IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received May 13, 2019, accepted June 22, 2019, date of publication June 27, 2019, date of current version July 17, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2925403

A Provable Semi-Outsourcing Privacy Preserving Scheme for Data Transmission From IoT Devices

XIAOSHUAI ZHANG, (Student Member, IEEE), CHAO LIU, STEFAN POSLAD, (Member, IEEE), AND KOK KEONG CHAI, (Member, IEEE)
School of Electrical Engineering and Computer Science, Queen Mary University of London, London E1 4NS, U.K.

Corresponding author: Xiaoshuai Zhang (xiaoshuai.zhang@qmul.ac.uk)

This work was supported in part by a Ph.D. Scholarship funded jointly by the China Scholarship Council and the Queen Mary University of London.

ABSTRACT A semi-outsourcing privacy-preserving scheme is proposed in this paper for the IoT data collection named semi-outsourcing privacy-preserving (SOPP), which supports delegated identity authentication for the IoT devices without revealing the transmitted data. Compared with other schemes that implement the authentication based upon using trusted cloud services, the design of our scheme SOPP can achieve the delegated authentication on untrusted public clouds while providing privacy-preserving data transmission. Meanwhile, the implemented one-way authentication can reduce the communication cost for the IoT devices (especially for the low-resource ones) to prolong their battery life. The performance of the SOPP scheme is demonstrated for its use in the resource-constrained IoT devices and compared with a benchmark trusted cloud scheme including one based upon certificates and an interactive (two-way) authentication scheme.

INDEX TERMS Privacy preservation, Internet of Things, cloud computing, communication security, public key encryption.

I. INTRODUCTION

Cloud computing has been widely applied as an access infrastructure to serve IoT devices for uploading and processing their collected data [1]. In cloud computing, the clouds can work as a gateway (or an agent) to verify IoT devices' identities and to collect (or aggregate) the data from IoT devices. In this way, an increasing number of companies are deploying IoT devices and clouds to gather health, public utility, and transportation use information [2], [3]. Stake-holders are raising concerns about the data privacy design requirements [4] and that the data security needs to adapt to low ICT resource IoT devices that have limited computing and energy resources [5], [6]. To satisfy the demands of privacy preservation for IoT devices, trusted infrastructures of cloud computing that preserve privacy are essential to be built in large-scale IoT networks. The privacy-preservation challenge or requirement is as follows. A user's private stored information needs to be kept confidential from the cloud infrastructure provider and other users that have access to that (multi-tenancy) cloud infrastructure.

Many small and medium-sized enterprises and organizations choose to deploy their services on untrusted public clouds. The reason is that the cost of building a complete private trusted cloud computing that contains enough computing resource and consummate security functions to support a range of security operations in large-scale IoT networks is not affordable for them. However, deploying services on untrusted public clouds raises concerns about how to ensure the data privacy and security during the data transmission process [7]. In this way, a novel secure data privacy scheme for IoT data transmission on untrusted clouds is proposed to accommodate this scenario.

Generally, there are two steps in the data collection (transmission) from IoT devices to clouds: authentication and data transmission. In the authentication, the clouds should verify the identity of the IoT device. If the IoT device's identity is valid, the clouds receive the uploaded data from the IoT device then transmit the data to the data collector (control center). The trusted private clouds have the permissions to access all the data in the described two steps authentication and data transmission, i.e. the clouds can verify the identity of the IoT device, as well as check the integrity and the correctness by decrypting the transported data from the IoT device since the clouds are trusted and reliable [8]. However, the

The associate editor coordinating the review of this manuscript and approving it for publication was Md. Arafatur Rahman.

plain identity of the IoT device and plaintext data should not be revealed to the untrusted public clouds to avoid personal data leakage because the public clouds are regarded as a multi-purpose and multi-user environment without reliable measures to safeguard users' data privacy [9].

Most of existing research focuses on data privacy and security analysis in trusted clouds [8], [10]. In [11], a data management framework is proposed to convert, aggregate, clean, and filter data on trusted clouds. However, the aforementioned framework uses centralized digitalized certificates to verify users' identities so that interactive dual authentication can be achieved. It means the IoT device has to communicate with the certificate authority and the cloud several times in order to authenticate the connections and to transport the data from the IoT device to the data collector. However, this increases the computing consumption and network throughput, whilst limiting their use for resource-constrained IoT devices [12]. In [13], trusted clouds and PKI are applied to IoT m-Health network, and clouds and act as an agent to aggregate and transmit encrypted data from sensors. However, they do not consider the resource limitations of weaker resource IoT devices and there are no related experiments or any discussion about the cost of computation and wireless transmission in this paper. Reference [14] applies trusted clouds to implement a similar agent structure in the service layer named *CMfg* that can manage middleware and computational resources in the clouds, but there is no security consideration (e.g. access control, reliability and confidentiality) in the design of *CMfg* to protect IoT devices' data in communications.

Research on addressing a stricter data privacy requirement is limited for untrusted clouds. A homomorphic encryption scheme for executing private queries on untrusted clouds is presented in [15]. In [16], authors use ring signature and computationally complex homomorphic authenticators to provide a privacy sharing scheme for untrusted clouds. However, the bilinear pairing used in homomorphism-related cryptographic operations costs much more computational resources when compared with the elliptic curve cryptography (ECC) [17]. In summary, the aforementioned data security schemes are cloud-centric and require intensive ICT resources on public clouds, which are not suitable to be used as a baseline to compare performance. Also, public cloud schemes are more computationally complex compared with a private cloud scheme. So the practical applicability of those public cloud schemes in resource-constrained IoT devices such as low CPU performance and limited power supply are restricted.

Compared with existing schemes for trusted clouds in [12]–[14], the novel security and privacy requirements for SOPP data exchange are:

1) SOPP can be applied to untrusted public clouds without exposing plaintext private data to the clouds.
2) The authentication process is delegated to public clouds to block invalid access, but the data decryption and integrity validation are implemented at the data center (semi-outsourcing).
3) According to the comparison with the interactive authentication scheme from [11] and the homomorphic methods from [15], [16], our proposed SOPP scheme's design is based upon ECC and is implemented as a one-way (non-interactive) authentication strategy between untrusted public clouds and IoT devices to decrease the throughput for IoT devices in order to achieve a longer battery duration [18].

Compared with our original work [19], the improvements of this paper are:

1) A systematic security model is constructed to guide the scheme design and the elaboration of the proposed scheme's security in terms of different security requirements (authentication, confidentiality, and integrity).
2) A theoretical proof is illustrated formally to show that SOPP can resist the one-wayness under a chosen ciphertext attack (OW-CCA) in the random oracle model.
3) More detailed results from the comparative experiments are demonstrated in the section IV.B to express the better performance of SOPP in terms of time efficiency and network throughput.
4) The proposed scheme SOPP is reconstructed carefully with the amended parts of public parameters generation and data encryption to ensure the cryptographic correctness.

The structure of this paper is organized as follows. In Section II, the preliminaries for scheme construction are presented, such as the definition of the Elliptic Curve Diffie-Hellman (ECDH) assumption. In Section III, the model and definitions used to illustrate our proposed scheme SOPP are introduced. Then the proposed SOPP scheme and performance of our scheme simulation are included in Section IV. Security analysis is discussed in Section V and the conclusion is presented in Section VI.

II. PRELIMINARIES

A. ELLIPTIC CURVE DIFFIE-HELLMAN (ECDH) ASSUMPTION

ECDH assumption [20] is a computational problem on elliptic curves that can be used in information exchange to ensure data confidentiality in public key cryptography. Let $E_p(a, b)$ be a cryptographic secure elliptic curve. Any probabilistic polynomial-time algorithm $\mathcal{A}$ computes uvP with its advantage: $Adv^{ECDH}_{\mathcal{A},E_p(a,b)} = Pr[c = uvP|u, v{\in}_R\mathbb{Z}^*_p, c = \mathcal{A}(P, uP, vP)]$, where, P is a point on $E_p(a, b)$ and $u, v{\in}_R\mathbb{Z}^*_p$. The ECDH assumption can hold if for any probabilistic polynomial-time algorithm $\mathcal{A}$, its advantage $Adv^{ECDH}_{\mathcal{A},E_p(a,b)}$ is negligible.

III. MODEL AND DEFINITION

A. SCHEME MODEL

In this section, we present our proposed scheme model and the comparison with existing baseline models. There are three types of entity in the system, which are the IoT devices, public (untrusted) clouds, and data center (also referred as control center in some scenarios such as smart grids) respectively. The system infrastructure is presented in Figure. 1.

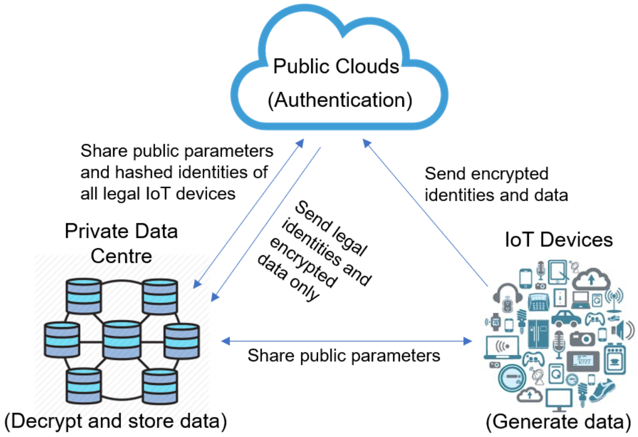
Public Clouds
(Authentication)
Share public parameters and hashed identities of all legal IoT devices
Send encrypted identities and data
Send legal identities and encrypted data only
Private Data Centre
IoT Devices
Share public parameters
(Decrypt and store data)
(Generate data)

FIGURE 1. **The infrastructure of proposed SOPP scheme model.**

Organisations and companies fully control and trust their own data center. However, there are some cases where the data center can be owned by or integrated into untrusted public clouds. In this paper, we assume that clouds are separate or independently managed, and the untrusted public cloud is only used for data gathering and exchange, it is not used for data management. Henceforth, it also implies an additional IoT security data requirement which requires avoiding private information leakage from the IoT device to the clouds and from the clouds to the data center.

The schematic work phases of the SOPP scheme model are described as follows. The first phase is used to broadcast the public security parameters to all the entities from the private data center, which share an identity list with the untrusted public clouds at the same time. Note that all the elements of the identity list are the IoT devices' hashed identities. Then, the IoT devices transport their hashed identities and encrypted data to the untrusted public in the next phase, when the data center collects data from the IoT devices. After receiving the hashed identities and the encrypted data, the public clouds can validate the identities of the data senders (the IoT devices) by querying the identities in the identity list. If the IoT devices' identities are valid, the untrusted public clouds will send the corresponding encrypted data from the identified IoT devices to the data center, otherwise, the data collection should be aborted. In the last phase, the received encrypted data are decrypted by the private data center, the data integrity is checked to avoid tampering, then if intact the plain data is saved for processing.

B. MODEL COMPARISON

Compared with the baseline model "data center - agents (trusted clouds) - IoT devices", the highlight of our scheme model is in obviates the need to use the trusted clouds to realize data privacy preservation. Since the cost of leasing public clouds is much less than the expense of building trusted private clouds, our scheme can ensure data privacy for IoT data transmission at a lower cost for the infrastructure construction.

Compared with another general model "data center - IoT devices", our model does not require us to deploy a large-scale data center to process data and satisfy security demands (e.g. authentication, validation, and cyber-attacks defense) since the untrusted public clouds are strong enough to assume a part of security work, i.e. cyber-attacks prevention (detection) and the authentication work can be delegated to untrusted public clouds. Meanwhile, applying our scheme can ensure data privacy when the IoT devices' data is transported via untrusted public clouds. Therefore, the companies and organizations only need to construct a small-scale data center for data validation and storage and lease the public clouds for access control without any private data leakage from IoT devices when deploying our scheme so that again the construction expense could be decreased.

C. SOPP SCHEME DEFINITION

The propose scheme SOPP consists of the following five phases:

1) *SETUP*(λ)

This algorithm uses the security parameter λ to generate the public parameter pp.

2) *KEYINITIALISE(pp)*

This phase is used to initialize all the required public keys (pk) and private keys (sk) based upon the public parameter pp for the next phases.

3) *ENCRYPT*(pp, M, ID, pk)

The algorithm encrypts the plaintext M and the IoT device's identity ID with the public parameter pp and the public key pk then outputs the ciphertext C.

4) *AUTHENTICATION*(pp, C, sk)

The public clouds decrypt the identity part of the ciphertext C with sk (shared by the data center) then transfer the data part $C' \subset C$ to the data center if a successful validation of the identity of the data sender occurs.

5) *DECRYPT*(pp, C', sk)

The data center decrypts the encrypted data C' with the private key sk to retrieve the sent data M then validate the integrity of M with this algorithm.

D. CORRECTNESS DEFINITION

The scheme is correct if for any $pp \leftarrow Setup(\lambda)$, and $(pk, sk) \leftarrow KeyInitialise(pp)$, the following conditions hold.

1. For any identity ID, the public clouds can always retrieve the identity $f(ID)$ via $Authentication(Encrypt(pp, M, ID, pk), sk)$, where f can be any secure one-way function.

2. For any plaintext M, $Decrypt(Encrypt(pp, M, ID, pk), sk)) = M$ always holds.

E. SECURITY DEFINITION

In this section, the adversary is defined at first to illustrate the definition of OW-CCA (i.e. one-wayness under a chosen ciphertext attack) security for SOPP.

1) ADVERSARY

Formally, the adversary defined for the security of our proposed scheme SOPP is:

- *Type-I adversary*: In the *Authentication* phase, the adversary cannot retrieve the plain message from the challenge ciphertext.

2) OW-CCA SECURITY

The definition of OW-CCA security model with the *Type-I adversary* for the *Authentication* phase in SOPP is as follows.

Game 1 $\mathcal{A}_1$ *is the given Type-I adversary, and the target device's index is* t $(1\leqslant t\leqslant n)$. *The game* 1 *between the challenger* $\mathcal{C}$ *and* $\mathcal{A}_1$ *is operated as follows:*

- *Setup*

$\mathcal{C}$ firstly generates the public parameter pp via running the algorithm *Setup*. Then, $\mathcal{C}$ generates n public and private key pairs (pk_i, sk_i) $(1\leqslant i\leqslant n)$ via running the algorithm *KeyGenerate*. The generated pp and all pk_i are given to the adversary $\mathcal{A}_1$.

- *Queries*

The following queries can be requested by $\mathcal{A}_1$ for polynomial times.

1. *Key retrieve query*(i): $\mathcal{C}$ responds with the private key sk_i.

2. *Authentication query*(i, C): $\mathcal{C}$ returns the trapdoor $\mathcal{T}_i$ to recover $f(ID)$.

3. *Decryption query*(i, C'): $\mathcal{C}$ decrypts C' with sk_i via running the algorithm $Decrypt(C', sk_i)$, and responds with the output message.

- *Challenge*

$\mathcal{C}$ picks a message M^* randomly, then computes the challenge ciphertext $C^* = Encrypt(M^*, pk_t)$ and finally responds the challenge ciphertext C^*.

- *Constraints*

(1) The target device's index t is not allowed to appear in the above *Key retrieve query*.

(2) The target device's index t and the challenge ciphertext C^* is not allowed to appear in the above *Decryption query*.

- *Guess*

$\mathcal{A}_1$ can win the game if its output $M^{*\prime}$ satisfies the condition $M^{*\prime} = M^*$.

Now, the advantage of $\mathcal{A}_1$ could be defined as:

$$Adv_{\mathcal{A}_1}^{OW-CCA}(\lambda) = Pr[M^* = M^{*\prime}].$$

Definition 1 (OW-CCA Security): The proposed scheme SOPP is OW-CCA secure if the advantage $Adv_{\mathcal{A}_1}^{OW-CCA}(\lambda)$ *of any probabilistic polynomial-time adversary* $\mathcal{A}_1$ *is negligible.*

3) AUTHENTICATION

The public clouds can check the identity part $C'\subset C$ to block the data transmission without the valid identity after receiving the encrypted data C from the IoT device.

4) CONFIDENTIALITY

The confidentiality of our scheme is to ensure that the untrusted public clouds cannot decrypt the plaintext M from the encrypted data C in the *Authentication* phase. To be specific, $\forall M \in \{0, 1\}^*, C = f(M)$, any probabilistic polynomial-time algorithm $\mathcal{B}$ computes M with its advantage

$$Adv_{\mathcal{B}} = Pr[c = M|c = \mathcal{B}(pp, C, sk)] < \varepsilon,$$

where M is plaintext data, C is encrypted data by the algorithm f that transmitted from the IoT device via the public clouds, pp and sk denote the public parameters and the known private key respectively in the phase *Authentication*, and ε represents a negligible probability.

5) INTEGRITY

After decrypting the encrypted data C' received from the untrusted public clouds, the data center can check the integrity of the decrypted data M to avoid invalid data manipulation by the attackers in the transmission from the untrusted public clouds to the data center.

IV. PROPOSED SCHEME

In this section, we present our proposed new SOPP scheme and demonstrate its validity. The experimental validations are analyzed by comparing the time cost and communication cost. Note that we use non-interactive (one-way) authentication scheme between IoT devices and public clouds (untrusted), so the two-way authentication is superfluous, which results in reducing the commutation cost between low-resource IoT devices and public clouds.

A. THE PROPOSED SOPP SCHEME

1) SETUP(λ)

This algorithm uses the security parameter λ to generate the public parameters pp in five steps.

1. Pick a cryptographic secure elliptic curve group $\mathbb{G}$ with a base point G on the curve, where the order of $\mathbb{G}$ is p.

2. Select three cryptographic secure hash functions: $H_1 : \{0, 1\}^*\rightarrow \{0, 1\}^\lambda$, $H_2 : \mathbb{G}\rightarrow \{0, 1\}^{2\lambda}$ and $H_3 : \mathbb{G}^3 \times \{0, 1\}^{2\lambda}\rightarrow \{0, 1\}^\lambda$.

3. Choose a symmetric encrypting and decrypting algorithm. Note that the selected algorithm to describe our SOPP scheme is the Advanced Encryption Standard (AES) [20].

4. The data center calculates the hashed identity $H_{ID} = H_1(ID)$ for all IoT devices in the network to create an identity list including all IoT devices' hashed identities as the elements, $List - H_{ID}$, then shares $List - H_{ID}$ with the untrusted public clouds.

5. Output $pp = (\mathbb{G}, p, G, H_1, H_2, H_3, AES)$

2) *KEYINITIALISE(pp)*

This algorithm randomly picks two numbers $a, b \in_R \mathbb{Z}_p^*$ and calculates the key pair:

$$(pk, sk_1, sk_2) = ((A = aG, B = bG), (a), (b))$$

The public keys A and B can be broadcast in the whole network. However, the private key $sk_2 = b$ is only shared with the public clouds, meanwhile, the private key, $sk_1 = a$, is kept such that it is known only by the data center secretly.

3) *ENCRYPT(pp, M, ID, pk)*

For the given sent data $M \in \{0, 1\}^\lambda$ and the identity of the IoT device $ID \in \{0, 1\}^\lambda$, the algorithm outputs the encrypted ciphertext $C = (C_1, C_2, C_3, C_4)$ via the following steps:

1. Compute $H_M = H_1(M), H_{ID} = H_1(ID)$.
2. Use AES to encrypt data M with the key H_{ID} then get the ciphertext $AES_{H_{ID}}(M)$. For decrypting $AES_{H_{ID}}(M)$ to recover the plaintext M, $AES'_{H_{ID}}$ is defined as the decryption process: $M = AES'_{H_{ID}}(AES_{H_{ID}}(M))$.
3. Pick two random numbers $r_1, r_2 \in_R \mathbb{Z}_p^*$ then compute

$$\begin{aligned} C_1 &= r_1 G \\ C_2 &= r_2 G \\ C_3 &= AES_{H_{ID}}(H_M || M) \oplus H_2(r_1 A) \\ C_4 &= H_{ID} \oplus H_3(r_2 B, C_1, C_2, C_3). \end{aligned}$$

4) *AUTHENTICATION(pp, C, sk$_2$)*

The public clouds receive the ciphertext C and computes $C_4 \oplus H_3(bC_2, C_1, C_2, C_3)$ to recover H_{ID}. Then, the public clouds check if H_{ID} belongs to the hashed identity list $List - H_{ID}$. If $H_{ID} \in List - H_{ID}$, the public clouds allow the transmission to send the ciphertext $C' = (H_{ID}, C_1, C_2, C_3)$ to the data center; if $H_{ID} \notin List - H_{ID}$, the public clouds deny the request for data transmission.

5) *DECRYPT(pp, C', sk$_1$)*

The data center can execute the next steps to retrieve the plaintext M from the received ciphertext $C' = (H_{ID}, C_1, C_2, C_3)$:

1. Recover $AES_{H_{ID}}(H_M || M)$ via computing

$$C_3 \oplus H_2(aC_1).$$

2. Decrypt $AES_{H_{ID}}(H_M || M)$ with the key H_{ID}

$$H_M || M = AES'_{H_{ID}}(AES_{H_{ID}}(H_M || M)).$$

3. If $H_1(M) = H_M$ holds, this algorithm outputs M; otherwise, it outputs $\perp$.

The entire workflow of this proposed scheme is depicted in the Figure. 2.

B. PERFORMANCE

The performance of the proposed scheme SOPP is compared with the scheme in [11] used as a baseline in terms of two aspects: time efficiency and network throughput. To be specific, the time consumption of the data transportation over Wi-Fi and the local computation in the IoT device is compared. Then, the network throughput including the quantity and the size of the transmitted packages in the two schemes will be evaluated. To build up the experiments, the low-resource IoT hub, Raspberry Pi 2 [21], is selected as the IoT device. Meanwhile, a conventional laptop with an Intel processor (3.30GHz) is used to perform as an node in the public clouds.

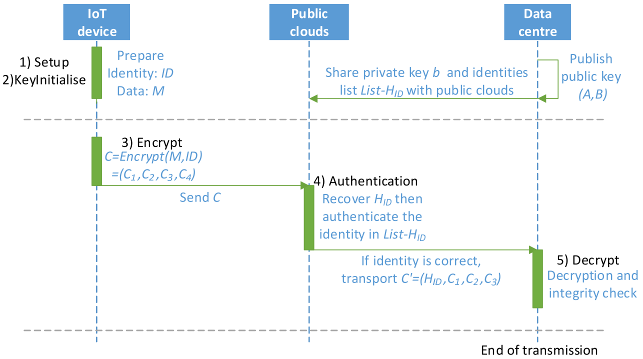
IoT device
Public clouds
Data centre
1) Setup
2)KeyInitialise
Prepare
Identity: *ID*
Data: *M*
Share private key *b* and identities list *List-H$_{ID}$* with public clouds
Publish public key *(A,B)*
3) Encrypt
C=Encrypt(M,ID)
=(C$_1$,C$_2$,C$_3$,C$_4$)
Send *C*
4) Authentication
Recover *H$_{ID}$* then authenticate the identity in *List-H$_{ID}$*
If identity is correct, transport *C'=(H$_{ID}$,C$_1$,C$_2$,C$_3$)*
5) Decrypt
Decryption and integrity check
End of transmission

FIGURE 2. The workflow of the proposed scheme SOPP.

We implement SOPP and the scheme in [11] based upon MIRACL [22], a cryptography SDK that can support all the required operations on elliptic curves and provide the needed hash functions and cryptographic algorithms. All the secure parameters and implemented experiments use an equivalent cryptographic security level (128-bit) [23] for both two schemes. Then the evaluation of the transmitted packages' quantity and size is implemented based upon TCP socket communication. The transmitted data size is set to 128 Bytes, and the used certificate size is 1024 Bytes (1 KB). In the experiments, we choose AES-256 for data encryption and decryption with SHA-256 as the hash algorithm. For public key operations, the elliptic curve we use is *secp160*.

1) *TIME EFFICIENCY COMPARISON*

For comparison of the time efficiency, the experiment is executed 5 rounds with 10 times in each round. The average time cost (for each round) of transporting data over Wi-Fi is shown in Figure. 3. Compared with the time cost of the scheme in [11], the reduction of the time cost in SOPP is about 75% on average. Meanwhile, the average total time cost including the time consumption of both local computation and data transmission is depicted in the Figure. 4. Although the time cost of local computation in SOPP is twice that of the scheme in [11], the total time cost of SOPP is around 60% less than that of the scheme in [11] because the data size required for authentication in our scheme SOPP is much smaller to lead to the much lower time cost for data transmission over Wi-Fi when compared with [11].

2) *COMMUNICATION THROUGHPUT COMPARISON*

In this experiment, the transmitted data size is set to be 256 bytes and 512 bytes respectively to obtain the average

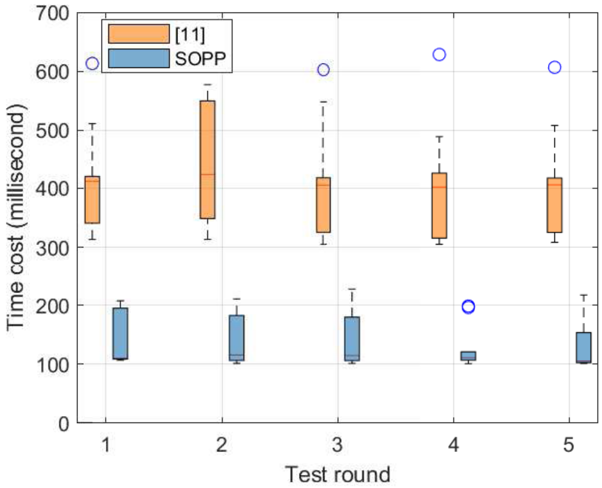

FIGURE 3. The comparison of time cost for data transmission over Wi-Fi.

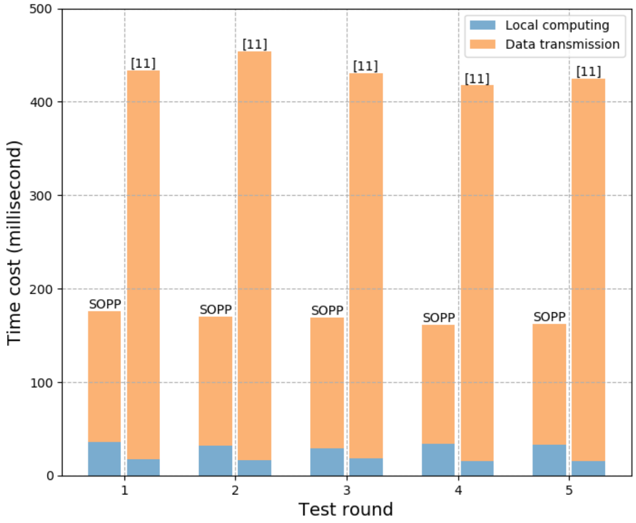

FIGURE 4. The comparison of the total time cost for local computing and data transmission.

result (over 100 times), which is determined by monitoring the network throughput. For each data size, the average size and quantity of the sent packages by the IoT device (Raspberry Pi 2) for data transmission are summarized in Table. 1. Compared with the package size and quantity of the scheme in [11], the size and the quantity of the sent packages are reduced by around 45% and 60% respectively because of the less required data used for authentication in SOPP.

To summarize: the performance experiments demonstrate that our scheme SOPP costs more time for local computing when compared with the scheme in [11]. However, the comparisons indicate that the proportion of the time cost for local computation (about 20% in SOPP and 5% in [11]) is much smaller than the proportion of the time cost for data transmission, therefore, the effect of the time cost of local computing is much less important. Meanwhile, compared with the scheme in [11], SOPP costs less time for data transmission, while the total transported data size is significantly decreased to achieve a lower network throughput in the experiments. Hence, the resource-constrained IoT devices can save more energy and lead to a longer battery life [12]. In conclusion, we state that the proposed scheme SOPP is more appropriate for data collection when these use resource-constrained IoT devices as data providers.

TABLE 1. A comparison of average data quantity and total data size in each data transmission.

	Transported data size (byte)	Quantity	Total size (byte)
[11]	256	27	4435
SOPP	256	15	1866
[11]	512	58	9094
SOPP	512	33	3850

V. SECURITY ANALYSIS

Based upon the security definition (Section III.E), the correctness of SOPP is demonstrated at first, then the cryptanalysis for the OW-CCA security of the proposed scheme is illustrated in the next section. The authentication, confidentiality and the integrity of SOPP are analyzed briefly as the last three parts in this section.

A. CORRECTNESS

1) AUTHENTICATION

In the *Authentication* phase, the public clouds can recover H_{ID} via computing $C_4 \oplus H_3(bC_2, C_1, C_2, C_3)$ based upon the algorithm *Authentication* in Section IV.A.4).

$$\begin{aligned} &C_4 \oplus H_3(bC_2, C_1, C_2, C_3) \\ &\quad = H_{ID} \oplus H_3(r_2B, C_1, C_2, C_3) \oplus H_3(bC_2, C_1, C_2, C_3) \\ &\quad = H_{ID} \oplus H_3(r_2B, C_1, C_2, C_3) \oplus H_3(br_2G, C_1, C_2, C_3) \\ &\quad = H_{ID} \oplus H_3(r_2B, C_1, C_2, C_3) \oplus H_3(r_2(bG), C_1, C_2, C_3) \\ &\quad = H_{ID} \oplus H_3(r_2B, C_1, C_2, C_3) \oplus H_3(r_2B, C_1, C_2, C_3) \\ &\quad = H_{ID} \end{aligned}$$

2) DECRYPT

When the data center receives $C' = (H_{ID}, C_1, C_2, C_3)$ from the public clouds, the data center can retrieve encrypted data $AES_{H_{ID}}(H_M||M)$ via computing $C_3 \oplus H_2(aC_1)$ based upon the algorithm *Decrypt* in Section IV.A.5).

$$\begin{aligned} &C_3 \oplus H_2(aC_1) \\ &\quad = AES_{H_{ID}}(H_M||M) \oplus H_2(r_1A) \oplus H_2(aC_1) \\ &\quad = AES_{H_{ID}}(H_M||M) \oplus H_2(r_1A) \oplus H_2(ar_1G) \\ &\quad = AES_{H_{ID}}(H_M||M) \oplus H_2(r_1A) \oplus H_2(r_1(aG)) \\ &\quad = AES_{H_{ID}}(H_M||M) \oplus H_2(r_1A) \oplus H_2(r_1A) \\ &\quad = AES_{H_{ID}}(H_M||M) \end{aligned}$$

Then the data center can decrypt $AES_{H_{ID}}(H_M||M)$ with the AES key $H_{ID} \in C'$ to get the plaintext $H_M||M$.

B. OW-CCA SECURITY

Theorem 1: According to Definition 1, our proposed scheme SOPP is OW-CCA secure based upon the ECDH assumption against a Type-I adversary in the random oracle model.

To be specific, let H_1, H_2 and H_3 be three random oracles and $\mathcal{A}_1$ be a Type-I adversary with the advantage $Adv_{\mathcal{A}_1}$ against our proposed scheme. Hypothetically, $\mathcal{A}_1$ requests a total of $Q_{H_2} > 0$ queries to the oracle H_2, then there is an algorithm $\mathcal{E}$ that can solve the ECDH problem with the advantage at least of $\frac{1}{Q_{H_2}}(Adv_{\mathcal{A}_1} - \frac{1}{2^\lambda})$.

Proof: The selected elliptic curve $E_p(a, b)$ with cryptographic security, the group $\mathbb{G}$ is based upon $E_p(a, b)$ and the three points on the curve $(G, \mu G, \upsilon G) \in E_p(a, b)$ consist of an instance of the ECDH problem and the target device's index is defined as t $(1 \leqslant t \leqslant n)$. $\mathcal{E}$ aims to compute $\delta^* = \mu\upsilon G$ via executing $\mathcal{A}_1$ as the subroutine. Next, $\mathcal{E}$ and $\mathcal{A}_1$ play the game defined by **Game 1**.

- *Setup*

$\mathcal{E}$ firstly generates the public parameter pp and then sends pp to $\mathcal{A}_1$. After that, $\mathcal{E}$ operates the algorithm *KeyInitialise* (Section IV.A.2) to generate n public and private key pairs (pk_i, sk_1^i, sk_2^i) $(1 \leqslant i \leqslant n,\ i \neq t)$. In this process, the target device's public key is defined as $pk_t = (A_t, B_t)$, $A_t = \mu_t G$, $B_t = \upsilon_t G$, where $\mu_t, \upsilon_t \in_R \mathbb{Z}_p^*$ is picked randomly. All pk_i are revealed to the adversary $\mathcal{A}_1$. Finally, $\mathcal{E}$ initializes three empty lists $List_{H_1}$, $List_{H_2}$ and $List_{H_3}$, and updates them continuously in response to random oracle queries. If the same input is asked multiple times, the same answer will be returned as well.

- *Queries*

$\mathcal{E}$ can respond to the queries requested by $\mathcal{A}_1$ in the following ways:

1. $Query_{H_1}(\gamma_1)$: $\mathcal{E}$ picks $\delta_1 \in \{0, 1\}^\lambda$ randomly and stores a new item (γ_1, δ_1) into $List_{H_1}$ and returns δ_1 as the answer.
2. $Query_{H_2}(\gamma_2)$: $\mathcal{E}$ picks $\delta_2 \in \{0, 1\}^{2\lambda}$ randomly and stores a new item (γ_2, δ_2) into $List_{H_2}$ and returns δ_2 as the answer.
3. $Query_{H_3}(\gamma_3, C_1, C_2, C_3)$: $\mathcal{E}$ picks $\delta_3 \in \{0, 1\}^\lambda$ randomly and stores a new item $(\gamma_3, C_1, C_2, C_3, \delta_3)$ into $List_{H_3}$ and returns δ_3 as the answer.
4. *Key retrieve query*(i): $\mathcal{E}$ sends the private key $sk_1^i = (\mu_i)$, $sk_1^i = (\upsilon_i)$ to $\mathcal{A}_1$.
5. *Authentication query*(i, C): $\mathcal{E}$ returns the trapdoor $\mathcal{T}_i = H_3(\upsilon_i C_2, C_1, C_2, C_3)$ to recover H_{ID}, where $C = (C_1, C_2, C_3, C_4)$.
6. *Decryption query*(i, C'): The definition of parameter C' is $C' = (C_1, C_2, C_3)$, and there is a conditional branch caused by i to be discussed.

- $i = t$: For each item (γ_2, δ_2) in the $List_{H_2}$, $\mathcal{E}$ performs the following operations.

(i) Compute $AES_{H_{ID}}(H_M||M) = C_3 \oplus \delta_2$ and $H_{ID} = C_4 \oplus H_3(\upsilon_t C_2, C_1, C_2, C_3)$;

(ii) Compute $M = AES'_{H_{ID}}(AES_{H_{ID}}(H_M||M))$;

(iii) If $H_1(M) = H_M$ holds, $\mathcal{E}$ returns M to $\mathcal{A}_1$. If there is no item in the $List_{H_2}$ satisfies the above condition, $\mathcal{E}$ returns $\perp$ to $\mathcal{A}_1$.

- $i \neq t$: $\mathcal{E}$ runs algorithm $Decrypt(pp, C, sk_1^i)$, and then sends the output to $\mathcal{A}_1$ as the answer.

- *Challenge*

$\mathcal{E}$ picks two random numbers, one is $r_1 \in_R \mathbb{Z}_p^*$ and the other one is $r^* \in \{0, 1\}^{2\lambda}$. Then, $\mathcal{E}$ generates a random message $M^* \in \{0, 1\}^\lambda$ and computes the ciphertext $C^* = (C_1^*, C_2^*, C_3^*, C_4^*)$ via the following operations.

$$\begin{aligned} C_1^* &= \upsilon G \\ C_2^* &= r_2 G \\ C_3^* &= r^* \\ C_4^* &= H_{ID} \oplus H_3(r_2 B_t, C_1^*, C_2^*, C_3^*) \end{aligned}$$

Note that the process of decrypting C_3^* is $r^* \oplus H_2(\mu C_1^*) = r^* \oplus H_2(\mu\upsilon G) = r^* \oplus H_2(\delta^*)$ by the definition of *Decrypt*.

Finally, $\mathcal{E}$ sends the ciphertext C^* to the adversary $\mathcal{A}_1$.

- *Constraints*

(1) The target device's index t is not allowed to appear in the *Key retrieve query*;

(2) The target device's index t and the challenge ciphertext C' are not allowed to appear in *Decryption query*.

- *Guess*

$\mathcal{A}_1$ outputs $M^{*'} \in \{0, 1\}^\lambda$ to response the challenge from $\mathcal{E}$. And at the same time, $\mathcal{E}$ picks a random item (γ_2, δ_2) from the $List_{H_2}$ as the solution to the above given instance of ECDH problem.

- *Analysis*

We first define an event E that means the adversary $\mathcal{A}_1$ issues a query $H_2(\delta^*)$ at a time point during the described game. Apparently, δ^* is at least in one item of $List_{H_2}$ at the end of this game if E happened.

However, if E does not happen, we can state that $Pr[M^* = M^{*'}|\neg E] = \frac{1}{2^\lambda}$. Furthermore, based upon the definition of Type-I adversary $(\mathcal{A}_1)$, $Adv_{\mathcal{A}_1} \leqslant Pr[M^* = M^{*'}]$ holds. Then, we can present the following derivation.

$$\begin{aligned} Pr[M^* = M^{*'}] &= Pr[M^* = M^{*'}|E]Pr[E] \\ &\quad + Pr[M^* = M^{*'}|\neg E]Pr[\neg E] \\ &\leqslant Pr[E] + Pr[M^* = M^{*'}|\neg E]Pr[\neg E] \\ &= Pr[E] + \frac{1}{2^\lambda}Pr[\neg E] \\ &= Pr[E] + \frac{1}{2^\lambda}(1 - Pr[E]) \\ &= \frac{1}{2^\lambda} + (1 - \frac{1}{2^\lambda})Pr[E] \end{aligned}$$

Therefore, the following inequation holds:

$$\frac{1}{2^\lambda} + (1 - \frac{1}{2^\lambda})Pr[E] \geqslant Pr[M^* = M^{*'}] \geqslant Adv_{\mathcal{A}_1}.$$

Finally, we can simplify the inequation to get:

$$Pr[E] \geqslant Adv_{\mathcal{A}_1} - \frac{1}{2^\lambda}.$$

In conclusion, at the end of the game between $\mathcal{E}$ and $\mathcal{A}_1$, the probability of δ^* in the item(s) of $List_{H_2}$ is at least

$Adv_{\mathcal{A}_1} - \frac{1}{2^\lambda}$. For $\mathcal{E}$, the probability of generating the correct answer $M^{*'} = M^*$ to solve the ECDH problem is at least $\frac{1}{Q_{H_2}}(Adv_{\mathcal{A}_1} - \frac{1}{2^\lambda})$. Therefore, the probability $Adv_{\mathcal{A}_1}$ is negligible when the ECDH assumption is intact. □

C. AUTHENTICATION

The public clouds first recover H_{ID} from the received data in the *Authentication* phase. If there is no matched identity when H_{ID} is searched for in the identity list $List - H_{ID}$, the integrity check indicates that the identity of the data source (the IoT device) is invalid or the data has been corrupted. Since H_{ID} is recovered from C_4, and C_4 is generated from H_{ID}, C_1, C_2, C_3, H_{ID} decrypted from $C_4 = H_{ID} \oplus H_3(r_2B, C_1, C_2, C_3)$ would not be found in the list $List - H_{ID}$ if H_{ID}, C_1, C_2, or C_3 is forged in transmission. Therefore, the calculation dependency between C_4 and (H_{ID}, C_1, C_2, C_3) can ensure the invalid transmission can be found and is blocked in *Authentication*.

D. CONFIDENTIALITY

The focus of our scheme's confidentiality is to ensure the untrusted public clouds cannot decrypt the encrypted data part C_3 of C in the *Authentication* process by the definition in Section III.B.3).

After receiving the encrypted data C from the IoT device, the public clouds can only retrieve H_{ID} with the algorithm *Authentication* in Section IV.A.4). If the public clouds attempt to decrypt C_3, they need an algorithm to calculate a number r^* equal to a or r_1 (because $AES_{H_{ID}}(H_M||M) = C_3 \oplus H_2(aC_1) = C_3 \oplus H_2(r_1A)$), where the public key (A, B) and the ciphertext C are known. However, the public clouds have no probabilistic polynomial-time algorithm as a subroutine in $\mathcal{B}$ to use $G, C_1 = r_1G, A = aG$ to calculate $r^* = a$ (or r_1) based upon the ECDH assumption. Hence, the advantage of the untrusted public clouds

$$\begin{aligned} Adv_{\mathcal{B}} &= Pr[c = M | c = \mathcal{B}(pp, C, sk_2)] \\ &= Pr[c = a \vee c = r_1 | c = \mathcal{B}(pp, C_3, H_{ID}, sk_2)] \end{aligned}$$

is negligible for generating r^* to recover M from C successfully, which means the confidentiality of M can be secured in the phase *Authentication*.

E. INTEGRITY

In our proposed scheme SOPP, the data integrity means to ensure that the transmitted data can be tamper-proof in the data transmission from public clouds to the data center. Here, we discuss two potential attacks during the data transmission.

a) The encrypted data $C_3 = AES_{H_{ID}}(H_M||M) \oplus H_2(r_1A)$ or the identity H_{ID} from the *Authentication* phase is manipulated by the attacker during data transmission.

According to the *Decrypt* phase, C_3 can be decrypted with the AES decryption algorithm to retrieve the plaintext M and H_M. However, if C_3 or H_{ID} (the key for AES decryption) is changed, the AES decryption algorithm cannot output the correct $H_M||M$. This is because AES encryption and decryption are symmetric, which means even if there is only one incorrect bit in C_3 or H_{ID}, the encryption or decryption result will be wrong. In this situation, the hash validation in the *Decrypt* phase will fail, i.e., $H_1(M) \neq H_M$ holds because of the incorrect M and H_M. Therefore, we can state that SOPP can take advantage of the integrity validation to block the manipulated data by the attacker between the untrusted public clouds and the data center.

b) The attacker can control the untrusted public clouds and the identity list $List - H_{ID}$ has been disclosed.

Since the public clouds have been controlled, the attacker can recover the IoT device's identity H_{ID} from the received ciphertext $C = (C_1, C_2, C_3, C_4)$ in the *Authentication* phase. Furthermore, the attacker can also substitute a forged invalid identity or another valid identity in the identity list $List - H_{ID}$ for H_{ID}. On the other hand, it is possible for the attacker to modify the ciphertext $C_3 \in C$ directly under this situation. However, the probability for the attacker to decrypt C_3 is still negligible based upon our security analysis in Section V.B and V.D. Regardless of the validity of the selected identity H_{ID}^* and the forged C_3^* by the attacker, when the data center receives the encrypted data $C'^* = (H_{ID}^*, C_1, C_2, C_3^*)$, the data center can use the algorithm *Decrypt* to validate the integrity of the decrypted data M^* and H_M^* from C^* to detect falsification because of the analysis illustrated in a). Hence, SOPP can ensure data integrity and avoid the plain data M to be leaked when public clouds are manipulated by the attacker.

Overall, according to the above security analysis, when attacks occur during the data transmission from the IoT device to the data center via the untrusted public clouds, SOPP can provide sufficient safeguards to ensure the confidentiality and integrity of the transmitted private data.

VI. CONCLUSION

In this paper, we propose a new privacy-preserving transmission scheme, Semi-Outsourcing Privacy Preserving (SOPP), for use in data transmission scenarios from IoT devices. Compared with original work constructed with PKI and trusted clouds, the use of trusted private clouds with high construction expense is not needed to deploy SOPP to ensure the security and privacy in data transmission. The authentication and the decryption (including the integrity validation) are divided (semi-outsourcing) to adapt to a more general cloud architecture for data acquisition from IoT devices, where the data center is separated from untrusted public clouds. Meanwhile, SOPP is more suitable to be deployed in resource-constrained IoT devices because it can lower the network throughput and time cost in data transmission. Therefore, SOPP is more economical and practical for use by small and medium-sized enterprises and organizations that cannot afford the cost for constructing large-scale trusted clouds but instead tend to take advantage of untrusted public clouds as part of their Information Communication Technology (ICT) service infrastructure.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.

[2] B. Seitz. *The Importance of IoT Data Collection*. Accessed: Aug. 20, 2016. [Online]. Available: https://buddy.com/blog/importance-iot-data-collection/

[3] Y. Mehmood, F. Ahmad, I. Yaqoob, A. Adnane, M. Imran, and S. Guizani, "Internet-of-Things-based smart cities: Recent advances and challenges," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 16–24, Sep. 2017.

[4] L. Titkov, S. Poslad, and J. J. Tan, "An integrated approach to user-centered privacy for mobile information services," *Appl. Artif. Intell.*, vol. 20, nos. 2–4, pp. 159–178, 2006.

[5] S. Poslad, M. Hamdi, and H. Abie, "Adaptive security and privacy management for the Internet of Things (ASPI 2013)," in *Proc. ACM Conf. Pervasive Ubiquitous Comput. Adjunct Publication*, 2013, pp. 373–378.

[6] W. Leister, M. Hamdi, H. Abie, and S. Poslad, "An evaluation framework for adaptive security for the IoT in ehealth," *Int. J. Adv. Secur.*, vol. 7, nos. 3–4, pp. 93–109, 2014.

[7] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina, "Controlling data in the cloud: Outsourcing computation without outsourcing control," in *Proc. ACM Workshop Cloud Comput. Secur.*, 2009, pp. 85–90.

[8] S. Bera, S. Misra, and J. J. P. C. Rodrigues, "Cloud computing applications for smart grid: A survey," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 5, pp. 1477–1494, May 2015.

[9] W. A. Jansen and T. Grance, "Guidelines on security and privacy in public cloud computing," NIST, Gaithersburg, MD, USA, Tech. Rep. (NIST SP)-800-144, 2011. doi: 10.6028/NIST.SP.800-144.

[10] H. Suo, J. Wan, C. Zou, and J. Liu, "Security in the Internet of Things: A review," in *Proc. Int. Conf. Comput. Sci. Electron. Eng. (ICCSEE)*, vol. 3, Mar. 2012, pp. 648–651.

[11] T. Fan and Y. Chen, "A scheme of data management in the Internet of Things," in *Proc. 2nd IEEE Int. Conf. Netw. Infrastruct. Digit. Content*, Sep. 2010, pp. 110–114.

[12] G. Margelis, R. Piechocki, D. Kaleshi, and P. Thomas, "Low throughput networks for the IoT: Lessons learned from industrial implementations," in *Proc. IEEE 2nd World Forum Internet Things (WF-IoT)*, Dec. 2015, pp. 181–186.

[13] C. Doukas, I. Maglogiannis, V. Koufi, F. Malamateniou, and G. Vassilacopoulos, "Enabling data protection through PKI encryption in IoT m-Health devices," in *Proc. IEEE 12th Int. Conf. Bioinf. Bioeng. (BIBE)*, Nov. 2012, pp. 25–29.

[14] F. Tao, Y. Zuo, L. Da Xu, and L. Zhang, "IoT-based intelligent perception and access of manufacturing resource toward cloud manufacturing," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1547–1557, May 2014.

[15] H. Hu, J. Xu, C. Ren, and B. Choi, "Processing private queries over untrusted data cloud through privacy homomorphism," in *Proc. IEEE 27th Int. Conf. Data Eng. (ICDE)*, Apr. 2011, pp. 601–612.

[16] B. Wang, B. Li, and H. Li, "Oruta: Privacy-preserving public auditing for shared data in the cloud," *IEEE Trans. Cloud Comput.*, vol. 2, no. 1, pp. 43–56, Jan./Mar. 2014.

[17] P. Szczechowiak, L. B. Oliveira, M. Scott, M. Collier, and R. Dahab, "NanoECC: Testing the limits of elliptic curve cryptography in sensor networks," in *Wireless Sensor Networks*. Berlin, Germany: Springer, 2008, pp. 305–320.

[18] N. R. Potlapally, S. Ravi, A. Raghunathan, and N. K. Jha, "A study of the energy consumption characteristics of cryptographic algorithms and security protocols," *IEEE Trans. Mobile Comput.*, vol. 5, no. 2, pp. 128–143, Feb. 2006.

[19] X. Zhang, S. Poslad, and Z. Ma, "A semi-outsourcing secure data privacy scheme for IoT data transmission," in *Proc. IEEE 28th Annu. Int. Symp. Personal, Indoor, Mobile Radio Commun. (PIMRC)*, Oct. 2017, pp. 1–5.

[20] D. Stinson, *Cryptography: Theory and Practice*. London, U.K.: Chapman & Hall, 2005.

[21] M. Maksimović, V. Vujović, N. Davidović, V. Miloševič, and B. Perišić, "Raspberry Pi as Internet of Things hardware: Performances and constraints," *Des. Issues*, vol. 3, p. 8, Jun. 2014.

[22] M. Scott. *MIRACL Multiprecision Integer and Rational Arithmetic C/C++ Library*. Accessed: Jun. 6, 2018. [Online]. Available: https://github.com/miracl/MIRACL

[23] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, "Recommendation for key management part 1: General (revision 3)," NIST, Gaithersburg, MD, USA, Tech. Rep. (NIST SP)–800-57, Pt1 Rev. 3, 2012, pp. 1–147, vol. 800, no. 57. 10.6028/NIST.SP.800-57p1r3.

XIAOSHUAI ZHANG received the B.Sc. and M.Sc. degrees from the Department of Computer Science and Technology, Ocean University of China, China. He is currently pursuing the Ph.D. degree with the School of Electronic Engineering and Computer Science, Queen Mary University of London, U.K. His current research interests include security and privacy in the IoT, blockchain, and applied cryptography.

CHAO LIU received the B.S. degree (Hons.) from the Beijing University of Posts and Telecommunications, China, in 2016. He is currently pursuing the Ph.D. degree with the School of Electronic Engineering and Computer Science, Queen Mary University of London, London, U.K. His research interests include the Internet of Things application, smart grid, resource allocation optimization, blockchain technology, and cryptocurrency trading.

STEFAN POSLAD received the Ph.D. degree from Newcastle University, Newcastle upon Tyne, U.K. He is currently an Associate Professor with the School of Electronic Engineering and Computer Science, Queen Mary University of London, London, U.K. He has authored the book *Ubiquitous Computing: Smart Devices, Environments and Interaction*, in 2009. His research interests are ubiquitous computing, the Internet of Things, management and security of distributed systems, semantic web, and software agents. He has led and been active in several international collaborative projects in these areas and has over 100 related research publications.

KOK KEONG CHAI (MICHAEL) received the B.Eng. degree (Hons.) and the M.Sc. and Ph.D. degrees, in 1998, 1999, and 2007, respectively. He is currently a Senior Lecturer (Associate Professor) with the School of Electronic Engineering and Computer Science, Queen Mary University of London (QMUL). He is the Internet of Things Programme Lead for the Joint Programme between QMUL and the Beijing University of Posts and Telecommunications and a member of the Communication Systems Research Group, QMUL. He has authored over 65 technical journal and conference papers in the areas of machine-to-machine communications, wireless communications, the Internet of Things, and smart grid. His research interests include radio resource allocation in device-to-device communication, mobile and wireless communications, energy efficiency of machine-to-machine communications, smart cities applications, smart energy charging schemes, and smart grids.