good performance in terms of communication, computation, and storage space for IoT devices [28]. However, they don’t provide a complete authentication protocol and don’t consider communication links between IoT devices.

These works mentioned in [23], [24], [26], [27] are all inconvenient in updating a new session key, and all need a central authentication server in the cloud layer to distribute the session key, which makes it inconvenient when register a new device, and the date transmitted known by the server. Different from these works, our method has no long-term master key, and updates the new session key conveniently, which makes the communication more secure.

After pointing out that many works [9], [13], [29] are high overhead and cannot guarantee anonymity and unlinkability, [11] proposes an anonymous lightweight authentication protocol for smart home devices. Although this work claims to be anonymous, the protocol is not anonymous to malicious devices connected in the same gateway. In addition, if some secret information, such as $TK$ has been guessed out, all the secret information will be cracked.

**B. DYNAMIC KEY GENERATION**

In order to update the symmetric key dynamically, the scheme of dynamic secret key generation using CSI and RSS can be used in many wireless communications. The principle of key generation based on physical layer information of wireless channel is channel reciprocity, temporal variation and spatial decorrelation [30]. Such random change of CSI can give us a random source for key generation. Spatial decorrelation means that if the distance between an eavesdropper (e.g., Eve) and legitimate device is greater than $\frac{\lambda}{2}$ ($\lambda$ is the wavelength of the wireless signal), Eve cannot obtain the same CSI between Alice and Bob, so Eve cannot obtain the same key between them. There are many related works in many wireless bands, such as IEEE 802.11 [31], [32], IEEE 802.15.4 [33], Bluetooth [34], UWB [35], LTE [7], 5G [36], [37] etc.

Using these schemes, two communication parties can get similar CSI/RSS, and then generate the same key. However, these works often focus on how to use channel state to get random and consistent keys but ignore the need for authentication between devices. Even more, once the channel state remains stable, the key will not be updated in real time.

Our protocol is not only anonymous and lightweight, but also can be applied to the authentication scenarios among sensors and gateways. It can also be extended to the FC architecture for device authentication and key update.

**III. SYSTEM MODEL**

**A. SYSTEM ARCHITECTURE**

We consider an architecture of FC, which contains different levels of wireless devices, as shown in Figure 1.

The server is connected with the authentication server, which stores the identity and key information of all subordinate devices (e.g. gateways which communicate with server directly) in the region. All its subordinate devices need to be authenticated to obtain the session key before communicating with each other. Gateway A, B and C are fog layer devices, which exchange information with other devices (e.g., server, gateway, sensor) through wireless signal(e.g., Wi-Fi, Zigbee etc.). Gateway A is the peer device of Gateway B. Because the distance from Gateway C to Server is too far, Gateway C can be the subordinate device of Gateway B. A1, A2 and C2 are edge devices, which only authenticate and exchange information with their superior device A, B and C.

Note that our protocol can be extended to a multiple level architecture. For example, multiple gateways can be connected under Gateway C, so that the superior gateway is equivalent to a small authentication server.

**B. THREAT MODEL**

We consider the Dolev-Yao attack model [38], which is widely used in the research of security protocols. This model assumes that the attackers have the following capacity:

1) Eavesdrop and intercept any message passing through the network.
2) Store the intercepted messages and the messages he constructs.
3) Send the intercepted messages and the messages he constructs to others.
4) Participate in the protocol as a legitimate user.

In order to realize the above assumption of Dolev-Yao attack model, the eavesdropper can be placed in any range to carry out passive attacks and active attacks, to
eavesdrop, intercept, change, replay the massage. In addition, the attacker may pretend to be a legitimate device to deduce the information of other legitimate devices.

We assume that the Server is tamper proof, and will not cooperate with other devices to make active attack. The hash functions, symmetric encryption functions, and protocols that all devices used are publicly known.

As shown in Figure 1, we consider the Server is trusted but curious. Therefore, under some privacy requirements, Gateway A and Gateway B still do not want the transmitted date known by the Server when they communicate. However, the key of Gateway A and Gateway B is distributed by Server. So, they also need a mechanism to avoid the Server knowing their session key.

IV. ANONYMOUS AUTHENTICATION PROTOCOL

In this section, we will introduce the details of the protocol. It is mainly divided into the following parts: registration stage, authentication stage, group key distribution and private key establishment stage. In the registration phase, authentication information is exchanged between the subordinate device and the superior device. In the authentication phase, the subordinate device sends a request to the superior device, and the two sides use hash function to authenticate anonymously and generate session key. In the phase of group key distribution and private key establishment, the Gateways will obtain the consistent key through the server, and initialize the private key by analyzing the channel state. All edge nodes can also register on the Gateway in this way.

The notations are listed in the Table 1 and functions are listed in the Table 2.

TABLE 1. Notations in the protocol.

| Notation | Meaning |
|----------|---------|
| id_A / id_s | Real identity of Gateway A / Server |
| R_A / R_s | Random number generated |
| R_a | A new version of R_a |
| R_a* | Unchecked R_a received from other device |
| CID_A | Pseudonym of Gateway A |
| key_dy | Dynamic key generated using CSI |
| n | Reused times of key_dy |
| p | Prop of the device |
| TK | Secret key used for authentication |
| SK | Session key after authentication |

TABLE 2. Functions in the protocol.

| Function | Description |
|----------|-------------|
| ET_K | Encryption with symmetric key TK |
| DT_K | Decryption with symmetric key TK |
| h | Hash function |
| ⊕ | XOR operation |
| || | Concatenation operation |

A. REGISTRATION STAGE

In this stage, the two communication parties exchange identity information and session key related information securely (e.g., random numbers). The detailed process is shown in Figure 2. We assume that the identity of Gateway A and Server are id_A and id_s, respectively. It includes following steps:

Step 1: Gateway A generates a random number R_A and sends id_A and R_A to Server.

Step 2: After receiving the message from Gateway A, Server generates a random number R_S and sends id_S and R_S to Gateway A. Gateway A and the Server exchange the identity and random number through a secure channel. And the dynamic key generation method using CSI described above is used to generate the dynamic key of both parties when they are transmitting the message. The first l bits of the dynamic key is key_dy, and n is the reused times of key_dy.

Step 3: Gateway A and Server store mutual information id_S, R_S, id_A, R_A, key_dy, n, and generate CID_A = h(id_A ⊕ key_dy ⊕ n) for later identification.

To make CID_A changed in time when key_dy can’t be updated, we use n to change CID_A. Due to the use of hash function, even when n change a little, the hash result of CID_A will change a lot, which can’t be tracked by adversary. So that even the CSI remains stable for a long time, the CID_A can be updated in time.

The secure channel can be some other channels to transmit secret information except wireless signal, such as, using an encrypted wired network to register devices remotely, or saving information on two devices in the different places manually. In a word, it’s a secure way to transmit information to different devices.

B. AUTHENTICATION STAGE

Before the communication between Gateway A and Server, they need to authenticate each other using their stored random number and generate the session key. The authentication process mainly includes four steps as shown in Figure 3.
Step 1: Gateway A calculates the variables as shown in Step 1 in Figure 3. $CID_A$ is the pseudonym of Gateway A, which is used to confuse the real identity information while authenticating on Server. $p$ is some attribute information of Gateway A. The devices with specific attribute can share the session key. $C_1$ is used to transmit some information encrypted with TK to Server. The above variables connected with Exclusive OR symbol $\oplus$ are used to avoid the adversary’s exhaustive attack to infer the contents of hash or ciphertext. $T_1$ is timestamp.

Step 2: After receiving the timestamp $T_1$, Server compares the difference between $T_1$ and the current timestamp $T_2$, to determine whether there is an intermediary tampering with the information. Then Server searches the saved variables $CID_A$ in the database, and extracts the corresponding variables for the next calculation and verification. $SK$ is used to be a new session key. When update key $dy$, if the number of key bits generated by channel state dynamically is enough, we let key $dy$ be next part of dynamic key, and let $n' = 0$; if not enough, the key $dy$ keeps unchanged and let $n' = n + 1$, which means the times of reuse.

Step 3: After receiving the returned message, Gateway A performs the calculation shown on the flowchart. Among them, the calculated results of $V_1$ can be used to verify the identity of Server, judge whether it has received the accurate $id_A$ and $R_A'$, and whether Gateway A received the accurate $R_S'$ and key $dy$. After confirming the identity and variables, Gateway A generates and updates all variables, and confirms the information with Server.

Step 4: After confirmation, the Server updates all variables and the pseudonym of Gateway A. We should note that in order to avoid desynchronization attack when confirm the session key, before Gateway A and Server can communicate with each other using new session key, they should store the former variables.

**C. MULTI-LEVEL DEVICE EXTENSION**

Our protocol can be extended to more levels in FC architecture. Each extension only needs to use the above protocol to register the new device on other nodes. For example, let Gateway C register on Gateway B and then let Gateway B be the server of Gateway C.

**V. SHARING KEY AMONG MULTIPLE FC NODES**

In most cases, FC nodes need to share secret key with each other. However, because the session keys are distributed by the server, there needs a method to make the FC nodes authenticate with each other while making the key shared among them cannot be leaked to the server. In this section, we will introduce our method.

**A. GROUP SECRET KEY GENERATION**

If Gateway A wants to share the key with the devices with the same attribute, during the authentication process, make the variable $p$ represent some attribute information of Gateway A, such as can/cannot share the key with other nodes. The Server can identify this variable to determine which devices to share the secret key with it.

When Server wants Gateway A and Gateway B to share the same key, the generated random number $R'_G$ can be modified to make the $SK$ of Gateway A and Gateway B consistent.

**B. PEER PRIVATE KEY GENERATION**

Because all authentication and encryption information of Gateway A and Gateway B are from Server, Server can know the communication content of both sides. In some cases of privacy requirements, fog devices do not want the superior equipment to know the results of computing, because the superior equipment is trusted but curious. We need an encryption method to enable Gateway A and Gateway B to obtain secret key that couldn’t be known by the superior device. At this time, we use the dynamic key generation method based on channel state information mentioned in the section 2 to coordinate the private key.

As shown in the Figure 4, the two parties should carry out the following steps respectively:
Step 1: Gateway A uses $C_{IA}$ to transmit pseudonym $cid_A$ and generates a random number $R_A$. Uses $C_1$ to transmit the encrypted variables to Gateway B.

Step 2: After calculating $cid_A$ and $R_A$, Gateway B uses $C_{IB}$ to transmit pseudonym $cid_B$ and generates a random number $R_B$. Uses $V_1$ to transmit the encrypted variables to Gateway A.

Step 3: After Gateway A obtaining the information of Gateway B and authenticates it successfully, it updates the variables, and sends a dynamic key generation request to Gateway B. It uses the results of channel state information measurement during the communication between the two parties to generate a dynamic key $key_{dy}$. In this way, the two devices of the same level in FC architecture can generate mutual authentication information, and use the previous authentication process mentioned in the authentication stage to authenticate and update the key.

After authenticating and updating the key, these two devices can either authenticate each other in the way with no central server, or make the device with high performance as the server in a certain way.

VI. SECURITY ANALYSIS

In this section, we will discuss the security of our protocol. Some security performance compared with other works are listed in the Table 3. The complete analysis of the prevention of attacks are explained in the subsection.

In Table 3, the anonymity of [11] is incomplete (use Inc instead), because the devices under the same gateway will know the identity of each other. Data integrity means that there is integrity verification of sensitive variables in the process of authentication. Fog layer authentication means that the gateways in the fog layer can authenticate each other. The reason that [24] is not lightweight is that it uses asymmetric encryption method to transmit information.

A. ANONYMITY

**Definition:** Anonymity means that no one can know the real identity of the device through the transmitted information except trusted devices.

**Theorem:** Our scheme guarantees the anonymity of both sides of communication.

**Proof:** We suppose that the adversary wants to eavesdrop all the transmitted information to get the real identity of the legitimate device. However, the adversary can only know pseudonym $C_{IA} = h(id_A \oplus key_{dy} \oplus n)$. This information is the hash result of the Exclusive OR result of dynamic key $key_{dy}$, reuse times $n$ and real identity $id_A$, which will be changed in each authentication, and there is no regular pattern. Only the server and the device itself can obtain the identity and variable information related to the device through the pseudonym. Although the message $C_1$ and $V_1$ contain the identity of $id_s$, they are encrypted using symmetric key which can be updated after authentication, so the adversary cannot get $id_s$.

B. UNLINKABILITY

**Definition:** Unlinkability means that it is impossible for an adversary to associate two authentication actions of the same device.

**Theorem:** Our scheme guarantees the unlinkability of messages from the same device.

**Proof:** In our protocol, the messages publicly transmitted are different every time, and the messages are encrypted or hashed information, between which there is no regularity, so the adversary cannot know the exact meaning and relationship between former information.

C. MUTUAL AUTHENTICATION

**Definition:** Mutual authentication is a security process in which both devices authenticate each other’s identities before actual communication occurs.

**Theorem:** Our scheme guarantees the mutual authentication of both the two devices which need to authenticate each other according to Lemma 1 and Lemma 2.

**Lemma 1:** Server can authenticate Gateway A.

**Proof:** $C_{IA}$ is the hash result of the combination of $id_A$, $key_{dy}$ and $n$, which is only known to the Server and Gateway A. After receiving $C_{IA}$, Server will check $C_{IA}$ in database
to get $id_A, R_A, key_{dy}, n, id_S$ corresponding to it. Server will use these information to compute $TK$. If Gateway A also has these information, $C_1$ will be encrypted using $TK$, which can be decrypted by Server. After decrypting, if Server gets the plaintext, Server can successfully authenticate the Gateway A.

Lemma 2: Gateway A can authenticate Server.

Proof: After authentication on Gateway A, Server will send $V_1$ to Gateway A, which is encrypted with secret key $TK$. If Server is the trusted device, $V_1$ can be decrypted by Gateway A using $TK$, thus Gateway A can authenticate Server. Further more, Gateway A can also check $key_{dy} \oplus R'_S \oplus R_S$ and $R'_A \oplus R'_S$ in $V_1$ to make sure that whether Server has the same information.

D. DATA INTEGRITY

Definition: Data integrity refers to the accuracy and reliability of the transmitted data, which is confirmed by another method.

Theorem: Our scheme guarantees the integrity of transmitted data.

Proof: After receiving $C_1$, Server can check the integrity of some information, such as $R_A, key_{dy}, T_1$, through the analysis of $CID_A, TK, C_1$. If these information are not integrity, the result of $CID_A, TK, C_1$ will be different. After receiving $V_1$ from Server, Gateway A can check whether it received accurate $R'_S$ and Server received accurate $R'_A$ using $key'_{dy} \oplus R'_S \oplus R_S$ and $R'_A \oplus R'_S$ encrypted in the $V_1$.

E. RESISTANCE TO REPLAY ATTACK

Definition: Replay attack means that the adversary pretends to be a legitimate device by replaying the messages sent by the legitimate device.

Theorem: Our scheme guarantees the resistance to replay attack.

Proof: In our protocol, all messages will be changed in different authentication, and some key messages contain timestamp. In the protocol of authentication stage, suppose that when Gateway A sends authentication request, the adversary blocks and eavesdrops $CID_A, C_1$ and $T_1$, wants to replay these messages at another time to pretend to be legitimate users. The adversary cannot modify the timestamp in $C_1$. Even if the server receives $CID_A, T_2 - T_1 > \Delta T$ will be found after decrypting $C_1$. Similarly, when the server returns $V_1$, the enemy cannot modify the timestamp in $V_1$. Even if the adversary cracked $C_1$ and modified the timestamp, sent the authentication request to the server and got the reply $V_1$ from the server, the session key could not be established, because some variables were covered with XOR, and the adversary doesn’t know some variables, such as $R_A$ and $R_S$.

F. RESISTANCE TO EXHAUSTIVE ATTACK

Definition: Exhaustive attack means that the adversary may try to obtain the plaintext by exhausting the keys and hash results using a very high performance computer. And when adversary get the secret information, he may calculate the key in the next authentication easily.

Theorem: Our scheme guarantees the resistance to exhaustive attack.

Proof: In our protocol, all sensitive information in messages, such as $id_A, R_S, R_A$, perform XOR operations with the known information. Even if the adversary gets the plaintext, they cannot know the exact values of these variables, thus cannot get the key used in the next authentication.

G. RESISTANCE TO DESYNCHRONIZATION ATTACK

Definition: Desynchronization attack means that the adversary interferes with the whole authentication process by blocking messages in the authentication process, and makes the legitimate users unable to authenticate in the future.

Theorem: Our scheme guarantees the resistance to desynchronization attack.

Proof: In previous systems, if the legitimate device and server cannot synchronize the pseudonym, they must use the real identity to authenticate or re-register, which will expose the real identity. However, in our protocol, on the one hand, we need to keep the previous variables before the authentication to recovery after authentication failure. On the other hand, even if the authentication fails and the pseudonym cannot be synchronized, we can use the dynamically generated key $key_{dy}$ to generate the pseudonym $CID_A$ to hide the real identity. Since the dynamically generated key can be generated by both parties at the same time, even if the synchronization of pseudonym fails, both parties can generate another dynamic key to generate a new pseudonym. The server can know which device the pseudonym belongs to by hashing each real identity with $key_{dy}$.

H. RESISTANCE TO IMPERSONATION ATTACK

Definition: Impersonation attack means that the adversary communicates with the server by pretending to be a legitimate user, such as Gateway A, and forging authentication messages.

Theorem: Our scheme guarantees the resistance to impersonation attack.

Proof: In this case, even if the adversary knows the real identity $id_A$ of Gateway A, and generates a dynamic $key_{dy}$ with the server, to obtain $CID_A$, the adversary also does not know the random number $R_A$ and $R_S$ used in the previous communication, and cannot carry out the next session key.

I. FORWARD SECRECY

Definition: Forward secrecy refers to that the elements used to generate the key are changed everytime, and no regular pattern between keys; if one key is cracked, the security of other keys will not be affected.

Theorem: Our scheme guarantees the forward secrecy.

Proof: In our protocol, when Gateway A and Server authenticate, random numbers $R'_A$ and $R'_S$ will be generated independently. The new session key $SK$ only uses the hash result of the combination of $R'_S, R_A, R_S$ and dynamic key
key\_dy, which has no relation with the previous random number. Even if SK is cracked, it is impossible to infer any information of the random number that constitutes the key. The key TK used in the authentication process is the result of exclusive or. Since RA is not used to generate the last round of session key SK, the adversary cannot know TK.

**J. RESISTANCE TO DEVICE COMPROMISED THREAT**

**Definition:** Device compromised threat means that the adversary can approach the device for physical attack to get secret information.

**Theorem:** Our scheme guarantees the resistance to device compromised threat.

**Proof:** We assume that the adversary can compose the gateway and connect its own antenna to the antenna of the gateway to obtain the dynamic key key\_dy generated each time. This method can break the previous schemes that only use channel state information to generate key dynamically. However, in our protocol, due to the random numbers RA and RS are generated independently by both parties, the adversary cannot know what the session key is. And because of the reuse number n, the adversary cannot know which part of dynamic key is used.

**K. SECURITY VERIFICATION USING AVISPA**

We adopt an industrial-strength security protocol analysis toolset, AVISPA [39], with HLPSL (The High Level Protocol Specification Language) to verify the security of the proposed protocols.

After transforming the HLPSL script into IF (Intermediate Format) using the translator HLPSL2IF, we can see that our goal and the request are shown in Figure 5, Figure 6 and Figure 7. The variable Ra2, Rs2, IDa, Dy1, Dy2 in the program are corresponding to R\_A, R\_S, id\_A, key\_dy, key\_dy in the protocol.

Finally, our proposed protocol is proved safe in the OFMC backend and CL-AtSe backend.

**L. BAN LOGICAL PROOF**

We can use BAN logic [40] to prove the security of our protocol. In our protocol, we expect that Gateway A and Server can authenticate each other and communicate with the same session key TK. We can verify that the newly generated session key SK and random number R\_A, R\_S are fresh. In Ban logical proof, we can prove the following goals. The details about Ban logical proof can be seen in appendix.

\[
\begin{align*}
G1: & \quad GA \equiv S \equiv GA \xrightarrow{TK} S \\
G2: & \quad S \equiv GA \equiv S \xrightarrow{TK} GA \\
G3: & \quad GA \equiv \#(R\_S, SK) \\
G4: & \quad S \equiv \#(R\_A, SK)
\end{align*}
\]

**VII. PERFORMANCE ANALYSIS**

In this section, we analyze the computation and traffic overhead of our protocol (ignoring the overhead of peer private key generation). In addition, the overhead of dynamic key generation using channel state information is not discussed in this work. We use Advanced Encryption Standard (AES) symmetric encryption algorithm to encrypt. The AES algorithm is currently the standard block cipher algorithm that has replaced the Data Encryption Standard (DES). It accepts 128 bits size block and the key size can be 128, 192, and 256 bits [41]. We use SHA-1 [42] as the hash function, which can generate 160 bits. The following statistics only consider the authentication stage.

**A. COMPUTATION COST**

First, we consider the computation cost of authentication stage. Table 4 summarizes the calculation amount of our work and other works. The numbers in the table represent the number of calculations of this function. Each column represents one device in an authentication.

In our protocol, the Gateway will perform two hash operations, one is to calculate CID\_A, another is to calculate SK, the same as Server. There will be 13 XOR operations for Gateway and 12 for Server in authentication stage. Although the number of our XOR operations is more than others, the calculation time of XOR operation is very fast and can be ignored.

If we do not consider the process of dynamic key generation using channel state information, the amount of computation required by our method is almost the same as [11].
TABLE 4. Computation cost compared with different works.

|                  | [10] Device | [10] HG | [27] FU | [27] FS | [11] Device | [11] Gateway | Our Gateway | Our Server |
|------------------|-------------|---------|---------|---------|-------------|--------------|-------------|------------|
| Hash operation   | 2           | 1       | 2       | 4       | 2           | 2            | 2           |
| MAC              | 1           | 1       | 1       | 3       | 3           | 1            | 1           |
| XOR operation    | 7           | 8       | 13      | 12      |             |              |             |
| Cryptosystem     | 1E+1D       | 1E+1D   | 1E+1D   | 2E+1D   | 1E+1D       | 1E+1D        |             |
| Key derivation   |             |         |         |         |             |              |             |
| function         | 1           | 1       |         |         |             |              |             |

TABLE 5. The storage cost compared with other works.

|                  | [10]Device | [10]Gateway | [27]FU | [27]FS | [11]Device | [11]Gateway | Our Gateway | Our Server |
|------------------|------------|-------------|--------|--------|------------|--------------|-------------|------------|
| Identity         | 2          | 2           | 1      | 3      | 2          | 2            | 2           | 2          |
| Key/Number       | 3          | 2           | 2      | 3      | 2          | 2            | 3           | 3          |
| Hash result      | 1          |             |         |        |            |              |             | 1          |

Therefore, the protocol is lightweight on both sides of the authentication.

It is worth noting that the device side of [11] is equivalent to our gateway, that is, the lower level device; the gateway side of [11] is equivalent to our server, that is, the superior level device. The above overhead does not include the process of dynamic key generation using channel state information. The dynamic key generation is usually low cost, as it only requires non-complex operations with only a change to the drivers [5].

B. COMMUNICATION COST

Next, we count the cost of communication in the authentication stage. Here we compare the overhead by counting the types of communication data, not the binary bits. For example, when a message in a communication is generated by hash functions, the length of the message is one hash; if the message is encrypted by AES, the length of the message is an AES length. We ignore the detailed length because it depends on the settings. The amount of data sent by the subordinate device to the superior device during certification is shown in Table 6.

In our protocol, the subordinate device needs to send $C_{ID_A}$ (a hash result), $C_1$ (a message encrypted with symmetric key) and a timestamp $T_1$.

The amount of data received by superior device from subordinate device is shown in Table 7. It can be seen that we need very little traffic in the authentication process.

We also calculate the total communication cost. Since these works do not specify the specific length of each variable, we need to make some assumptions. We assume that the length of number, Hash, MAC, HMAC results are 16 bytes. The length of identity and timestamp are 4 bytes. The length of encrypted data is the same as the length of decrypted data. The length of other shorter variables is 1 byte. The total communication cost of different works are shown in the Figure 8.

The overhead also does not include the process of dynamic key generation using channel state information. Because the measurement of channel state can take place during the whole process of communication, which does not generate additional traffic. There will be some communication requirements in the consistency check after sampling in dynamic key generation, which mainly depends on the length of the check code.

C. STORAGE COST

We also compare our storage cost with other works, as shown in the Table 5. We divide the key and random number into one category, because the length of this kind of information is not
clearly indicated in other works, and their length is relatively similar.

The numbers in Table 5 means the number of this kind of information saved in the devices after one device registering on another device. We can see that there is little difference in storage requirements between different works.

VIII. CONCLUSION
In this paper, we propose a protocol that can authenticate devices and dynamically generate session key in FC architecture. Our protocol uses hash, symmetric key and physical layer dynamic key generation technique to achieve that goal, which is not only anonymous and lightweight, but also can be applied for each device in the whole FC architecture, to make the edge nodes, fog nodes and cloud server authenticate each other and make data transmission more secure. We also analyze the performance and security of our method, which proves that our method is lightweight and secure.

APPENDIX
BAN LOGICAL PROOF
In this section, we use BAN logic [40] to prove the security of our protocol. BAN logic is widely used in reasoning and proving the security of protocol. It uses the existing rules and logic symbols to reasoning and prove the security goals. The process is mainly divided into the following four steps: (1) transforming the informal protocol into logical symbols for representation; (2) finding out the initial assumptions in the protocol and converting them into logical symbols for representation; (3) listing the security objectives to be proved; (4) using the specific process, initial assumptions and existing rules of the protocol, proving whether the protocol can achieve the security goals.

A. BAN LOGIC NOTATIONS
In BAN logic, we first define some symbols, as shown below.

\( P, Q, R \): Principal entities.

\( X, Y \): Message.

\( K \): Secret key.

\( P \models X \): The principal \( P \) believes the message \( X \).

\( P \models X \): \( P \) once said \( X \), i.e., at some time the principal \( P \) sent a message including \( X \).

\( P \triangleleft X \): \( P \) sees \( X \), i.e., suppose a device once sent a message containing \( X \) to \( P \), then \( P \) can read and repeat \( X \) (after some decryption operations).

\( P \models X \): \( P \) has control over \( X \), i.e., principal \( P \) is authoritative about \( X \) and should be trusted.

\#(\( X \)): Fresh(\( X \)), i.e., \( X \) has not been sent recently during protocol execution.

\( \langle M \rangle_K \): Message \( M \) is encrypted with secret key \( K \).

\( \langle M \rangle_N \): \( M \) is combined with the secret parameter \( N \).

\( P \triangleleft Q \): \( P \) and \( Q \) share a secret \( K \) which will not be disclosed.

\( P \models M \): The principal \( X \) and \( Y \) have message(\( M \)) that contains the secret parameters.

B. LOGIC RULES
In the following rules, we assume that if the conditions above the horizontal line are met, then we can infer the following results under the line. We use these rules to perform logical derivation.

1) Message-meaning rule

\[ P \models X \triangleleft P, P \models \langle M \rangle_K \]
\[ P \models Q \models M \]

2) Nonce-verification rule

\[ P \models \#(\langle M \rangle), P \models Q \triangleleft M \]
\[ P \models Q \models M \]

3) Control rule

\[ P \models Q \models M, P \models Q \models M \]
\[ P \models M \]

4) Seeing Rules

\[ P \models \langle M \rangle_N, P \models \langle M, N \rangle \]
\[ P \models M \]

5) Fresh rule

\[ P \models \#(\langle M \rangle) \]
\[ P \models \#(M, N) \]
C. MESSAGE IDEALIZATION

Next, we idealize the messages delivered by both parties of the protocol (Gateway A is represented by GA, and Server is represented by S). First let’s look at the formal presentation of the notations in authentication phase.

M1:
\[ S \leftarrow CID_A (= h (id_A \oplus R_A \oplus key_{dy} \oplus n)) \]
\[ S \leftarrow C_1 (= E_{TK} (id_S \parallel n \parallel R'_A \parallel R_A \parallel T_1 \parallel p)) \]
\[ S \leftarrow T_1 \]
M2:
\[ GA \leftarrow V_1 \]
\[ GA \leftarrow T_3 \]

D. ASSUMPTIONS

In our protocol, one principle believes that: (a) it has shared the key and other secret information with another principle, i.e., \( id_A, id_S, R_A, R_S, key_{dy}, n \); (b) the random numbers and timestamps generated by a principle each time are fresh; (c) the legal principle can control the variables that can be changed. The initial assumptions are as follows:

For Server:

A1: \[ S \equiv GA \xrightarrow{CID_A,C_1,T_1} S \]
A2: \[ S \equiv GA \xrightarrow{TK} S \]
A3: \[ S \equiv \left( GA \Rightarrow GA \xrightarrow{TK} S \right) \]
A4: \[ S \equiv \left( R'_S \right) \]
A5: \[ S \equiv \left( T_1 \right) \]
A6: \[ S \equiv \left( GA \Rightarrow id_A \right) \]

For Gateway A:

A7: \[ GA \equiv S \xrightarrow{V_1,T_3} GA \]
A8: \[ GA \equiv S \xrightarrow{TK} GA \]
A9: \[ GA \equiv \left( S \Rightarrow S \xrightarrow{TK} GA \right) \]
A10: \[ GA \equiv \left( R'_A \right) \]
A11: \[ GA \equiv \left( T_3 \right) \]
A12: \[ GA \equiv \left( S \Rightarrow id_S \right) \]

E. EXPECTED GOALS

In our protocol, we expect that Gateway A and Server can authenticate each other and communicate with the same session key \( TK \). They can verify that the newly generated session key \( SK \) and random number \( R'_A, R'_S \) are fresh. The details are as follows:

G1: \[ GA \equiv S \equiv GA \xrightarrow{TK} S \]
G2: \[ S \equiv GA \xrightarrow{TK} GA \]
G3: \[ GA \equiv \left( R'_S, \left( SK \right) \right) \]
G4: \[ S \equiv \left( R'_A, \left( SK \right) \right) \]

F. LOGIC VERIFICATION

Prove G1: \[ GA \equiv S \equiv GA \xrightarrow{TK} S \]

According to A7: \[ GA \equiv S \xrightarrow{V_1,T_3} GA \], we know that GA believes \( S, V_1, T_3 \) contains confidential variables, and can verify the accuracy of the data.

According to M2: \[ GA \leftarrow V_1, GA \leftarrow T_3 \], we know that GA receives \( V_1 \) and \( T_3 \). Due to \( T_3^* = T_3, id_S^* = id_S, n^* = 0, n^* = n + 1 \), we obtain that:

\[ GA \leftarrow \{ V_1 \}_TK, GA \leftarrow \{ T_3 \} \]

By applying message-meaning rules and A11, we can obtain:

\[ GA \equiv \left( S \Rightarrow \left| V_1 \right|_TK \right) \]
\[ GA \equiv \left( S \Rightarrow \{ T_3 \} \right) \]

If GA trust \( S, \{ T_3 \}, R_A, R_S \), then it can also trust \( TK \). According to A8, we can know that:

\[ GA \equiv \left( S \Rightarrow GA \left( = R_S \oplus R_A \oplus key_{dy} \oplus id_A \right) \right) \equiv S \]

The same process of proof can prove goal G2.

Prove G3: \[ GA \equiv \# \left( R'_S, SK \right) \]

According to M2: \[ GA \leftarrow V_1, V_1 \] contains \( n', T_3, key_{dy} \oplus R'_S \oplus R'_S \oplus R'_A \), GA can get \( R'_S \) from known information, and then check whether it is correct. According to G1, fresh rule and A4 we can get:

\[ GA \equiv \left( R'_S \right) \]

Due to \( (SK)_{\#R'_S} \), we can get:

\[ GA \equiv \# \left( SK \right) \]

The same process of proof can prove goal G4.

REFERENCES

[1] P. Zhang, M. Zhou, and G. Fortino, “Security and trust issues in fog computing: A survey,” Future Gener. Comput. Syst., vol. 88, pp. 16–27, Nov. 2018.
[2] S. Yi, Z. Qin, and Q. Li, “Security and privacy issues of fog computing: A survey,” in Proc. Int. Conf. Wireless Algorithms, Syst., Appl. Cham, Switzerland: Springer, 2015, pp. 685–695.
[3] M. Mukherjee, R. Matam, L. Shu, L. Maglaras, M. A. Ferrag, N. Choudhury, and V. Kumar, “Security and privacy in fog computing: Challenges,” IEEE Access, vol. 5, pp. 19293–19304, 2017.
[4] J. Zhang, Q. Li, X. Wang, B. Feng, and D. Guo, “Towards fast and lightweight spam account detection in mobile social networks through fog computing,” Peer-Peer Netw. Appl., vol. 11, no. 4, pp. 778–792, Jul. 2018.
[5] J. Zhang, T. Q. Duong, A. Marshall, and R. Woods, “Key generation from wireless channels: A review,” IEEE Access, vol. 4, pp. 614–626, 2016.
[6] H. Liu, Y. Wang, J. Yang, and Y. Chen, “Fast and practical secret key extraction by exploiting channel response,” in Proc. IEEE INFOCOM, Apr. 2013, pp. 3048–3056.
[7] K. Chen, B. B. Natarajan, and S. Shattil, “Secret key generation rate with power allocation in relay-based LTE—A networks,” IEEE Trans. Inf. Forensics Security, vol. 10, no. 11, pp. 2424–2434, Nov. 2015.
[8] F. Marino, E. Paolini, and M. Chiani, “Secret key extraction from a UWB channel: Analysis in a real environment,” in Proc. IEEE Int. Conf. Ultra-WideBand (ICUWB), Sep. 2014, pp. 80–85.
[9] B. Vaidya, D. Makrakis, and H. T. Mouftah, “Device authentication mechanism for smart energy home area networks,” in Proc. IEEE Int. Conf. Consum. Electron. (ICCE), Jan. 2011, pp. 787–788.
[10] P. Kumar, A. Gurtov, J. Iinatti, and P. H. Ha, “Anonymous secure framework in connected smart home environments,” in Proc. IEEE Sensors J., vol. 16, no. 1, pp. 254–264, Jan. 2016.
[11] K. Han, J. Kim, T. Shon, and D. Ko, “A novel secure key paring protocol for RF4CE ubiquitous smart home systems,” Pers. Ubiquitous Comput., vol. 17, no. 5, pp. 945–949, Jun. 2013.
[13] M. Burrough and J. Gill, “Smart thermostat security: Turning up the heat,” Univ. Illinois Urbana-Champaign, Urbana, IL, USA, Tech. Rep. UIUC CS523, 2015.

[14] V. Odelu, A. Kumar Das, M. Wazid, and M. Conti, “Provably secure authenticated key agreement scheme for smart grid,” IEEE Trans. Smart Grid, vol. 9, no. 3, pp. 1900–1910, May 2018.

[15] J. Yuan and S. Yu, “Efficient privacy-preserving biometric identification in cloud computing,” in Proc. IEEE INFOCOM, Apr. 2013, pp. 2652–2660.

[16] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich, “SCFI—A system for secure face identification,” in Proc. IEEE Symp. Secur. Privacy, Berkeley/Oakland, CA, USA, May 2010, pp. 239–254.

[17] M. Haghighat, S. Zonouz, and M. Abdel-Mottaleb, “CloudID: Trustworthy cloud-based and cross-enterprise biometric identification,” Expert Syst. Appl., vol. 42, no. 21, pp. 7905–7916, Nov. 2015.

[18] A. S. Bommagani, M. C. Valenti, and A. Ross, “A framework for secure cloud-empowered mobile biometrics,” in Proc. IEEE MILCOM. Commun. Conf., Oct. 2014, pp. 255–261.

[19] P. Hu, H. Ning, T. Qiu, H. Song, Y. Wang, and X. Yao, “Security and privacy preservation scheme of face identification and resolution framework using fog computing in Internet of Things,” IEEE Internet Things J., vol. 4, no. 5, pp. 1143–1155, Oct. 2017.

[20] J. Zhu and J. Ma, “A new authentication scheme with anonymity for VOLUME 8, 2020.

[21] M. Haghighat, S. Zonouz, and M. Abdel-Mottaleb, “CloudID: Trustworthy cloud-based and cross-enterprise biometric identification,” Expert Syst. Appl., vol. 42, no. 21, pp. 7905–7916, Nov. 2015.

[22] A. S. Bommagani, M. C. Valenti, and A. Ross, “A framework for secure cloud-empowered mobile biometrics,” in Proc. IEEE MILCOM. Commun. Conf., Oct. 2014, pp. 255–261.

[23] B. Huang, X. Cheng, Y. Cao, and L. Zhang, “Lightweight hardware based secure authentication scheme for fog computing,” in Proc. IEEE/ACM Symp. Edge Comput. (SEC), Oct. 2018, pp. 433–439.

[24] M. H. Ibrahim, “Octopus: An edge-fog mutual authentication scheme,” Int. J. Netw. Secur., vol. 18, no. 6, pp. 1089–1101, Nov. 2016.

[25] A. Ben Amor, M. Abid, and A. Meddeb, “A privacy-preserving authentication scheme in an edge-fog environment,” in Proc. IEEE/ACCS 14th Int. Conf. Comput. Syst. Appl. (AICCSA), Oct. 2017, pp. 1225–1231.

[26] F. Wang, Y. Xu, L. Zha, X. Du, and M. Guizani, “LAMANCO: A lightweight anonymous mutual authentication scheme for N-times computing offloading in IoT,” IEEE Internet Things J., vol. 6, no. 3, pp. 4462–4471, Jun. 2019.

[27] A. Albakri, M. Maddumala, and L. Harn, “Hierarchical polynomial-based key management scheme in fog computing,” in Proc. 17th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun./12th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE), Aug. 2018, pp. 1593–1597.

[28] G. Rahman and C. C. Wen, “Mutual authentication security scheme in fog computing,” Int. J. Adv. Comput. Sci. Appl., vol. 10, no. 11, pp. 443–451, Jan. 2019.

[29] A. Albakri, L. Harn, and S. Song, “Hierarchical key management scheme with probabilistic security in a wireless sensor network (WSN),” Secur. Commun. Netw., vol. 2019, pp. 1–11, Jul. 2019.

[30] S. Guillet, B. Bouchard, and A. Bouzouane, “Correct by construction security approach to design fault tolerant smart homes for disabled people,” Procedia Comput. Sci., vol. 21, pp. 257–264, Oct. 2013.

[31] N. Patwari, J. Croft, S. Jana, and S. K. Kasera, “High-rate uncorrelated bit extraction for shared secret key generation from channel measurements,” IEEE Trans. Mobile Comput., vol. 9, no. 1, pp. 17–30, Jan. 2010.

[32] W. Xi, X.-Y. Li, C. Qian, J. Han, S. Tang, J. Zhao, and K. Zhao, “KEEP: Fast secret key extraction protocol for D2D communication,” in Proc. IEEE 22nd Int. Symp. Qual. Service (IWQoS), May 2014, pp. 350–359.

[33] R. Guillaume, F. Winzer, A. Czywik, C. T. Zenger, and C. Paar, “Bringing PHY-based key generation into the field: An evaluation for practical scenarios,” in Proc. IEEE 82nd Veh. Technol. Conf. (VTC-Full), Sep. 2015, pp. 1–5.

[34] H. Liu, J. Yang, Y. Wang, and Y. Chen, “Collaborative secret key extraction leveraging received signal strength in mobile wireless networks,” in Proc. IEEE INFOCOM, Mar. 2012, pp. 927–935.

[35] S. N. Premnath, P. L. Gowda, S. K. Kasera, N. Patwari, and R. Ricci, “Secret key extraction using Bluetooth wireless signal strength measurements,” in Proc. 11th Annu. IEEE Int. Conf. Sens., Commun., Netw. (SECON), Jun. 2014, pp. 293–301.

[36] R. Wilson, D. Tse, and R. A. Scholtz, “Channel identification: Secret sharing using reciprocity in ultrawideband channels,” IEEE Trans. Inf. Forensics Security, vol. 2, no. 3, pp. 364–375, Sep. 2007.

[37] L. Jiao, N. Wang, P. Wang, A. Alipour-Fanadi, J. Tang, and K. Zeng, “Physical layer key generation in 5G wireless networks,” IEEE Wireless Commun., vol. 26, no. 5, pp. 48–54, Oct. 2019.

[38] L. Jiao, J. Tang, and K. Zeng, “Physical layer key generation using virtual AoA and AoD of mmWave massive MIMO channel,” in Proc. IEEE Conf. Commun. Netw. Secur. (CNS), May 2018, pp. 1–9.

[39] D. Dolev and A. Yao, “On the security of public key protocols,” IEEE Trans. Inf. Theory, vol. 29, no. 2, pp. 198–208, Mar. 1983.

[40] A. Armanto, “Deliverable d2. 1: The high level protocol specification language,” AVISPA Project, Tech. Rep. IST-2001-39252, 2003. [Online]. Available: http://www.avispa-project.org/

[41] M. Burrows, M. Abadi, and R. M. Needham, “A logic of authentication,” Proc. Roy. Soc. London A, Math., Phys. Eng. Sci., vol. 426, no. 1871, pp. 233–271, 1989.

[42] R. Velayutham and J. M. Suganya, “Security authentication through AES and fine-grained distributed data access control using clustering in wireless sensor networks,” in Proc. 3rd Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT), Jul. 2012, pp. 1–6.

[43] D. Eastlake and P. Jones, “Us secure hash algorithm 1 (sha1),” Internet Soc., Reston, VA, USA, Tech. Rep. RFC 3174, 2001.