IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2018.DOI

Enabling Robust and Privacy-Preserving Resource Allocation in Fog Computing

LEI ZHANG[1,2], (Member, IEEE), JIANGTAO LI[1].

[1]Shanghai Key Laboratory of Trustworthy Computing, School of Computer Science and Software Engineering, East China Normal University, Shanghai, 200062, China

[2]State Key Laboratory of Cryptology, P.O. Box 5159, Beijing, 100878, China

Corresponding author: Lei Zhang (e-mail: leizhang@sei.ecnu.edu.cn).

This work is supported in part by the National Key R&D Program of China (No. 2017YFB0802000), by the NSF of China under Grants 61572198, 61321064.

ABSTRACT Fog computing is an extension of cloud computing and enables computing directly at the edge of the network. In fog computing paradigm, fog nodes reside between smart end devices and the cloud. Benefit from the structure of fog computing, fog computing can provide services with low latency, location awareness and mobility. Since the fog nodes are not as powerful as the cloud, resource allocation techniques are usually adapted to optimize the utilization of the resources of fog nodes. However, current resource allocation techniques are not privacy-preserving, i.e., an attacker can easily find end devices' sensitive information. In this paper, we propose a privacy-preserving resource allocation scheme for fog computing. The new proposal has constant message expansion, and, is secure against both an eavesdropper and a smart gateway that is employed to perform the resource allocation algorithm. Our scheme is also robust, since it achieves full key compromise resistance which guarantees that even if the private keys of all the fog nodes in a fog system are corrupted the scheme remains secure.

INDEX TERMS Fog computing, resource allocation, user privacy, internet of things.

I. INTRODUCTION

Internet of Things (IoT) is a framework that enables efficient message exchanges between various end devices (e.g., intelligent manufacturing equipments, intelligent vehicles, smart phones) and controllers [1]–[3]. In industries, IoT has promising applications in industrial automation, safer mining production, food supply chain, transportation, smart grid etc. [4]–[7]. With the rapid development of IoT, tremendous data is generated by various IoT devices everyday [8]. Cloud computing [9] is a promising platform to process the data generated by those IoT devices, since it has powerful enough computational and storage capabilities. However, the quality of cloud service will be affected by various factors, such as the stability of the connection between a device and the cloud. At the same time, latency sensitive services require real-time response. To address the above issues, fog computing is introduced [1], [10], [11]. In fog computing, fog nodes (e.g., access points, smart gateways and edge routers) reside between end devices and the cloud. Low latency services can be easily handled, since fog computing is implemented at the edge of the network. In addition to low latency, fog computing may also have the advantage of location awareness and mobility [12].

Typically, a fog node is an edge device with limited computational and storage capabilities. This is quite different from the cloud which is usually considered as a powerful enough entity. Resource allocation technique [13], [14] for fog computing enables a group of fog nodes to cooperate with each other and optimize the utilization of fog nodes. In such a resource allocation system, an entity (e.g., a smart fog gateway) is usually employed to forward different types of data to suitable host(s) (the cloud or fog node(s)) [15]. For instance, on receiving a message (task), the gateway will forward it to the most suitable host(s) for execution.

To realize different optimization goals (i.e., resource allocation), various factors (e.g., type of the service requested, time sensitivity, computational complexity) should be considered by the gateway to make a decision. For instance, in [18], a novel optimization strategy for path selection in Internet of Vehicles is proposed. In their scheme, both safety factors and computational cost should be taken into consideration. To realize such a system, the task data has to be attached with meta-data which defines the required factors (e.g., type of the service requested, time sensitivity,

computational complexity). For instance, the topic address in meta-data [19] will specify what kind of service is requesting; the time sensitivity will point out whether this task requires real-time response; the computational complexity will imply whether this task requires high computing resources.

Meta-data without any protection will expose end devices' private information. For instance, an attacker may learn what kind of service an end device is requesting. A trivial way to enable privacy-preserving resource allocation in fog computing is to use mature cryptographic tools. For instance, an end device may encrypt the meta-data under the public key of the gateway and then send the encrypted meta-data to the gateway. Obviously, an eavesdropper will no longer have the access to the meta-data. However, by using this method, the gateway may still learn all the meta-data. If the gateway is malicious or corrupted, end devices' private information will be leaked. Hence, it is desirable to design a privacy-preserving resource allocation scheme that is secure even against the gateway. That is, the gateway is able to check whether a specific factor is contained in the received meta-data or not, without learning the content of the meta-data.

In recent rears, only a few resource allocation techniques have been proposed for fog computing [12], [14]–[17], [20]–[24]. Among them, a promising solution for resource allocation in fog computing is to use smart gateways. The concept of smart gateway [15] was introduced to improve the utilization of network and fog/cloud resources. The data collected by an end device will be transmitted to a smart gateway for preprocessing, e.g., data filtering, service monitoring and management, and resource allocation. In [15], Aazam et al. used a smart gateway for data processing management. The gateway is responsible for deciding whether data to be processed should be sent to a cloud or a fog node. Furthermore, the gateway may also process the data itself.

There are also several schemes address the privacy issues in fog computing. In [25], a fog node (a.k.a. cloudlet) is used to provide efficient and privacy-preserving electronic medical data sharing. In [26], the authors considered privacy-preserving disease prediction in fog and cloud computing. In [27], the authors proposed a privacy-preserving outsourced calculation toolkit which allows the cloud/fog node to perform computation on outsourced data. We note that, all the above solutions for resource allocation in fog computing do not consider meta-data privacy.

A. OUR CONTRIBUTION

To the best of our knowledge, there is no privacy-preserving resource allocation scheme that considers meta-data privacy in fog computing. In this paper, we propose a privacy-preserving resource allocation scheme for fog computing by introducing a new security tool called contributory public key searchable encryption (CPKSE).

Our scheme allows a group of fog nodes (who need to deal with tasks in a cooperative manner) interactive with each other to generate a group public key and form a fog system. Each fog node in the fog system then may generate a set of partial trapdoors by using our CPKSE and send it to the gateway. Each partial trapdoor is corresponding to a legal keyword (i.e., a factor defined in meta-data). Then the gateway may generate a full trapdoor by using the partial trapdoors corresponding to the same (legal) keyword form the fog nodes. In our scheme, a piece of encrypted meta-data generated by an end device may consist of one or several keywords encrypted under the group public key. To realize resource allocation, an end device generates and sends the encrypted meta-data (together with the original data) to the gateway. With the full trapdoors, the gateway is able to determine which trapdoor(s) are corresponding to the encrypted keyword(s) in the encrypted meta-data received, and then performs the resource allocation task. Since the keyword(s)/factor(s) defined in the encrypted meta-data is(are) not revealed to the outsiders and the gateway, the privacy of the end device is protected.

Besides the privacy of the end devices, our proposed construction is also robust, i.e., it satisfies the property of full key compromise resistance. The property means that even if the private keys of all the fog nodes in the fog system are corrupted, an attacker is not able to extract any information from meta-data. We also propose a formal security model to define the security of our scheme. The new model captures the privacy protection and full key compromise resistance properties. The security of our scheme is analyzed in the proposed model. Finally, we also show the efficiency of our scheme by several experiments.

B. PAPER ORGANIZATION

The rest of the paper is organized as follows. Section II is the background. We propose our scheme in Section III. Section IV analyzes the security of the proposed scheme. Section V evaluates the performance of our scheme. Section VI is the conclusion.

II. BACKGROUND

A. SYSTEM ARCHITECTURE

As shown in Figure 1, the system consists of five types of entities: trusted authority (TA), end devices, smart gateway, fog nodes and cloud.

- The TA is a fully trusted entity. It is used to generate the system parameters and issue private keys for the fog nodes.
- End devices have limited computational power and storage memory. The end devices are fog users who will send various tasks to fog nodes.
- On receiving a task, the smart gateway will forward the task to a proper fog node or several fog nodes or the cloud. In this paper, the gateway is assumed to be powerful and semi-trusted, i.e., curious but honest. It will deal with the tasks honestly, and will also try to violate the privacy of end devices.
- The fog nodes are implemented through a variety of edge devices. These end devices are equipped with limited computation/storage capability. Generally, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2868920, IEEE Access

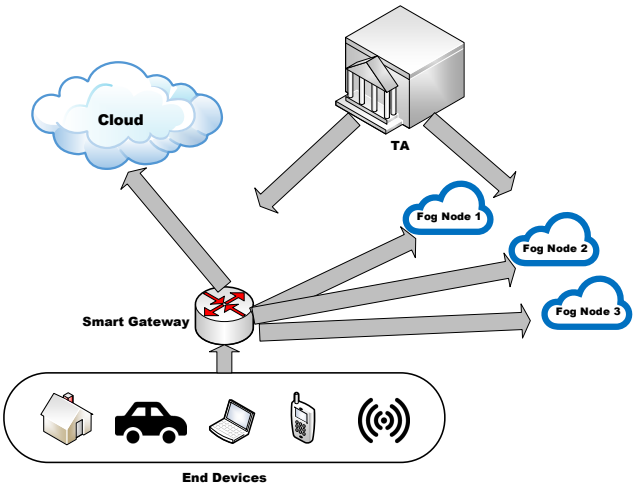


FIGURE 1. System Architecture.

fog nodes will deal with different tasks in an cooperative manner.

- The cloud provides services with powerful enough computational and storage resources. The cloud is responsible for the tasks that are not time sensitive or require a powerful enough execution environment.

B. DESIGN GOALS

In a privacy-preserving resource allocation scheme for fog computing, an attacker could be a gateway and/or an eavesdropper. Our design goals can be summarized as the following:

1) User privacy: For a gateway, it requires that the gateway should be able to check whether a piece of encrypted meta-data contains a legal keyword without learning the contents in the meta-data. For an eavesdropper, it requires that the eavesdropper cannot distinguish whether two encrypted keywords contains the same keyword. A more formal definition is called indistinguishability. It guarantees that, given an encryption of a keyword randomly chosen between two keywords given by an attacker, the attacker cannot distinguish whether the encrypted keyword is corresponding to the first or second keyword.
2) Full key compromise resistance: An attacker cannot violate user privacy, even if the private keys of all the fog nodes in the system are corrupted by an attacker.
3) Constant message expansion: The encrypted keyword corresponding to a keyword should not grow with the number of fog nodes.

C. SEARCHABLE ENCRYPTION

Searchable encryption is a cryptographic primitive that allows a user (who has the trapdoor) to search encrypted data (i.e., encrypted keyword in this paper) without leaking the content of the data. Symmetric searchable encryption (SSE) [28] and public key encryption with keyword search (PEKS) [29] are two branches of searchable encryption. SSE requires a secret key shared between a sender and a trapdoor generator. PEKS enables anyone who has the knowledge of the public key of an entity (fog node in this paper) to produce encrypted keywords. The entity may generate trapdoors corresponding to the keywords using its private key. Using SSE and PEKS, we may achieve user privacy against malicious gateways. However, who will generate a trapdoor is the key problem in SSE/PEKS-based resource allocation schemes for fog computing.

In a cooperative fog system, usually, the fog nodes in the system should decide what kind of information can be leaked to the gateway. It means that the fog nodes should generate a trapdoor in an cooperative manner. Searchable encryption scheme for multiple receivers (fog nodes in this paper) [30] is a tool to solve the problem. However, in current searchable encryption schemes for multiple receivers, if an attacker corrupts one of the receivers, he is able to generate any trapdoor and hence recover the keyword. Although threshold schemes [31] have been proposed to overcome this drawback, these schemes have long ciphertext (i.e., encrypted keyword) size (the ciphertext corresponding to a keyword grows linearly with the number of fog nodes). In this paper, we introduce a new notion called contributory public key searchable encryption (CPKSE) to deal with the resource allocation problem in fog computing.

D. IDENTITY-BASED CRYPTOSYSTEM

Our scheme is designed in identity-based cryptosystem [32], [33] which aims to eliminate the certificate management problem is traditional PKI based system. In our scheme, the public key of a fog node is just its identity. Therefore, there is no need to manage a complicated certificate management system to update and revoke certificates etc.

III. PROPOSED SCHEME

A. HIGH LEVEL DESCRIPTION

As illustrated in Figure 2, our proposed scheme consists of following six phases:

- Initialization: The TA generates a master secret and a list of system parameters. The master secret will be used in the next phase. The system parameters are published.
- Registration: In this phase, each fog node in the system need to be enrolled by the TA. On receiving a registration request from the fog node with identity ID_i, the TA generates a private key based on this identity using the master secret.
- Group Generation: Suppose a group of fog nodes with identities $ID_1, ..., ID_k$ want to form a fog system. The fog nodes in the group interactive with each other to generate a group public key. The group public key is then published.
- Keyword Generation: In this phase, the fog system securely generates a list of keywords for an end device or a group of end devices that want to use the fog system.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2868920, IEEE Access

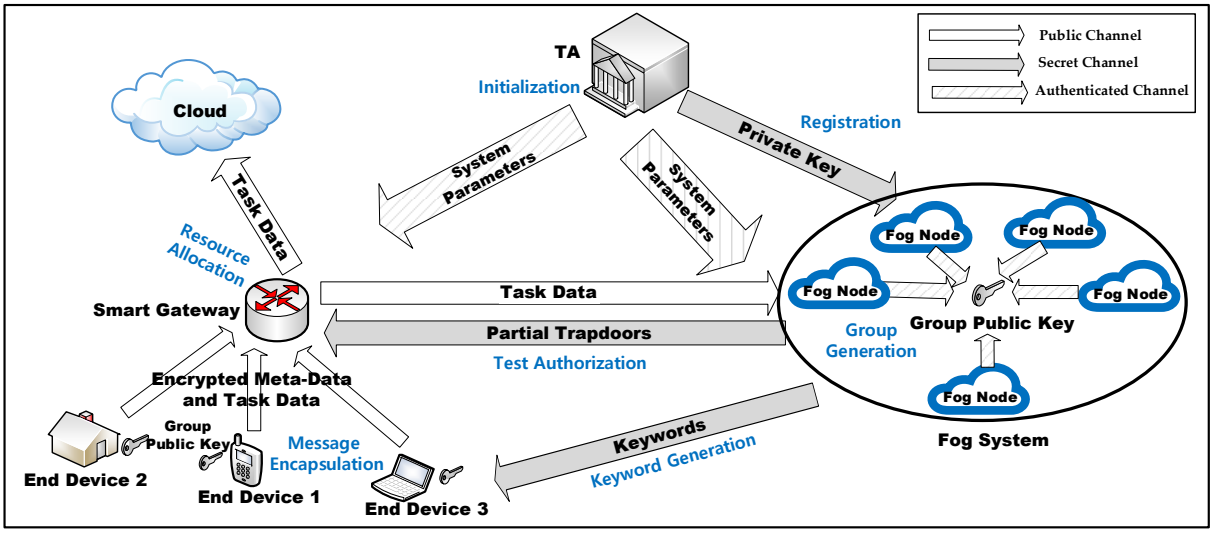


FIGURE 2. High Level Description of Our Scheme.

- Message Encapsulation: Any end device that has the knowledge of the group public key is able to generate encrypted meta-data which may consist of one or several keywords encrypted under the group public key of the fog system. The encrypted meta-data is send to the gateway together with the task data.
- Test Authorization: To authorize a gateway the permission of testing a keyword m (i.e., a specific factor defined in meta-data), each fog node in the group with identity ID_i generates a partial trapdoor on m. On receiving the partial trapdoors generated by all the fog nodes in the group, the gateway is able to generate the final trapdoor.
- Resource allocation: On receiving encrypted meta-data, the gateway is able to run a Test algorithm to determine whether a legal keyword is in the encrypted meta-data. However, the gateway cannot learn the content of the keyword. The gateway will then perform resource allocation based on the test result.

We note that the first six phases and the Test algorithm is the last phase form our CPKSE scheme.

B. THE PROPOSAL

Our scheme is implemented with bilinear maps [34], [35]. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two multiplicative groups of prime order q, and g be a generator of $\mathbb{G}_1$. A map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is called a bilinear map if it satisfies 1) Bilinearity: $\hat{e}(g^\beta, g^\gamma) = \hat{e}(g,g)^{\beta\gamma}$ for all $\beta, \gamma \in \mathbb{Z}_q^*$. 2) Non-degeneracy: There exists $u, v \in \mathbb{G}_1$ such that $\hat{e}(u,v) \neq 1$. 3) Computability: There exists an efficient algorithm to compute $\hat{e}(u,v)$ for any $u, v \in \mathbb{G}_1$. The construction comes as follows.

- Initialization: On input a security parameter ℓ, the TA chooses two cyclic multiplicative groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with prime order q, where $\mathbb{G}_1$ is generated by g and there exists a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$. The TA randomly selects $\eta \in \mathbb{Z}_q^*$ as the system master secret, computes $y = g^\eta$, chooses cryptographic hash functions $H_1, H_2 : \{0,1\}^* \longrightarrow \mathbb{G}_1$, $H_3 : \mathbb{G}_2 \longrightarrow \{0,1\}^\ell$, publishes the system parameters

$$\text{param} = (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, g, y, H_1, H_2, H_3).$$

- Registration: Each fog node needs to be enrolled by the TA to obtain its private key. For a fog node with identity $ID_i \in \{0,1\}^*$, the TA generates the private key for the fog node as follows:
  1) Compute $f_i = H_1(ID_i)$.
  2) Output the private key $s_i = f_i^\eta$.
- Group Generation: Assume a group of fog nodes with identities $ID_1, ..., ID_k$ decide to establish a fog system. This phase allows them to generate a group public key. The fog nodes first negotiate an unique group ID

$$GID = ID_1||ID_2||\ldots||ID_k||serial\ number.$$

  Then the fog nodes does the following:
  1) For $1 \leq i \leq k$, the i-th fog node with identity ID_i chooses a random $\alpha_i \in \mathbb{Z}_q^*$ and computes $u_i = g^{\alpha_i}$. u_i is sent to other fog nodes via an authenticated channel.
  2) Each fog node is able to compute and publish the group public key $E = (u, \Lambda)$, where

$$u = \prod_{i=1}^{k} u_i, \Lambda = \hat{e}(\prod_{i=1}^{k} H_1(ID_i), y).$$

- Keyword Generation: In this phase, a list of keywords is generated by the fog system, and sent to an end device or a group of end devices that want to use the fog system through a secure channel. To resistant the keyword guessing attack, we may choose a random string as a keyword. The end device(s) and the fog system have to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2868920, IEEE Access

record the corresponding relations between the keyword and a required factor.

- Message Encapsulation: An end device may send encrypted meta-data with the task data to the gateway. For a keyword m in the meta-data, an end device selects $\beta \in \mathbb{Z}_q^*$, computes the encrypted keyword (a, b), where

$$a = g^{\beta}, b = H_3((\hat{e}(H_2(GID, m), u) \cdot \Lambda)^{\beta}).$$

- Test Authorization: Let the keywords generated by the fog system for an end device or a group of end devices be $(m_1, ..., m_n)$. To authorize a gateway the permission to test a keyword $m_l, l \in \{1, ..., n\}$, the i-th fog node in the group with $GID = ID_1||ID_2||\ldots||ID_k||serial\ \ number$ computes the partial trapdoor

$$v_i = s_i H_2(GID, m_l)^{\alpha_i},$$

and sends v_i, tag_i to the gateway via a secret channel, where $tag \in \{0, 1\}$. If $tag = 0$, it denotes that the i-th fog node is unwilling to perform the task defined by m; else if $tag = 1$, it denotes that the node is willing to perform the task defined by m. On receiving $\{v_i, tag_i\}_{1 \le i \le k}$, the gateway computes

$$T_{m_l} = \prod_{i=1}^{k} v_i.$$

T_{m_l} is the trapdoor for testing whether an piece of encrypted meta-data intended for the group with group ID GID contains a keyword m_l. We note that the fog nodes may issue a pool of trapdoors corresponding to different keywords for the gateway. A keyword is able to specify whether the task data is time sensitive, need high computation/storage resource, or belong to a specific topic address, and so on.

- Resource Allocation: Let (a_j, b_j) be an encrypted keyword in encrypted meta-data. The gateway with trapdoor T_{m_l} runs following Test algorithm

$$H_3(\hat{e}(T_{m_l}, a_j)) \stackrel{?}{=} b_j.$$

If the equation holds, the gateway learns that a legal keyword m_l is contained in the encrypted meta-data; otherwise m_l is not contained in the encrypted meta-data. After testing all the keywords in the encrypted meta-data, then the gateway may perform resource allocation based on the tags corresponding to the trapdoor. We note that if none of the fog nodes is willing to perform the task, then the task is sent to the cloud.

IV. SECURITY ANALYSIS

In the section, we first define the security model for our resource allocation scheme. In particular, our model capture the properties of user privacy and full key compromise resistance. Then we show that our proposal is secure in this model.

A. SECURITY MODEL

As discussed in Section II-B, an adversary could be a gateway or an eavesdropper. In the following, we say an adversary is a type I adversary, if the adversary is a gateway; otherwise, if an adversary is an eavesdropper, then we say the adversary is a type II adversary. Obviously, our scheme should be secure against both types of adversaries. In the following, we propose two games to define the security of our scheme.

Game 1 is illustrated in Table 1. It captures the behaviors of a type I adversary. The game runs between a challenger $\mathcal{CH}$ and a type I adversary $\mathcal{A}$. $\mathcal{CH}$ will simulate the real environment for $\mathcal{A}$. $\mathcal{A}$ will try to learn the content in encrypted meta-data. The game has following three phases:

Initial: It is corresponding to our Initialization phase. $\mathcal{CH}$ generates the master-secret and the system parameters, then sends the system parameters to $\mathcal{A}$.

Training: $\mathcal{A}$ can perform a polynomially bounded number of following types of queries in an adaptive manner.

- Private key query: It is corresponding to the Registration phase. $\mathcal{A}$ can request the private key of a fog node with identity ID_i. In response, $\mathcal{CH}$ outputs the private key corresponding to ID_i. This query is used to model the full key compromise resistance property of our scheme. $\mathcal{A}$ is allowed to obtain the private keys of all the fog nodes in the system.
- Group public key query: It is corresponding to the Group Generation phase. $\mathcal{A}$ can request the group public key corresponding to a group of fog nodes. On receiving such a query, $\mathcal{CH}$ returns the group public key corresponding to the group.
- Trapdoor query: It is corresponding to the Test Authorization phase. $\mathcal{A}$ can request the trapdoor of a keyword m corresponding to a designated group. On receiving such a query, $\mathcal{CH}$ returns the trapdoor corresponding to m.

We note that the Keyword Generation and Resource Allocation phases will not influence the security of our scheme and can be performed as in the real world. Therefore, in the security model will not define the queries corresponding to these two phases.

Output: $\mathcal{A}$ chooses a group ID GID^* and sends GID^* to $\mathcal{CH}$. $\mathcal{CH}$ randomly chooses a keyword m^*, encrypts m^* to generate a trapdoor T_{m^*}. T_{m^*} is sent to $\mathcal{A}$. $\mathcal{A}$ wins the game if he can decrypt T_{m^*} correctly, i.e., $m^* = m^{*\prime}$.

We say our scheme is secure against a type I adversary, if for any probabilistic polynomial time adversary $\mathcal{A}$, the advantage for $\mathcal{A}$ to win the above game is negligible. $\mathcal{A}$'s advantage is defined to be

$$\mathsf{Adv}(\mathcal{A}) = |\Pr[\xi' = \xi] - \frac{1}{2}|.$$

Game 2 is illustrated in Table 2 and captures the behaviors of a type II adversary. The game runs between a challenger $\mathcal{CH}$ and a type II adversary $\mathcal{A}$. $\mathcal{CH}$ will simulate the real

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2868920, IEEE Access

TABLE 1. Security Model Corresponding to a Type I Adversary

Challenger $\mathcal{CH}$		Type I adversary $\mathcal{A}$
Generate *system parameters*, *master-secret*		
	$\xrightarrow{\textit{System parameters}}$	
	$\xleftarrow{\textit{Private key query / Group public key query / Trapdoor query}}$	
	$\xrightarrow{\textit{Private key / Group public key / Trapdoor}}$	
	$\vdots$	
		Select a group ID GID^*
	$\xleftarrow{GID^*}$	
Choose a keyword m^* at random and compute the corresponding trapdoor T_{m^*}		
	$\xrightarrow{T_{m^*}}$	
		Output a guess $m^{*\prime}$

TABLE 2. Security Model Corresponding to a Type II Adversary

Challenger $\mathcal{CH}$		Type II adversary $\mathcal{A}$
Generate *system parameters*, *master-secret*		
	$\xrightarrow{\textit{System parameters}}$	
	$\xleftarrow{\textit{Private key query / Group public key query / Trapdoor query}}$	
	$\xrightarrow{\textit{Private key / Group public key / Trapdoor}}$	
	$\vdots$	
		Select two keywords m_0, m_1 and a group ID GID^* corresponding to a group of fog nodes
	$\xleftarrow{m_0, m_1, GID^*}$	
Choose $\xi \in \{0,1\}$ at random and compute an encrypted keyword C^* corresponding to m_ξ		
	$\xrightarrow{C^*}$	
		Output a guess ξ'

environment for $\mathcal{A}$. $\mathcal{A}$ will try to distinguish whether two encrypted keywords contains the same keyword. The game has following three phases:

Initial: It is corresponding to our Initialization phase. $\mathcal{CH}$ generates the master-secret and the system parameters, then sends the system parameters to the adversary $\mathcal{A}$.

Training: $\mathcal{A}$ can perform a polynomially bounded number of following types of queries in an adaptive manner.

- Private key query: It is corresponding to the Registration phase. $\mathcal{A}$ can request the private key of a fog node with identity ID_i. In response, $\mathcal{CH}$ outputs the private key corresponding to ID_i. This query is used to model the full key compromise resistance property of our scheme. $\mathcal{A}$ is allowed to obtain the private keys of all the fog nodes in the system.
- Group public key query: It is corresponding to the Group Generation phase. $\mathcal{A}$ can request the group public key corresponding to a group of fog nodes. On receiving such a query, $\mathcal{CH}$ returns the group public key corresponding to the group.
- Trapdoor query: It is corresponding to the Test Authorization phase. $\mathcal{A}$ can request the trapdoor of a keyword m corresponding to a designated group. On receiving such a query, $\mathcal{CH}$ returns the trapdoor corresponding to m.

Similar to game 1, we will not define the queries corresponding to Keyword Generation and Resource Allocation phases.

Output: $\mathcal{A}$ chooses two keywords m_0, m_1 and a group ID GID^*. m_0, m_1, GID^* are sent to $\mathcal{CH}$. It requires that the trapdoor of m_0 and m_1 corresponding to group GID^* have

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2868920, IEEE Access

not been queried. However, we allow $\mathcal{A}$ to corrupt all the fog nodes in the group corresponding to GID^*. In response, $\mathcal{CH}$ randomly chooses $\xi \in \{0,1\}$, encrypts m_ξ under the group public key GID^* to obtain an encrypted keyword C^*. C^* is sent to $\mathcal{A}$. Finally, $\mathcal{A}$ outputs its guess $\xi' \in \{0,1\}$.

We say our scheme is secure against a type II adversary if $\mathcal{A}$'s advantage

$$\mathsf{Adv}(\mathcal{A}) = |\Pr[\xi' = \xi] - \frac{1}{2}|$$

is negligible.

B. SECURITY RESULT

The security of our scheme is based on the one-way property of the hash function [36] and the Bilinear Diffie-Hellman (BDH) assumption [29]. The latter is briefly reviewed next.

BDH Problem: Given $g, g^{\theta_1}, g^{\theta_2}, g^{\theta_3}$ for unknown $\theta_1, \theta_2, \theta_3 \in \mathbb{Z}_q$, compute $\hat{e}(g,g)^{\theta_1\theta_2\theta_3}$.

BDH Assumption: Let $\mathcal{B}$ be an algorithm which has advantage

$$\mathsf{Adv}(\mathcal{B}) = \Pr\left[\mathcal{B}(g, g^{\theta_1}, g^{\theta_2}, g^{\theta_3}) = \hat{e}(g,g)^{\theta_1\theta_2\theta_3}\right]$$

in solving the BDH problem. The BDH assumption states that $\mathsf{Adv}(\mathcal{B})$ is negligible for any polynomial-time algorithm $\mathcal{B}$.

Theorem 1: Suppose $\mathcal{A}$ is an type I adversary and wins the game in Section IV-A with advantage $\mathsf{Adv}(\mathcal{A})$ in time τ. Then there exists an algorithm to break the one-way property of the hash function.

Proof. Let $\mathcal{CH}$ be a challenger, $\mathcal{A}$ be a type I adversary who can win game 1 defined in Section IV-A. In the following, we show that if $\mathcal{A}$ can violate the user privacy property then $\mathcal{CH}$ can use $\mathcal{A}$ to break the one-way property of the hash function which is assumed to be hard.

Initial: $\mathcal{CH}$ first runs our Initialization to generate the system master-secret and the system parameters. The system parameters are sent to $\mathcal{A}$.

Training: $\mathcal{CH}$ answers $\mathcal{A}$'s queries defined in Section IV-A as in the real world. For instance, if a private key query is asked, then $\mathcal{CH}$ answers this query by using the algorithm in our Registration.

Output: At some point, $\mathcal{A}$ chooses a group id GID^* corresponding to $\mathbb{L}^*_{ID} = \{ID^*_1, ..., ID^*_x\}$ and $E^* = (u^*, \Lambda^*)$ and sends (GID^*, E^*) to $\mathcal{CH}$. $\mathcal{CH}$ randomly chooses a message m^* and sets w^* to be the trapdoor corresponding to m^*, where w^* is randomly chosen from $\mathbb{G}_1$. Finally, $\mathcal{A}$ outputs his guess $m^{*\prime}$. $\mathcal{A}$ wins the game if $m^* = m^{*\prime}$. In other word, if $\mathcal{A}$ break our scheme, then he has to break the one-way property of the hash function which is assumed to be hard.

Theorem 2: Suppose a type II adversary $\mathcal{A}$ who asks q_3 queries to H_3, q_t queries to Trapdoor with maximal group size N, and wins game 2 defined in Section IV-A with advantage $\mathsf{Adv}(\mathcal{A})$ in time τ. Then there exists an algorithm to solve the BDH problem with advantage

$$\frac{1}{Ne^2}(\frac{2}{2q_t+2})^2\mathsf{Adv}(\mathcal{A}).$$

Proof. Let $\mathcal{CH}$ be a challenger, $\mathcal{A}$ be a type II adversary who can win game 2 defined in Section IV-A. In the following, we show that if $\mathcal{A}$ can break our scheme, then $\mathcal{CH}$ can use $\mathcal{A}$ to solve the BDH problem (i.e., given $(g, g^{\theta_1}, g^{\theta_2}, g^{\theta_3})$ to compute $e(g,g)^{\theta_1\theta_2\theta_3}$) which is assumed to be hard.

Initial: $\mathcal{CH}$ sets $y = g^{\theta_1}$, then selects the system parameters

$$\mathsf{param} = (\mathbb{G}_1, \mathbb{G}_2, \hat{e}, g, y, H_1, H_2, H_3),$$

and gives param to $\mathcal{A}$. Assume at most L fog nodes can be enrolled by the TA. When the i-th fog node is first initialize, $\mathcal{CH}$ flips a coin $coin_i$ that yields 1 with probability δ and 0 with probability $1-\delta$. For the i-th fog node, we assume the identity of the node is ID_i.

Training: We treat H_1 and H_2 as random oracles which are controlled by $\mathcal{CH}$. $\mathcal{CH}$ answers $\mathcal{A}$'s queries as follows:

H_1 queries: $\mathcal{CH}$ maintains an initially empty list H_1^{list}. On input ID_i, $\mathcal{CH}$ does the following:

- If there is a tuple (ID_i, μ_i, f_i, s_i) on H_1^{list}, return f_i as the answer.
- Else, pick a random $\mu_i \in \mathbb{Z}_q^*$, set $f_i = g^{\mu_i}, s_i = y^{\mu_i}$, add (ID_i, μ_i, f_i, s_i) to H_1^{list} and respond with f_i.

H_2 queries: $\mathcal{CH}$ keeps an initially empty list H_2^{list}. On input (GID_i, m_i), $\mathcal{CH}$ does the following:

- If there is a tuple $(GID_i, m_i, \nu_i, w_i, H_2coin_i)$ on H_2^{list}, return w_i as the answer.
- Else flip a coin H_2coin_i that yields 1 with probability δ and 0 with probability $1-\delta$, randomly select $\nu_i \in \mathbb{Z}_q^*$ and do the following:
  - If $H_2coin_i = 1$, set $w_i = g^{\nu_i}g^{\theta_2}$, add $(GID_i, m_i, \nu_i, w_i, H_2coin_i)$ to H_2^{list} and return w_i as the answer.
  - Else, compute $w_i = g^{\nu_i}$, add $(GID_i, m_i, \nu_i, w_i, H_2coin_i)$ to H_2^{list} and return w_i as the answer.

H_3 queries: $\mathcal{CH}$ keeps an initially empty list H_3^{list}. On input Ω_i, $\mathcal{CH}$ does the following:

- If there is a tuple (Ω_i, b_i) on H_3^{list}, return b_i as the answer.
- Else randomly choose $b_i \in \{0,1\}^\ell$, add (Ω_i, b_i) to H_3^{list} and return b_i as the answer.

Private key queries: On input ID_i, $\mathcal{CH}$ first makes an H_1 query on ID_i, then recovers (ID_i, μ_i, f_i, s_i) from H_1^{list} and returns s_i as the answer.

Group public key queries: On input $GID_j = (ID_1||...||ID_k ||serial\ number)$, for $1 \le i \le k$, $\mathcal{CH}$ randomly chooses $\alpha_i \in \mathbb{Z}_q^*$ and does the following:

- If $coin_i = 0$, compute $u_i = g^{\alpha_i}$.
- Else, compute $u_i = g^{\alpha_i}g^{-\theta_1}$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2868920, IEEE Access

Finally, $\mathcal{CH}$ computes

$$E = (u = \prod_{i=1}^{k} u_i, \Lambda = \hat{e}(\prod_{i=1}^{k} H_1(ID_i), y)),$$

adds

$$(GID_j, u_1, ..., u_k, \alpha_1, ..., \alpha_k, E, \Lambda)$$

to G^{list}.

Trapdoor queries: $\mathcal{CH}$ manages a list T^{list} of the format (GID_l, m_l, T_l). On input (GID_j, m_j), $\mathcal{CH}$ first recovers

$$(GID_j, u_1, ..., u_k, \alpha_1, ..., \alpha_k, E, \Lambda)$$

from G^{list}. Assume

$$GID_j = (ID_1||...||ID_k||serial\ number).$$

$\mathcal{CH}$ asks an H_1 query on ID_i and finds

$$(ID_i, \mu_i, f_i, s_i)$$

on H_1^{list} for $1 \leq i \leq k$; asks an H_2 query on (GID_j, m_j) and recovers

$$(GID_j, m_j, \nu_j, w_j, H_2coin_j)$$

from H_2^{list}. If GID_j, m_j appears in a tuple (GID_j, m_j, T_j) on T^{list}, $\mathcal{CH}$ returns T_j as the answer. Otherwise, for $1 \leq i \leq k$, $\mathcal{CH}$ does the following:

- If $coin_i = 0$, use the Trapdoor algorithm to generate v_i, since $\mathcal{CH}$ has the knowledge of α_i and the private key corresponding to ID_i.
- Else if $H_2coin_i = 0$, compute $v_i = s_i u_i^{\nu_j}$.
- Else, abort (Event 1).

If $\mathcal{CH}$ does not abort, $\mathcal{CH}$ computes $T_j = \prod_{i=1}^{k} v_i$, adds (GID_j, m_j, T_j) to T^{list}.

Output: $\mathcal{A}$ chooses two keywords m_0^*, m_1^*, a group ID $GID^* = ID_1^*||...||ID_x^*||serial\ number^*$ and group public key (E^*, Λ^*) and sends $(GID^*, E^*, \Lambda^*, m_0^*, m_1^*)$ to $\mathcal{CH}$. $\mathcal{CH}$ randomly chooses $\xi \in \{0,1\}$, $\Omega^* \in \{0,1\}^\ell$ and sends (g^{θ_3}, Ω^*) to $\mathcal{A}$. Finally, $\mathcal{A}$ outputs his guess $\xi' \in \{0,1\}$.

If $\xi' = \xi$, $\mathcal{CH}$ recovers the tuple $(GID^*, u_1^*, ..., u_x^*, \alpha_1^*, ..., \alpha_x^*, E^*, \Lambda^*)$ from G^{list} and (ID_l, μ_l, f_l, s_l) from H_1^{list} for $l \in [1, x]$. $\mathcal{CH}$ asks an H_2 query on (GID^*, m_ξ^*) and recovers $(GID^*, m_\xi^*, \nu_\xi^*, w_\xi^*, H_2coin_\xi^*)$ from H_2^{list}. Assume $coin_l^*$ is corresponding to ID_l^*, $l \in [1, x]$. It requires that only one of the coins equals to 1. $\mathcal{CH}$ picks a random pair (Ω_i, b_i) from the H_3^{list} and outputs

$$\Omega_i / (e(g^{\theta_3 \sum_{i=1}^{x} \mu_i^*}, y) e(u^{*\nu_\xi^*}, g^{\theta_3}) e(g^{\theta_2 \sum_{i=1}^{x} \alpha_i^*}, g^{\theta_3}))$$

as the answer of the BDH problem. Next, we show the probability for $\mathcal{CH}$ to get the right solution of the BDH problem.

If $\mathcal{CH}$ does not abort, then $\mathcal{A}$ cannot find any inconsistency between the simulation and the real world. Hence, $\Pr[\xi' = \xi] \geq \mathsf{Adv}(\mathcal{A})$. It is easy to see that $\mathcal{CH}$ will abort if Event 1 happens. It can be seen that

$$\Pr[\neg \text{Event 1}] \geq ((1-\delta)(1-N\delta))^{q_t}.$$

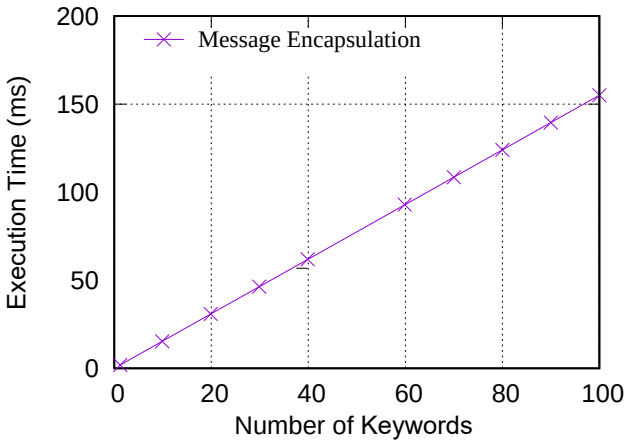


FIGURE 3. Execution Time of the Message Encapsulation Phase.

Next we show the probability for $\mathcal{CH}$ to output the solution of the BDH problem. For $\mathcal{CH}$ to solve the BDH problem, it requires that $coin_l^* = 1$ for an index $l \in [1, x]$ and $H_2coin_\xi^* = 1$. It is easy to see that the probability is at least $N\delta^2$. Overall, we have the probability for $\mathcal{CH}$ to solve the BDH problem is

$$\begin{aligned} & \frac{1}{q_{H_3}} N\delta^2((1-\delta)(1-N\delta))^{q_t}\mathsf{Adv}(\mathcal{A}) \\ \geq\ & \frac{1}{q_{H_3}} N\delta^2(1-N\delta)^{2q_t}\mathsf{Adv}(\mathcal{A}) \\ \geq\ & \frac{1}{q_{H_3} Ne^2}(\frac{2}{2q_t+2})^2\mathsf{Adv}(\mathcal{A}). \end{aligned}$$

V. PERFORMANCE EVALUATION

In this section, we evaluate the efficiency of our scheme by using cryptographic library MIRACL [37]. The simulations were performed on a Linux machine using an Intel Core i7-4790 at a frequency of 3.6 GHz. We selected an SSP curve with 80 bits security level. To the best of our knowledge, there is no privacy-preserving resource allocation considers that considers meta-data privacy in fog computing. In particular, our scheme considers the security requirements specified in Section II-B for the first time. Therefore, we only evaluate the efficiency of the proposed scheme. Further, in the following, we will only evaluate the efficiency of our Message Encapsulation phase and the efficiency of the Test algorithm in our Resource Allocation phase, since other phases only need to be performed once.

Figure 3 shows the efficiency of our Message Encapsulation phase. As shown in Figure 3, the computational cost grows linearly as the number of keywords to be encrypted grows. For a single keyword, the execution time is about 1.55 ms. Usually, only several encrypted keywords will be included in encrypted meta-data. Thus, the total execution time to generated encrypted meta-data will not be large.

Figure 4 shows the execution time of the Test algorithm in our Resource Allocation phase in the worst case. As discussed above, usually, only several encrypted keywords

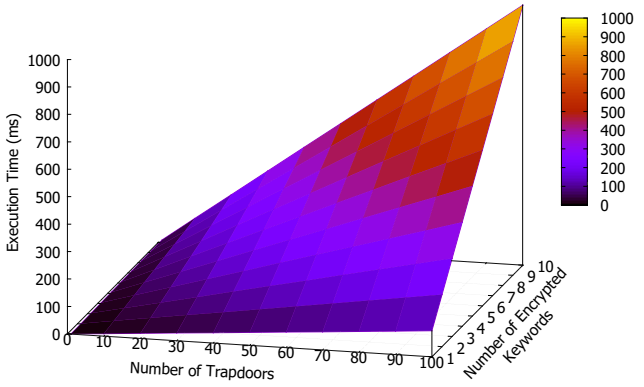


FIGURE 4. Execution Time of the Test Algorithm.

will be included in encrypted meta-data. Therefore, in our simulation, we assume the number of encrypted keywords in encrypted meta-data ranges form 1 to 10. Further, typically, an end device will only allow a smart gateway to search for a limited number of keywords. In our simulation, the number of trapdoors issued to a gateway is chosen from 1 to 100.

VI. CONCLUSION

Resource allocation is one of the key techniques to optimize the utilization of the resources of fog nodes. In this paper, we have investigated the security challenges in existing resource allocation techniques for fog computing and proposed a concrete privacy-preserving resource allocation scheme for fog computing. Formal security analysis have showed that our scheme satisfies user privacy and full key compromise resistance. Besides, our scheme has constant message expansion.

REFERENCES

[1] IoT, from Cloud to Fog Computing. Blogs@Cisco - Cisco Blogs. Https://blogs.cisco.com/perspectives/iot-from-cloud-to-fog-computing.

[2] L. Zhang, X. Meng, K.R. Choo, Y. Zhang, and F. Dai, "Privacy-preserving cloud establishment and data dissemination scheme for vehicular cloud", IEEE Transactions on Dependable and Secure Computing, DOI: 10.1109/TDSC.2018.2797190.

[3] L. Zhang, Q. Wu, J. Domingo-Ferrer, B. Qin, C. Hu, "Distributed aggregate privacy-preserving authentication in VANETs", IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 3, pp. 516-526, 2017.

[4] P. Hu, H. Ning, T. Qiu, Y. Zhang, and X. Luo, "Fog computing based face identification and resolution scheme in internet of things", IEEE Transactions on Industrial Informatics, vol. 13, no. 4, pp.1910-1920, 2017.

[5] B. Tang, Z. Chen, G. Hefferman, S. Pei, T. Wei, H. He, and Q. Yang, "Incorporating intelligence in fog computing for big data analysis in smart cities", IEEE Transactions on Industrial Informatics, vol. 13, no. 5, pp.2140-2150, 2017.

[6] L. Xu, W. He, and S. Li, "Internet of things in industries: A survey", IEEE Transactions on industrial informatics, vol. 10, no. 4, pp.2233-2243, 2014.

[7] L. Zhang, C. Hu, Q. Wu, J. Domingo-Ferrer, B. Qin, "Privacy-preserving vehicular communication authentication with hierarchical aggregation and fast response", IEEE Transactions on Computers, vol. 65, no. 8, pp. 2562-2574, 2016.

[8] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (IoT): A vision, architectural elements, and future directions", Future generation computer systems, vol. 29, no. 7, pp.1645-1660, 2013.

[9] J. Li, L. Zhang, J. Liu, H. Qian, Z. Dong, "Privacy-preserving public auditing protocol for low performance end devices in cloud", IEEE Transactions on Information Forensics and Security, vol. 11, no. 11, pp. 2572-2583, 2016.

[10] J. Liu, J. Li, L. Zhang, F. Dai, Y. Zhang, X. Meng, and J. Shen, "Secure intelligent traffic light control using fog computing", Future Generation Computer Systems, vol. 78, part 2, pp. 817-824, 2018.

[11] Y. Lai, F. Yang, L. Zhang, Z. Lin, "Distributed public vehicle system based on fog nodes and vehicular sensing", IEEE Access, vol. 6, pp. 22011-22024, 2018.

[12] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics", In Big Data and Internet of Things: A Roadmap for Smart Environments, 2014, pp. 169-186.

[13] A. Kapsalis, P. Kasnesis, I.S. Venieris, D.I. Kaklamani, and C. Patrikakis, "A cooperative fog approach for effective workload balancing", IEEE Cloud Computing ,vol. 4, no. 2, pp.36-45, 2017.

[14] J. Xu, B. Palanisamy, H. Ludwig, and Q. Wang, "Zenith: Utility-aware resource allocation for edge computing", In Proc. IEEE International Conference on Edge Computing (EDGE), 2017, pp. 47-54.

[15] M. Aazam, and E. Huh, "Fog computing and smart gateway based communication for cloud of things", In Proc. International Conference on Future Internet of Things and Cloud (FiCloud), 2014, pp. 464-470.

[16] M. Chen, and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network", IEEE Journal on Selected Areas in Communications, vol. 36, no. 3, pp.587-597, 2018.

[17] M. Chen, Y. Hao, L. Hu, M.S. Hossain, and A. Ghoneim, " Edge-CoCaCo: Toward joint optimization of computation, caching, and communication on edge cloud", IEEE Wireless Communications, vol. 25, no. 3, 2018.

[18] Y. Qian, M. Chen, J. Chen, M.S. Hossain, and A. Alamri, "Secure enforcement in cognitive internet of vehicles", IEEE Internet of Things Journal, vol. 5, no. 2, pp.1242-1250, 2018.

[19] U. Hunkeler, H. Truong, and A. Stanford-Clark, "MQTT-S—A publish/subscribe protocol for wireless sensor networks", In Proc. 3rd IEEE/Create-Net International Conference on COMmunication System softWAre and MiddlewaRE (COMSWARE), 2008, pp. 791-798.

[20] C. Prazeres, and M. Serrano, "Soft-iot: Self-organizing fog of things", In Proc. International Conference on Advanced Information Networking and Applications Workshops (WAINA), 2016, pp. 803-808.

[21] M. Chen, Y. Hao, Y. Li, C. Lai, and D. Wu, "On the computation offloading at ad hoc cloudlet: architecture and service modes", IEEE Communications Magazine, vol. 53, no. 6, pp.18-24, 2015.

[22] H. Hong, P. Tsai, and C. Hsu, "Dynamic module deployment in a fog computing platform", In Proc. 18th Asia-Pacific Network Operations and Management Symposium (APNOMS), 2016, pp. 1-6.

[23] V. Souza, W. Ramirez, X. Masip-Bruin, E. Marin-Tordera, G. Ren, and G. Tashakor, "Handling service allocation in combined fog-cloud scenarios", In Proc. IEEE International Conference on Communications (ICC), 2016, pp. 1-5.

[24] M. Aazam and E. Huh, "Dynamic resource provisioning through fog micro datacenter", In Proc. IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops), 2015, pp. 105-110.

[25] M. Chen, Y. Qian, J. Chen, K. Hwang, S. Mao, and L. Hu, "Privacy protection and intrusion avoidance for cloudlet-based medical data sharing", IEEE Transactions on Cloud Computing, to be published, doi: 10.1109/TCC.2016.2617382.

[26] X. Liu, R.H. Deng, Y. Yang, H.N. Tran, and S. Zhong, "Hybrid privacy-preserving clinical decision support system in fog-cloud computing. Future Generation Computer Systems", vol. 78, pp.825-837, 2018.

[27] X. Liu, R. Deng, K. Choo, Y. Yang, and H. Pang, "Privacy-preserving outsourced calculation toolkit in the cloud", IEEE Transactions on Dependable and Secure Computing, DOI: 10.1109/TDSC.2018.2816656.

[28] D. Song, D. Wagner, and A. Perrig, (2000). "Practical techniques for searches on encrypted data", In Proc. IEEE Symposium on Security and Privacy (S&P), 2000, pp. 44-55.

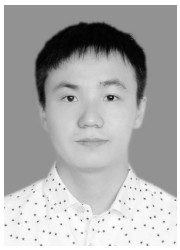
[29] D. Boneh, G. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search", In Proc. International conference on the theory and applications of cryptographic techniques (Eurocrypt), 2004, pp. 506-522.

[30] Y. Hwang, and P. Lee, "Public key encryption with conjunctive keyword search and its extension to a multi-user system", In Proc. International Conference on Pairing-Based Cryptography (Pairing), 2007, pp. 2-22.

[31] S. Zhang, Y. Mu, and G. Yang, "Threshold broadcast encryption with keyword search", In Proc. International Conference on Information Security and Cryptology (ICISC), 2015, pp. 322-337.

[32] L. Zhang, Q. Wu, J. Domingo-Ferrer, B. Qin, Z. Dong, "Round-efficient and sender-unrestricted dynamic group key agreement protocol for secure group communications", IEEE Transactions on Information Forensics and Security, vol. 10, no. 11, pp. 2352-2364, 2015.
[33] B. Liu, L. Zhang, "An improved identity-based batch verification scheme for VANETs", In Proc. International Conference on Intelligent Networking and Collaborative Systems (INCos 2013), 2013, pp. 809-814.
[34] L. Zhang, "OTIBAAGKA: A new security tool for cryptographic mix-zone establishment in vehicular ad hoc networks", IEEE Transactions on Information Forensics and Security, vol. 12, no. 12, pp. 2998-3010, 2017.
[35] J. Li, L. Zhang, "Sender dynamic, non-repudiable, privacy-preserving and strong secure group communication protocol", Information Sciences, vol. 414, pp. 187-202, 2017.
[36] P. Rogaway and T. Shrimpton, "Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance", In Proc. International Workshop on Fast Software Encryption (FSE), 2004, pp. 371-388.
[37] L. Zhang, Q. Wu, A. Solanas, J. Domingo-Ferrer, "A Scalable Robust Authentication Protocol for Secure Vehicular Communications", IEEE Transactions on Vehicular Technology, vol. 59, no. 4, pp. 1606-1617, 2010.

LEI ZHANG received his Ph.D. degree from Universitat Rovira i Virgili, Tarragona, Spain, in 2010. Since then, he has been with Universitat Rovira i Virgili, Tarragona, Spain, as a Postdoctoral Researcher. He is a full Professor with the School of Computer Science and Software Engineering, East China Normal University, Shanghai, China. He has been a holder/co-holder of more than 10 China/Spain funded (key) projects. His fields of activity are information security, VANET security, cloud/fog security, cryptography, and data privacy. He has authored over 70 publications. He is the editors of several international journals, and, was the Guest Editor of Future Generation Computer Systems. He has served in the program committee of more than 40 international conferences in information security and privacy.



JIANGTAO LI received his Ph.D. degree from East China Normal University, Shanghai, China. He received his B.S. degree (awarded outstanding graduates) in mathematics and applied mathematics from Henan Normal University, China. His research interests include information security, public key cryptography and network security.