Received February 21, 2020, accepted March 16, 2020, date of publication March 30, 2020, date of current version April 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2984317

Hybrid Lightweight Proxy Re-Encryption Scheme for Secure Fog-to-Things Environment

OSAMA A. KHASHAN
College of Computing and Informatics, Saudi Electronic University, Riyadh 11673, Saudi Arabia
e-mail: o.khashan@seu.edu.sa

ABSTRACT Fog computing is a promising paradigm that can mitigate the heavy burden on cloud-central processing of the vast amount of IoT data. Although fog computing has the advantages of low latency, storage, and computing resources that serve IoT applications and things, it hardly suffers from security and privacy challenges. Proxy re-encryption (PRE) is an effective cryptographic solution to ensure the security of fog-to-things communication. However, in PRE schemes, the problem of a processing delay caused by offloading significant computational load to the proxy for re-encryption, and the heavy computation operations of data owner encryption and user decryption due to asymmetric cryptographic use, are still unresolved in the literature. In this paper, we propose a hybrid proxy re-encryption scheme that combines lightweight symmetric and asymmetric encryption algorithms to establish secure communications in fog-to-things computing. In the proposed scheme, the computational cost incurred by fog nodes to carry out the re-encryption process is highly efficient. Meanwhile, the scheme can reduce the encryption and decryption overheads for end-users with resource-constrained devices. Security and performance analyses are conducted, and the results indicate that our scheme is secure, highly efficient, and lightweight.

INDEX TERMS Lightweight cryptography, Internet-of-things, Fog computing, Proxy re-encryption, ECC.

I. INTRODUCTION

The Internet of Things (IoT) is an intelligent network consisting of a massive number of heterogeneous devices connected to and communicating with each other over a network. The "Things" in IoT refers to the smart devices that have limited computing capabilities, with the ability to automatically sense and react to the working environment. IoT has greatly revolutionized our lifestyle by offering us the convenience and flexibility through innovative applications, such as smart homes, smart cities, smart vehicles, health-monitoring systems, and many other IoT applications [1].

IoT is growing rapidly, with billions of devices anticipated to be connected in the coming years [2]. With this significant growth of IoT, huge volumes of data are being generated and exchanged between IoT devices for real-time analytics. On the other hand, most IoT devices are resource-constrained devices characterized by low processing power, restricted memory capacity, limited battery life, and low communication capabilities [3].

The associate editor coordinating the review of this manuscript and approving it for publication was Minho Jo.

Despite the wide adoption of IoT technology by society and business, the big data generated has triggered a demand to evolve the architecture of data processing, communication, and storage, and to address other challenges lying ahead in the development of IoT [4].

Cloud computing can provide on-demand services and enable cloud users to relieve the high storage and computation burden locally. However, the prevalence of IoT applications has posed a challenge to centralized cloud computing. The huge volume of data generated by IoT applications that stored in cloud servers has resulted in unbearable transmission latency, which has degraded the quality of services to end-users [5]. In addition, real-time applications, such as e-health, smart cities, etc., require a significant of communication resources that degrade communication performance. These applications require predictable and low communication latency to meet the real-time decision requirements of IoT applications [6].

To mitigate these issues, the fog computing paradigm is introduced. In fog computing, fog nodes are located between the cloud server and IoT objects, as shown in Fig. 1. The distribution of fog devices can also address the scalability of the cloud by reducing central processing and communications.

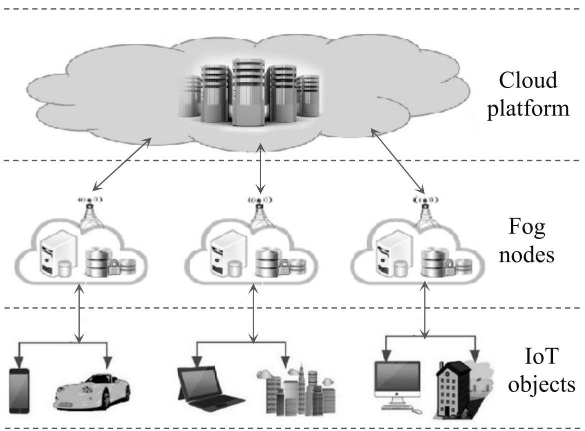

FIGURE 1. Cloud, fog, IoT devices framework.

Compared to the cloud, fog nodes are much closer to end devices in the edge network and have lower latency [7]. Fog computing can reduce the response time for a user by about 20%, and the data traffic between the edge and cloud network by 90%. It can also decrease the response delay of real-time applications by 50% [24].

Although fog computing is promising for efficiently serving IoT applications and things, data security is a critical challenge [9]. The issues of risk become even more critical when low-resource devices exchange sensitive data because of their direct access to the internet [10]. Most of the research on fog computing and IoT focuses on architecture issues and applications development without properly supporting security and privacy-preserving mechanisms [23]. When sensitive data are outsourced to fog nodes, data owners lose physical control over their data and, hence, become vulnerable to various security and privacy attacks, as fog devices are not considered fully trusted.

To tackle the IoT security and privacy challenges, many security operations can be carried out while transmitting, storing, and retrieving data in fog computing. Nevertheless, implementing complex and resource-intensive security functions to resource-constrained IoT devices is infeasible. It usually leads to high computational overhead, limits the devices' lifetimes, and exhibits significant latency for time-sensitive applications that require immediate response [11]. This can be overcome by offloading the security functions to the fog nodes in place of resource-constrained IoT nodes. As encryption is considered the most effective mechanism for protecting data confidentiality and access control, it would be efficient to offload cryptographic operations to the fog nodes of the network. However, fog nodes should also offer flexible, controllable, and lightweight data sharing and fine-grained access control.

Proxy re-encryption (PRE) realizes the efficient sharing of the encrypted data by allowing a semi-trusted proxy to automatically convert a ciphertext encrypted with a delegator's public key into another ciphertext of the same message under a delegatee's public key. The proxy performs the transformation by using a re-encryption key generated by the delegator, and without revealing the plain text or private keys to the proxy. Thus, each fog node is treated as a proxy node to perform the heavy computations involved in the re-encryption process in place of the constrained IoT devices and leaves them with less computation to decrypt their relevant ciphertexts.

However, in PRE, sharing outsourced data with a large number of users may induce an expensive overhead for rekeying and re-encrypting the data between users. Further, offloading too much computational load to a proxy may introduce an extra delay in the re-encryption process. This, in turn, could cause an efficiency problem for real-time IoT applications that require real-time communication and data sharing [13].

Existing research works have focused on addressing the latency problem by offloading the greatest possible amount of computational cost to the proxy. On the other hand, most of PRE schemes employ asymmetric ciphers to encrypt and decrypt all communicated data between the nodes and the fog server. Nevertheless, asymmetric encryption is relatively slow and computationally intensive as compared to symmetric encryption [14]. Therefore, it is unfeasible to encrypt a large amount of IoT data using asymmetric algorithms. Symmetric encryption can utilize fewer resources with low computational complexities and its security is high as long as the key is kept secret [15].

A. OUR CONTRIBUTIONS

In this paper, we make the following main contributions:

- We propose a secure data communication scheme based on the proxy re-encryption for the fog computing environment. The proposed scheme combines hybrid lightweight symmetric and asymmetric cryptographic algorithms that require less computation time in dealing with encryption and decryption processes for resource-limited end user nodes.
- We demonstrate that the proposed scheme addresses the latency limitation of existing PRE schemes that offload much computing overhead to proxy nodes. We show that our scheme can achieve less computational cost with nearly constant-time for fog nodes to re-encrypt ciphertexts as compared to existing PRE schemes that incur much computational overhead, with computing times increase linearly as message sizes increase.
- We provide security and performance analyses of our scheme. The results show that our scheme is secure, confidential, and collusion safe. The results also show that our scheme is very lightweight and suitable for real-time IoT applications.

B. PAPER ORGANIZATION

The remainder of our paper is structured as follow. Section 2 reviews some works related to our approach. Section 3 provides some of the background and main properties of a lightweight cryptography. Section 4 explains our proposed scheme and discusses the security analysis.

Section 5 discusses the results and performance evaluation. Finally, the conclusion is drawn in Section 6.

II. RELATED WORK

In this section, we review some works related to our proposed scheme. Variants of PRE schemes have been presented in the literature since Blaze et al introduced it in 1998 [12]. These schemes can be broadly divided into two categories, Unidirectional (i.e., attribute-based PRE, conditional PRE, identity-based PRE, and time-based PRE) and Bidirectional (i.e., type-based PRE, and threshold-based PRE) schemes [37]. Various properties can be guaranteed by the proxy re-encryption, including unidirectionality, non-interactivity, non-transitivity, non-transferability, proxy invisibility, collusion resistance, key optimality, and original access, which have been extensively discussed in the literature [21]. Depending on the combination of these properties, more recent PRE schemes with different features have been developed, in which the evaluation of a PRE scheme is performed based on the presence of these properties.

Attribute-based PRE is presented in [31]–[33] to provide fine-grain access control over outsourced data by allowing a proxy to transform a ciphertext under a set of attributes into another ciphertext under another set of attributes. A conditional PRE (C-PRE) was introduced in [29], in which only ciphertexts that satisfy a certain condition can be transformed by the proxy. The authors in [30] introduced a special kind of C-PRE to enable a ciphertext to be transformed by a proxy only from a specified sender. This gives the delegator exclusive rights to authorize the delegation. The authors in [35] defined identity-based PRE (IB-PRE), in which a ciphertext under the delegator's identity is transformed into another ciphertext under the delegatee's identity. Later, the authors in [41] proposed IB-PRE without random oracles. A further proposed scheme in [36] extended the IB-PRE to support a conditional re-encryption property, which makes it secure against condition and identity chosen-ciphertext attacks. Type-based PRE is proposed by [39], in which each of the delegator's ciphertexts is combined with a type, and a proxy re-encryption is considered only if a delegatee has the identical type in his public key. This helps the data owner achieve fine-grained delegation control. A recent scheme in [40] combined the concept of the PRE with certificate-based encryption and proposed a certificate-based PRE that helped resist the adaptive chosen ciphertext attack. Another proposed PRE scheme in [34] is based on a keyword search, in which the ciphertext is re-encrypted if it contains a keyword match with some information related to the re-encryption key. The authors in [18] presented a broadcast PRE that enables the delegator to delegate the decryption right to a group of users at a time. A further proposed scheme in [38] is based on conditional broadcast PRE to dynamically add users to the sharing group without the need to change the encryption public key.

However, most of the existing schemes for secure data communication and sharing in cloud computing suffer from a limitation, which is the delay lag between user request and cloud response due to the huge augmentation of outsourced data and cryptographic operations of expansive numbers of connected users [27], [44].

Fog computing aims to address issues related to the cloud by moving some tasks to the edge of the network. It can be used to perform heavy computation operations and reduce the computational overhead required on resource-constrained devices. Nonetheless, few works have focused on security and privacy in fog computing [23].

The authors in [24] have discussed the security challenges, threats, and requirements of fog computing. The schemes introduced in the literature based on PRE for fog computing have adopted technologies similar to those that have been introduced for cloud environments. Literature [25]–[27] has applied ciphertext-policy attribute-base encryption (CP-ABE) to establish secure access control over encrypted data in fog computing, thereby enabling a data owner to specify the access policy over a universe of attributes that a user needs to decrypt the ciphertext. A further proposed scheme in [23] uses identity-based cryptography to provide secure communication through unauthorized users in fog. The authors in [43] presented an ID-based PRE scheme with auxiliary input. The scheme was shown to resist the leakage of secret keys caused by side channel attacks in fog computing.

The significant drawback of the use of ABE, CP-ABE, and ID-based cryptography in fog computing is the computational cost in the decryption phase, which includes several pairing operations that are accompanied by the complexity of policy [26], [27]. On the other hand, fog computing involves a large number of dynamic and concurrent communications among IoT devices. Due to resource limitations of IoT/Fog networks, the use of traditional key management and public key cryptographic primitives to secure the information exchange among the connected devices failed to support fog computing. Traditional cryptographic mechanisms are computationally too expensive for most objects and do not satisfy the real-time operation requirements of IoT [44].

It has been proved that lightweight cryptography is more compatible for work in a fog environment [4]. The authors in [44], [45] presented PRE schemes based on symmetric ciphers. The main disadvantage of these schemes is that they rely on the assumption of an individual pre-shared secret key with a high risk of key exposure. These schemes require a trusted distribution of symmetric keys to establish secure communication. To tackle the above limitation, the authors in [4] presented a PRE scheme that used lightweight asymmetric encryption based on elliptic curve cryptography (ECC). It was proved that the scheme provided efficient outsourcing security in fog computing. Nonetheless, sharing outsourced data with a large number of simultaneous users will lead to the offloading of too much computational load to the proxy, which, in turn, will introduce a large computational cost to the proxy, with an extra delay in response time to IoT objects.

Motivated by the issues mentioned above, we propose a PRE scheme for secure fog-to-things communication.

According to the proposed scheme, the IoT data is encrypted and decrypted using symmetric encryption. Moreover, the fog node will concentrate only on the re-encryption of a symmetric key associated with the corresponding ciphertext. We state that our approach greatly saves on the computation cost of the encryption, re-encryption, and decryption processes, which makes it more suited for a large IoT data size (i.e., multimedia data), and for the large number of concurrent delegatees.

III. LIGHTWEIGHT BASED CRYPTOGRAPHIC SOLUTION

In this section, we first specify the general security and performance properties of lightweight cryptography. Then, we present the functional description of the XXTEA symmetric cipher [28] followed by the ECC asymmetric cipher in terms of supported security properties and performance.

A. SECURITY AND PERFORMANCE PROPERTIES

With numerous IoT devices having limited computational capabilities, one of the major considerations for an efficient cryptographic scheme in fog computing is that it is lightweight. The use of conventional encryption algorithms stresses the resources of both fog and IoT nodes. Such resources include processing power, memory size, power consumption, and energy. Lightweight ciphers require fewer resources and can realize a good trade-off between security and performance.

Generally, performance can be expressed in terms of latency or waiting time, power consumption, and throughput. The high performance in lightweight cryptographic algorithms can be realized by implementing the following:

- *Smaller block sizes* (64 or 80 bits), when a small block size is used, it limits the size of the plaintexts to be encrypted.
- *Smaller key sizes* (80-bit or less), where a smaller key size would lead to higher efficiency and minimize power consumption.
- *Simpler rounds*, the functions carried out in each round of lightweight ciphers are comparatively simpler than those used in the conventional encryption ciphers. The larger number of rounds degrades the performance.
- *Simpler key schedules*, in which for a given key, a key schedule is a kind of algorithm that calculates the subkeys for rounds.

Many encryption algorithms in the literature have been shown to achieve high robustness (i.e., secure against common attacks). However, many other ciphers registered attacks on most rounds published on them; their use certainly comes at a high risk. Hence, as each cryptographic algorithm has its own distinct feature, it is important to choose an efficient encryption cipher in all aspects. At the same time, the encryption algorithm should be able to provide adequate protection against attacks in fog-to-things computing.

B. XXTEA LIGHTWEIGHT CRYPTOGRAPHY

The Corrected Block Tiny Encryption Algorithm (XXTEA) is a lightweight block cipher that was designed to improve upon its predecessor, XTEA. The XXTEA is an unbalanced Feistel network cipher that operates on blocks of arbitrary size with multiple words of 32 bits, using a key length of 128 bits. The internal structure consists of simple operations of shifts, additions, and XORs. The block can be viewed as a circular array. A single full cycle in XXTEA consists of looping through the block words, a round function that involves both immediate neighbors to each word in the block, a full cycle number, and the key. A single round for a fixed-size block can be briefly described as $vr \leftarrow vr + F(vr-1, vr+1, r, k)$. A full cycle includes n rounds, where n is the number of words in the block. The number of full cycles involved in the block equals $6 + 52/n$. Further, the generated encrypted file can be the same exact size as the original file.

XXTEA has the property that eliminates the need for using the mode of operation on messages larger than the block size, which can be applied directly to encrypt the entire message. A single bit changed in XXTEA will lead to a change in half of the bits of the entire block, which eliminates the possibility of cut and paste attacks [16]. XXTEA consumes only a small amount of computational cost and memory. Based on a study in [17], XXTEA consumes about 42% and 70% less, in terms of computation and memory consumption, compared to AES-128 and AES-256, respectively. Considering the features provided by XXTEA, which is gentler on resources and requires much less memory and shorter execution time, with a reasonable security level, it is an ideal candidate for resource-constrained fog-to-things networks.

C. ELGAMAL ECC LIGHTWEIGHT CRYPTOGRAPHY

In 1985, T. ElGamal [22] introduced the concept of the public key encryption approach based on the discrete logarithm problem. The concept is based on hiding the message m using α^{k} and, β^{k} where $\beta = \alpha^{\mathrm{a}}$, α is a primitive root of a large prime p, and k is a random integer. The parameters (α, β, p) are public. The sender uses k to compute $(\alpha^{\mathrm{k}}, \beta^{\mathrm{k}})$ and send them to the receiver, which uses a secret a to retrieve the message m using $m = \left(\alpha^{\mathrm{k}}\right)^{-\mathrm{a}} * \left(\beta^{\mathrm{k}} m\right) = \left(\alpha^{\mathrm{a}}\right)^{-\mathrm{k}} * \left(\beta^{\mathrm{k}} m\right) = \left(\beta^{-\mathrm{k}}\right) * \left(\beta^{\mathrm{k}} m\right)$ [47].

ECC was presented independently by Koblitz [48] and Miller [49] in 1985, and introduced the implementation of public key encryption using elliptic curve groups of arithmetic over a finite field. Later, the elliptic curve groups over a finite field were used to implement ElGamal's public key cryptosystem and producing ElGamal based-ECC, which is described well in [19], [46].

The ECC cryptographic operation can only encrypt and decrypt a point on the curve over finite fields and not messages. Thus, converting a message to a point (encoding), converting a point to a message (decoding), and public key validation are important cryptographic functions in ECC. A relatively small key size (i.e., 160 bits) is used in ECC, however, the security is comparable with those of other public key cryptosystems, such as DSA and RSA, which use a much bigger key (i.e., 1024 bits). With a smaller key, ECC can offer faster implementation, lower memory consumption, and a higher throughput rate [19]. Furthermore, the security of ECC depends on the difficulty of the elliptic curve discrete logarithm problem.

An elliptic curve is represented over a non-singular cubic polynomial equation with two unknowns over the finite field G. $\mathbb{F}_p$ denotes the field of integers modulo the prime number p. The elliptic curve E over $\mathbb{F}_p$ is defined in an equation of the form $y^2 = x^3 + ax + b(modp)$, where $a, b \in \mathbb{F}_p$, and $\neq 4a^3 + 27b^2 0(modp)$. Here, the elements of the finite field are integers between 0 and $p - 1$, where p is chosen so that there is a finitely large number of points on the elliptic curve to make a cryptosystem secure [46]. A pair (x, y), where $x, y \in \mathbb{F}_p$, is a point on the curve if it satisfies the above equation together with the element at infinity O. If P, Q, and R are points on $\mathbb{F}_p(a, b)$, in which $P = (x_1, y_1)$, $Q = (x_2, y_2)$, and $P, Q \neq \infty$; then, $P + Q = R = (x_3, y_3)$, where for each different a and b value, a different elliptic curve is created. For ECC key generation, let P be a point in $E(\mathbb{F}_p)$ that has the prime order n, the cyclic subgroup of $E(\mathbb{F}_p)$ is $P = \{\infty, P, 2P, 3P, \ldots, (n-1)P\}$, in which $\{p, E, P, n\}$ are public domain parameters. A private key d is generated as a random number in the interval $[1, n-1]$. The corresponding public key Q is calculated as $Q = dP$. The addition of points $P(x_1, y_1)$ and $Q(x_2, y_2)$ in ECC to yield $R(x_3, y_3)$ can be algebraically calculated as $x_3 = m^2 - 2x_1 modp$, and $y_3 = -y_1 + s(x_1 - x_3) modp$, while m is the slope of the line expressed as $m = 3x_1^2 + a)/2y_1)$. The common way to express the multiplication of a point in an elliptic curve is by point doubling (i.e., $P + P = 2P = R$).

The ElGamal ECC is the analog of the ElGamal cryptosystem. The multiplication operations are replaced by addition, whilst exponentiation is replaced by multiplication. The basic cryptographic operations in the ElGamal ECC are performed as follows. The parameters (α, β, p) and the curve C are made public, in which α and β are points on the elliptic curve. The sender breaks the message m into smaller blocks, encodes each block as an integer modulo the prime number p, and then uses a random k to compute two points on the curve using $s(x_3, y_3) = x_3, \mathrm{k} * \alpha)$ and $w(x_4, y_4) = (x_4, m + \mathrm{k} * \beta)$, where the pair (s, w) is sent to the receiver. Then, the receiver decrypts the message using $m = y_4 - (\mathrm{a} * y_3) = (m + k * \beta) - \mathrm{a} * (k * \alpha) = m + \mathrm{a} * k * \alpha - \mathrm{a} * k * \alpha$ [47].

IV. PROPOSED HYBRID LIGHTWEIGHT PROXY RE-ENCRYPTION SCHEME

In this section, we introduce the system model for the proposed fog-based PRE scheme. Then, we provide a concrete construction of the security model used in this work, which is followed by the security analysis.

A. SYSTEM DEFINITION

Our proposed scheme is based on proxy re-encryption that enables semi-trusted fog servers to forward messages between things. In this paper, we implement a hybrid encryption approach using lightweight symmetric and asymmetric

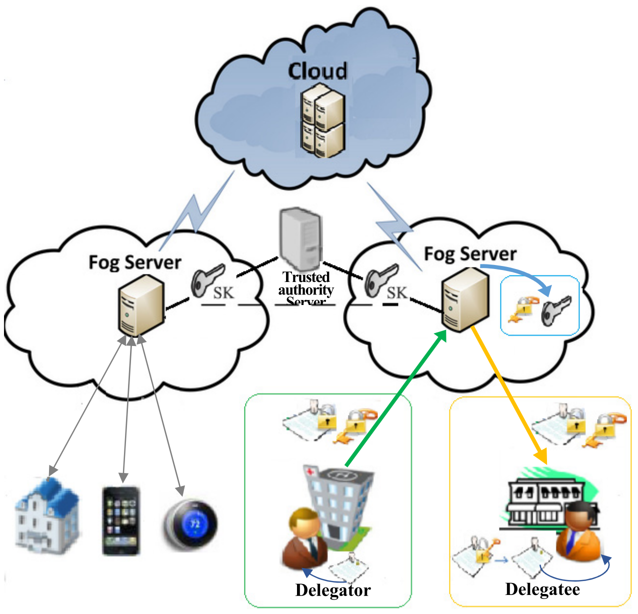

FIGURE 2. Description of the proposed fog-based proxy re-encryption scheme.

encryption ciphers that are both commonly accepted for use with lightweight devices. We illustrate our proposed scheme in figure 2. The system consists of users connected to fog nodes, each with a unique identifier registered to the fog nodes. Each user is represented by a public and private key pair. The system includes a set of fog nodes that act as proxy servers enabling connection and communication between users and servers. Because fog computing is a distributed architecture, one specific fog node can be identified as a fully trusted authority node that is in charge of handling users' registration, generating public system parameters as well as secret keys for the fog node and other connected users. In addition, the trusted authority node is not involved in any step of PRE, and is responsible for securely storing the secret keys.

The system consists of four phases: Key generation, Delegator encryption, Fog re-encryption, and Delegatee decryption.

In the Key generation phase, the public system parameters and the secret keys are generated and sent to the involved parties. For a particular delegation, the delegator (user A) first interacts with a nearby fog to communicate and share data with a specified delegatee (user B). Then, the fog node communicates with the trusted authority node to generate the re-encryption key and send it securely to the concerned fog server, then send the secret keys to both the delegator and the delegatee.

The Delegator encryption phase is composed of two major steps: symmetric encryption of data, and asymmetric encryption for the symmetric cipher key. In the hybrid encryption routine, a thing first generates a random integer value used as a key for the symmetric cipher. Then, the symmetric key is used to encrypt the delegator's message using the symmetric encryption cipher. Subsequently, the symmetric key is encrypted using the asymmetric cryptographic cipher, which is then attached along with the ciphertext transmitted to the target fog node.

In the Fog re-encryption phase, the fog node, using the re-encryption key, converts the attached encrypted symmetric key of A into another ciphered key for user B, without revealing any of the message content or the keys of the parties.

Finally, in the Delegatee decryption step, the delegatee firstly decrypts the re-encrypted ciphered key. After that, the recovered symmetric key is used to decrypt the ciphertext and retrieve the original message.

B. ALGORITHMS

The proposed PRE scheme consists of five algorithms. Each algorithm is defined based on its input and output parameters as follows:

1. **System Setup (*Setup*)**
   - Input: Security parameter n.
   - Output: System parameters SP= (E, q, p, e, G, s).
   - Syntax: ***Setup*** $(1^n) \rightarrow SP$.
2. **Key Generation (*KeyGen*)**
   - Input: System parameters SP.
   - Output: Secret keys sk_A, sk_B for users A and B, and a re-encryption key rk for a proxy server.
   - Syntax: ***KeyGen*** $(SP) \rightarrow sk_A, sk_B, rk$.
3. **Data Owner Encryption (*OwnerEnc*)**
   - Input: Message m, system parameters SP, and the public key pk_A of a party A.
   - Output: The ciphertext CT_A and ciphered key CK_A.
   - Syntax: ***OwnerEn*** $(m, SP, Pk_A) \rightarrow CT_A, CK_A$.
4. **Fog Re-encryption (*FogReEnc*)**
   - Input: The ciphered key CK_A, system parameters SP, and the re-encryption key rk.
   - Output: The ciphered key CK_B.
   - Syntax: ***FogReEnc*** $(CK_A, SP, rk) \rightarrow CK_B$.
5. **User Decryption (*UserDec*)**
   - Input: The ciphertext CT_A, ciphered key CK_B, system parameters SP, and a secret key sk_B of user B.
   - Output: The original message m.
   - Syntax: ***UserDec*** $(CT_A, CK_B, SP, sk_B) \rightarrow m$.

C. CONSTRUCTION OF THE SECURITY MODEL

In this section, we provide the construction details of the security model in our proposed scheme. The essential requirement in fog computing is to inflict less computational complexity on resource-constrained IoT devices, where the heavy computational load is offloaded to the fog node. First, we propose efficient encryption and decryption of IoT data using symmetric encryption that is performed in IoT nodes. Second, the scheme is further designed to reduce the computational overhead incurred by the fog node in which the delegation task is restricted to re-encrypting the symmetric cipher key instead of re-encrypting the entire message. The workflow of

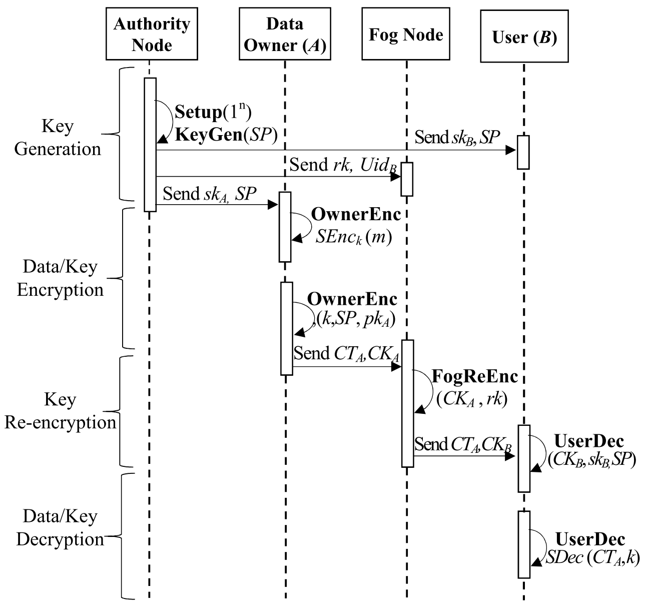

FIGURE 3. Sequence of workflow in the proposed scheme.

the proposed scheme is illustrated in figure 3. The construction details are as follows.

1) SYSTEM SETUP

Consider $E(\mathbb{F}_q)$ be an elliptic curve defined over the finite field $\mathbb{F}_q$, where q is a large prime number and G is a point on E of order p. Let G1, G2 are two multiplicative groups of prime order p. A bilinear map is a function $e : G1 \times G1 \rightarrow G2$, and $s = e(G1, G1) \in G2$. The key authority node runs *setup* algorithm to take a security parameter t as input, and outputs the public system parameters SP= (E, q, p, e, G, s).

2) KEY GENERATION

The key authority node runs *KeyGen* algorithm, which takes the system parameters SP as input, and then performs the following:

1) Choose a random number $r \in \mathbb{Z}_p^*$.
2) Compute public and private key pairs (sk, pk) for users A and B such that, $sk_A = ar \in \mathbb{Z}_p^*, \in pk_A = arG$, $sk_B = br \in \mathbb{Z}_p^* pk_B = brG$.
3) Compute a fog re-encryption key rk from the secret key sk_A of A, and the public key pk_B of B such that $rk = (ar)^{-1}brG = a^{-1}bG$
4) Return (sk_A, pk_A), (sk_B, pk_B) and rk.

3) DATA OWNER ENCRYPTION

The delegator runs *OwnerEnc* algorithm that takes the message m, the system parameters SP, and the public key pk_A as inputs. Then, it applies the XXTEA and ElGamal ECC cryptographic algorithms to encrypt the message m and the symmetric key k, respectively, as follows:

- Choose a symmetric key $k \in \mathbb{Z}_p^*$ uniformly at random.
- Break the message into blocks of size b.

- Encrypt the blocks of message m using the number of rounds n with the symmetric key k by using XXTEA encryption algorithm, denoted as $\mathrm{C}T_A = SEnc_k(m)$.
- Represent the symmetric key k as a point on the elliptic curve E using the embedding function $f(k) \rightarrow \mathrm{P}_k$.
- Choose a secret number $v \in \mathbb{Z}_\mathrm{p}^*$ at random.
- Encrypt the represented key P_k using the public key pk_A of A such that, $CK_A = (\alpha, \beta) = (vpk_A, \mathrm{s}^v\mathrm{G} + \mathrm{P}_k)$.
- Return CT_A and CK_A.

4) FOG RE-ENCRYPTION

The *FogReEnc* algorithm is enabled when the target fog node obtains CK_A and rk. Thus, it converts the CK_A encrypted by user A to another form so that it can be decrypted only by user B, as follows:

- Compute $CK_B = (\alpha', \beta') = (e(\alpha, rk), \beta) = (e(v\mathrm{arG}, \mathrm{a}^{-1}\mathrm{bG}), \mathrm{s}^v\mathrm{G} + \mathrm{P}_k) = (\mathrm{s}^{v\mathrm{rb}}, \mathrm{s}^v\mathrm{G} + \mathrm{P}_k)$.
- Return CK_B.

5) USER DECRYPTION

The user node runs *UserDec* algorithm upon receiving the CT_A and CK_B from the fog server. Using the secret key sk_B of user B, CK_B is decrypted to retrieve the symmetric key k, which is then used to decrypt CT_A and retrieve the message m, as follows:

- Compute

$$\begin{aligned} \mathrm{P}_k &= \beta' - (\alpha')^{\frac{1}{\mathrm{rb}}}\mathrm{G} \\ &= \mathrm{s}^v\mathrm{G} + \mathrm{P}_k - (\mathrm{s}^{v\mathrm{rb}})^{\frac{1}{\mathrm{rb}}}\mathrm{G} \\ &= \mathrm{P}_k \\ f^{-1}\mathrm{P}_k &= k \end{aligned}$$

- Decrypt the ciphertext CT_A using XXTE encryption algorithm. The corresponding decryption is denoted as $SDec_k\ (CT_A) = m$.
- Return the message m.

D. SECURITY ANALYSIS

In this section, we discuss the security strength of our proposed scheme from the aspects of correctness, data confidentiality, and collision attack resistance.

1) CORRECTNESS

The encrypted message sent from one party to another in the fog network should be correctly decrypted. The correctness of a cryptographic construction depends on the ability of the decryption algorithms to produce the expected results using proper keys. Our scheme deals with the correctness of symmetric and asymmetric encryption and decryption. In the user message and symmetric key encryption algorithms, we compute $CT \leftarrow OwnerEnc\ (m)$, $CK \leftarrow FogReEnc\ (OwnerEnc\ (k))$, where $pk, sk,$ and rk are generated by the *setup* and *keyGen* algorithms. The correctness of our scheme can be asserted as follows:

- $Decrypt\ (sk_A, CK_A) \rightarrow k$
- $Decrypt\ (sk_B, FogReEnc\ (rk, CK_A)) \rightarrow k$
- $Decrypt\ (k, CT_A) \rightarrow m$

This indicates that the security correctness has been perceived in our scheme, and it has been proved by implementation.

2) DATA CONFIDENTIALITY

The confidentiality of the data can be guaranteed against users who do not hold the set of system parameters and private keys. The trusted authority node uniquely generates the system parameters and a secret key, and securely sends them to each particular user. Every time, the secret number and symmetric key are randomly generated by the data owner node to encrypt the message and the symmetric key. The fog nodes are assumed to be semi-trusted for performing the re-encryption computations. However, they are still unable to recover the value of required secret keys to retrieve the message. In the decryption phase, the identity of the delegatee is pre-determined by the authority node to the respective fog node. Therefore, the delegatee is the only user who has a valid secret key that can decrypt the symmetric key and then decrypt the data. Thus, our scheme satisfies the need for data confidentiality.

3) COLLUSION RESISTANCE

In our scheme, the trusted authority node generates the secret keys for different users. Because the secret key involves a random number r that is uniquely associated with each user, creating a set of components in different secret keys is meaningless; illegal users cannot obtain the decryption keys via collusion activities. Thus, the proposed scheme is collusion resistant.

V. PERFORMANCE ANALYSIS AND DISCUSSION

In this section, we analyzed the computational cost of our proposed hybrid lightweight proxy-based re-encryption scheme. We carried out experiments to measure the time required for encryption, re-encryption, and decryption operations.

All algorithms are implemented using the Java programming language. We utilize the NICS Crypto library to support proxy re-encryption, and the Bouncy Castle library for the use of other cryptographic primitives. The experiments are conducted on a test-bed that includes one laptop and two Android mobile phones playing the roles of the fog (or the trusted authority node), and IoT nodes, respectively. The laptop is equipped with Intel Core i3-2120, a 3.3 GHz CPU, 4 GB RAM, and Windows 7 (32-bit version). The first Android phone is the Honor 6×, while the second phone is the Samsung Galaxy Note 5. Both Android phones are equipped with octa-core processors and 3 GB running memory. The execution time is measured on multiple message sizes. Each experimental result is the mean of 20 runs.

We first measured the computation time of encryption and decryption functions using standard encryption algorithms such as AES-256 and RSA-1024. As it can be observed from Fig.4, the execution time of symmetric encryption and decryption using AES-256 over various data sizes is lower than that of the asymmetric encryption and decryption using

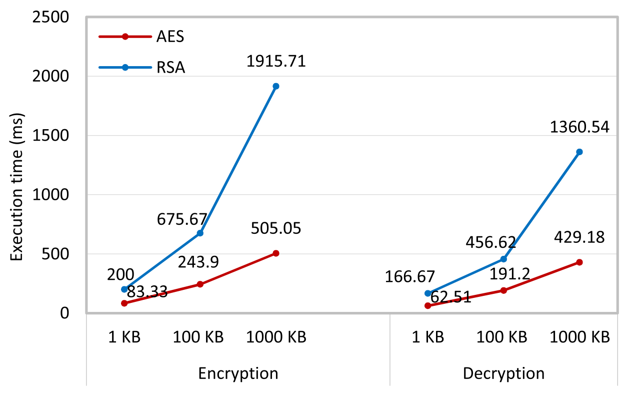

FIGURE 4. Execution time of the encryption and decryption using AES and RSA for various data sizes.

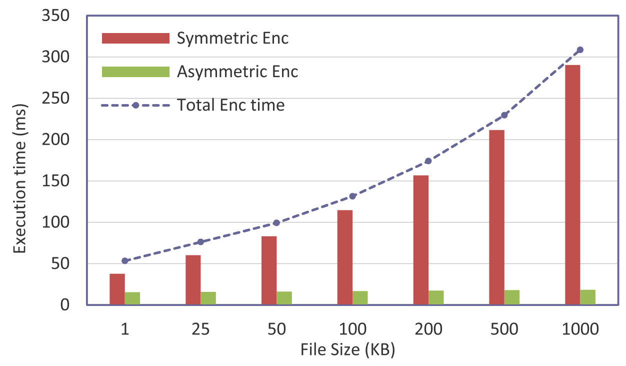

FIGURE 5. Execution time of the symmetric and asymmetric encryption of proposed scheme on multiple data sizes.

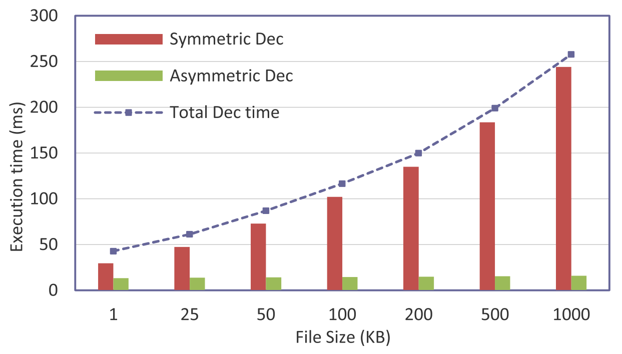

FIGURE 6. Execution time of the symmetric and asymmetric decryption of our scheme on multiple data sizes.

TABLE 1. Throughput comparison (KB/ms) of the encryption and decryption methods for our proposed scheme, AES, and RSA.

Method	Encryption			Decryption		
	1 KB	100 KB	1000 KB	1 KB	100 KB	1000 KB
Proposed	0.018	0.76	3.24	0.023	0.857	3.877
AES-256	0.012	0.41	1.98	0.016	0.523	2.33
RSA-1024	0.005	0.148	0.522	0.006	0.219	0.735

RSA-1024 algorithm which takes longer to process similar size messages.

Next, we evaluated the performance efficiency of our scheme. At first, the laptop generates all the keys and system parameters and sends them to both android mobile phones. Then, the first mobile phone encrypts the text of certain sizes using the XXTEA encryption algorithm. Next, it encrypts the symmetric key using ECC. After that, the laptop re-encrypts the encrypted symmetric key. Finally, the second mobile phone decrypts the transformed ciphered key. This is followed by the decrypting of the ciphertext.

The XXTEA symmetric algorithm operates on an array size of 32 bits and a 128-bit key length. The asymmetric encryption uses a 160-bit elliptic curve group based on the supersingular curve $y^2 = x^3 + x$ over a 512-bit finite field, which is used to achieve the security level of 128 bits.

First, we measured the computation cost of the key generation in our scheme. We found that the computation time required by the fog node to generate and send the keys to other parties is about 36 milliseconds. Fig. 5 and Fig. 6 show the execution time of the encryption and decryption, respectively, over various data sizes in our scheme. The total time as shown in the figures is the sum of symmetric and asymmetric encryption (Fig. 4), and the sum of symmetric and asymmetric decryption (Fig. 5), respectively.

From the results, we can observe that the symmetric encryption and decryption time increases linearly as the message size increases. Furthermore, it can be seen that the asymmetric encryption and decryption time is approximately constant because the symmetric key size is fixed to 128 bits in all message sizes. Table 1 illustrates the encryption and decryption throughput, which indicates the amount of data that can be processed per unit time, of proposed scheme over various data sizes compared to the AES and RSA standard ciphers. As shown from the results, our scheme outperforms the AES and RSA in encryption and decryption processes.

After that, we evaluated the computation cost of re-encryption process executed by the fog node in our scheme. We found that our scheme consumes on average, about 35 milliseconds to perform the re-encryption process, regardless of the message size. From table 2, it can be noticed the significant difference in the re-encryption time between our scheme, and the ECC and RSA encryption algorithms which are widely used in PRE schemes. This fast execution of the re-encryption process emanates from the fact that our scheme does not re-encrypt the entire message, where a symmetric cipher key attached to the ciphertext is only re-encrypted. The entire message must be re-encrypted in other PRE schemes, which impose additional computation overhead on fog nodes with time-delay for the re-encryption process that increases with the increasing message sizes and increasing number of users.

We compared our scheme with that of Diro *et al.* [4], which adopts ECC to perform the proxy-based cryptographic operations in fog-to-things computing. Fig. 7 provides the

TABLE 2. Execution time (ms) of the re-encryption method for our proposed scheme, ecc, and rsa.

Method	1 KB	100 KB	1000 KB
Proposed	34.78	35.21	35.94
ECC	178.96	332.29	998.63
RSA	194.21	591.86	1758.53

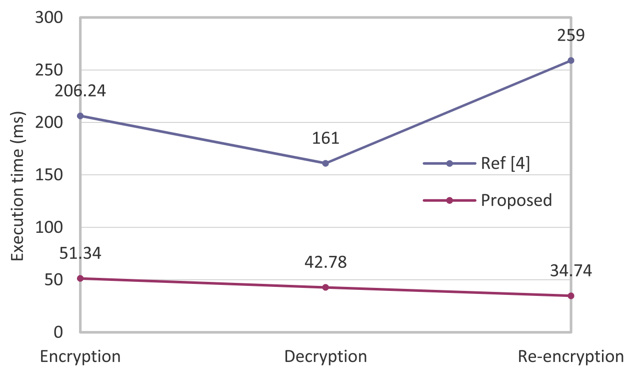

FIGURE 7. Execution time of encryption, decryption, and re-encryption for the proposed scheme vs Ref [4] for 1024 bytes of message.

comparison of the computation times of encryption, decryption, and re-encryption for a message of size 1 KB and a security level of 128 bits between our scheme and the one in [4]. As it can be observed that the data owner's encryption time and the user's decryption time in our scheme are smaller than those in [4]. Moreover, a significant difference can be seen in the re-encryption time on a proxy server, which becomes more obvious as the message size increases.

In summary, the experimental results show that our scheme incurs less computation time for the encryption of the data owner, the re-encryption of fog nodes, and the decryption of the user. Therefore, due to the computational cost of fog nodes and end users are greatly reduced, the proposed scheme is more suitable for real-time IoT applications as well as for the large amounts of communicated data in fog-to-things environment.

VI. CONCLUSION

In this paper, we have proposed a lightweight proxy-based re-encryption scheme for fog-to-things computing. In the proposed scheme, a hybrid cryptosystem based on lightweight encryption algorithms is presented to reduce the computational overhead on fog nodes as well as end users with highly constrained devices. The performance analysis shows that the proposed scheme achieves a significant reduction in the processing time that fog nodes require to carry out the re-encryption process. Whereas, the re-encryption implementation with the ECC for the symmetric cipher key attached to the ciphertext consumes a constant-time of approximately 35 milliseconds, regardless of message size. In addition, the implementation of encryption and decryption based on the XXTEA and ECC lightweight algorithms has proved that our scheme incurs less computation cost than standard ciphers, and thus, best fits the resource-constrained devices in fog networks. The results indicate that the proposed scheme is very lightweight, which makes it promising for real-time IoT applications.

As a future work, the proposed scheme will be implemented and evaluated in realistic IoT scenarios. We will further explore the possibility of enhancing security and performance using intelligent cryptography.

REFERENCES

[1] S. Maitra and K. Yelamarthi, "Rapidly deployable IoT architecture with data security: Implementation and experimental evaluation," *Sensors*, vol. 19, no. 11, p. 2484, 2019.
[2] P. Maiti, J. Shukla, B. Sahoo, and A. K. Turuk, "Mathematical modeling of qos-aware fog computing architecture for iot services," in *Emerging Technologies in Data Mining and Information Security*. Singapore: Springer, 2019, pp. 13–21.
[3] R. W. L. Coutinho and A. Boukerche, "Modeling and analysis of a shared edge caching system for connected cars and industrial IoT-based applications," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2003–2012, Mar. 2020.
[4] A. A. Diro, N. Chilamkurti, and Y. Nam, "Analysis of lightweight encryption scheme for Fog-to-Things communication," *IEEE Access*, vol. 6, pp. 26820–26830, 2018.
[5] Y. Miao, J. Ma, X. Liu, J. Weng, H. Li, and H. Li, "Lightweight fine-grained search over encrypted data in fog computing," *IEEE Trans. Services Comput.*, vol. 12, no. 5, pp. 772–785, Sep. 2019.
[6] R. Lu, K. Heung, A. H. Lashkari, and A. A. Ghorbani, "A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced IoT," *IEEE Access*, vol. 5, pp. 3302–3312, 2017.
[7] H. Atlam, R. Walters, and G. Wills, "Fog computing and the Internet of Things: A review," *Big Data Cognit. Comput.*, vol. 2, no. 2, p. 10, 2018.
[8] Y. Yu, R. Chen, H. Li, Y. Li, and A. Tian, "Toward data security in edge intelligent IIoT," *IEEE Netw.*, vol. 33, no. 5, pp. 20–26, Sep. 2019.
[9] P. Zhang, J. K. Liu, F. R. Yu, M. Sookhak, M. H. Au, and X. Luo, "A survey on access control in fog computing," *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 144–149, Feb. 2018.
[10] B. J. Mohd and T. Hayajneh, "Lightweight block ciphers for IoT: Energy optimization and survivability techniques," *IEEE Access*, vol. 6, pp. 35966–35978, 2018.
[11] M. Al-khafajiy, T. Baker, A. Waraich, D. Al-Jumeily, and A. Hussain, "IoT-fog optimal workload via fog offloading," in *Proc. IEEE/ACM Int. Conf. Utility Cloud Comput. Companion (UCC)*, Zurich, Switzerland, Dec. 2018, pp. 359–364.
[12] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1998, pp. 127–144.
[13] S. Fugkeaw and H. Sato, "Improved lightweight proxy re-encryption for flexible and scalable mobile revocation management in cloud computing," in *Proc. IEEE 9th Int. Conf. Cloud Comput. (CLOUD)*, San Francisco, CA, USA, Jun. 2016, pp. 894–899.
[14] A. Ragab, G. Selim, A. Wahdan, and A. Madani, "Robust Hybrid Lightweight Cryptosystem for Protecting IoT Smart Devices," in *Int. Conf. Secur., Privacy Anonymity Comput., Commun. Storage*. Cham, Switzerland: Springer, 2019, pp. 5–19.
[15] O. A. Khashan, A. M. Zin, and E. A. Sundararajan, "Performance study of selective encryption in comparison to full encryption for still visual images," *J. Zhejiang Univ. Sci. C*, vol. 15, no. 6, pp. 435–444, Jun. 2014.
[16] E. M. Galas and B. D. Gerardo, "Feasibility assessment on the implementation of the enhanced XXTEA on IoT devices," in *Proc. IEEE 9th Int. Conf. Syst. Eng. Technol. (ICSET)*, Shah Alam, Malaysia, Oct. 2019, pp. 178–182.
[17] K. Biswas, V. Muthukkumarasamy, X. W. Wu, and K. Singh, "Performance evaluation of block ciphers for wireless sensor networks," in *Advanced Computing and Communication Technologies*. Singapore: Springer, 2016, pp. 443–452.

[18] M. Sun, C. Ge, L. Fang, and J. Wang, "A proxy broadcast re-encryption for cloud data sharing," *Multimedia Tools Appl.*, vol. 77, no. 9, pp. 10455–10469, May 2018.

[19] K. R. ., "Elliptic curve ElGamal encryption and signature schemes," *Inf. Technol. J.*, vol. 4, no. 3, pp. 299–306, Mar. 2005.

[20] J. Hou, M. Jiang, Y. Guo, and W. Song, "Efficient identity-based multi-bit proxy re-encryption over lattice in the standard model," *J. Inf. Secur. Appl.*, vol. 47, pp. 329–334, Aug. 2019.

[21] Z. Qin, H. Xiong, S. Wu, and J. Batamuliza, "A survey of proxy re-encryption for secure data sharing in cloud computing," *IEEE Trans. Services Comput.*, early access, Apr. 6, 2016, doi: 10.1109/TSC.2016.2551238.

[22] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. 31, no. 4, pp. 469–472, Jul. 1985.

[23] N. Farjana, S. Roy, M. J. N. Mahi, and M. Whaiduzzaman, "An identity-based encryption scheme for data security in fog computing," in *Proc. Int. Joint Conf. Comput. Intell.* Singapore: Springer, 2020, pp. 215–226.

[24] J. Ni, K. Zhang, X. Lin, and X. Shen, "Securing fog computing for Internet of Things applications: Challenges and solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 601–628, 1st Quart., 2018.

[25] A. Alrawais, A. Alhothaily, C. Hu, X. Xing, and X. Cheng, "An attribute-based encryption scheme to secure fog communications," *IEEE Access*, vol. 5, pp. 9131–9138, 2017.

[26] Q. Huang, Y. Yang, and L. Wang, "Secure data access control with ciphertext update and computation outsourcing in fog computing for Internet of Things," *IEEE Access*, vol. 5, pp. 12941–12950, 2017.

[27] P. Zhang, Z. Chen, J. K. Liu, K. Liang, and H. Liu, "An efficient access control scheme with outsourcing capability and attribute update for fog computing," *Future Generat. Comput. Syst.*, vol. 78, pp. 753–762, Jan. 2018.

[28] D. J. Wheeler and R. M. Needham, "Correction to XTEA," Comput. Lab., Univ. Cambridge, Cambridge, U.K., Draft Tech. Rep., Oct. 1998. [Online]. Available: https://www.cl.cam.ac.uk/archive/djw3/xxtea.ps

[29] J. Weng, R. H. Deng, X. Ding, C. K. Chu, and J. Lai, "Conditional proxy re-encryption secure against chosen-ciphertext attack," in *Proc. 4th Int. Symp. Inf., Comput., Commun. Secur., Sydney Aust.*, 2009, pp. 322–332.

[30] P. Zeng and K.-K. Raymond Choo, "A new kind of conditional proxy re-encryption for secure cloud storage," *IEEE Access*, vol. 6, pp. 70017–70024, 2018.

[31] Y. Kawai, "Outsourcing the re-encryption key generation: Flexible ciphertext-policy attribute-based proxy re-encryption," in *Proc. Int. Conf. Info Secur. Pract. Exper.*, May 2015, pp. 301–315.

[32] S. Kim and I. Lee, "IoT device security based on proxy re-encryption," *J. Ambient Intell. Hum. Comput.*, vol. 9, no. 4, pp. 1267–1273, 2018.

[33] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *Proc. 5th ACM Symp. Inf., Comput. Commun. Secur.*, Beijing China, Apr. 2010, pp. 261–270.

[34] H. Hong and Z. Sun, "Towards secure data sharing in cloud computing using attribute based proxy re-encryption with keyword search," in *Proc. IEEE 2nd Int. Conf. Cloud Comput. Big Data Anal. (ICCCBDA)*, Chengdu, China, Apr. 2017, pp. 218–223.

[35] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *Proc. Int. Conf. Appl. Cryptogr. Netw. Secur.*, Zhuhai, China, Jun. 2007, pp. 288–306.

[36] K. Liang, Z. Liu, X. Tan, D. S. Wong, and C. Tang, "A CCA-secure identity-based conditional proxy re-encryption without random oracles," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, Seoul South Korea, Nov. 2012, pp. 231–246.

[37] R. Roy and P. P. Mathai, "Proxy re-encryption schemes for secure cloud data and applications: A survey," *Int. J. Comput. Appl.*, vol. 164, no. 5, pp. 1–6, 2017.

[38] L. Jiang and D. Guo, "Dynamic encrypted data sharing scheme based on conditional proxy broadcast re-encryption for cloud storage," *IEEE Access*, vol. 5, pp. 13336–13345, 2017.

[39] Q. Tang, "Type-based proxy re-encryption and its construction," in *Proc. Int. Conf. Cryptol.*, Kharagpur, India, Dec. 2008, pp. 130–144.

[40] C. Sur, Y. Park, S. U. Shin, K. H. Rhee, and C. Seo, "Certificate-based proxy re-encryption for public cloud storage," in *Proc. 7th Int. Conf. Innov. Mobile Internet Services Ubiquitous Comput.*, Taichung, Taiwan, Jul. 2013, pp. 159–166.

[41] C. K. Chu and W. G. Tzeng, "Identity-based proxy re-encryption without random oracles," in *Proc. Int. Conf. Inf. Secur.*, Valparaíso, Chile, Oct. 2007, pp. 189–202.

[42] K. Fan, J. Wang, X. Wang, H. Li, and Y. Yang, "A secure and verifiable outsourced access control scheme in fog-cloud computing," *Sensors*, vol. 17, no. 7, p. 1695, 2017.

[43] Z. Wang, "Leakage resilient ID-based proxy re-encryption scheme for access control in fog computing," *Future Gener. Comput. Syst.*, vol. 87, pp. 679–685, Apr. 2018.

[44] L. Ferretti, M. Marchetti, and M. Colajanni, "Fog-based secure communications for low-power IoT devices," *ACM Trans. Internet Technol.*, vol. 19, no. 2, p. 27, 2019.

[45] A. Vishwanath, R. Peruri, and J. S. He, "Security in fog computing through encryption," *Int. J. Inf. Technol. Comput. Sci.*, vol. 8, no. 5, pp. 28–36, 2016.

[46] D. Boruah and M. Saikia, "Implementation of ElGamal Elliptic Curve Cryptography over prime field using C," in *Proc. Int. Conf. Inf. Commun. Embedded Syst. (ICICES)*, Chennai, India, vol. 2014, pp. 1–7.

[47] R. Sunuwar and S. K. Samal. *Elgamal Encryption using Elliptic Curve Cryptography*. Accessed: Dec. 21, 2019. [Online]. Available: https://cse.unl.edu/~ssamal/crypto/EEECC.pdf

[48] N. Koblitz, "Elliptic curve cryptosystems," *Math. Comput.*, vol. 48, no. 177, pp. 203–209, 1987.

[49] V. S. Miller, "Use of elliptic curves in cryptography," in *Proc. Conf. Theory Appl. Cryptograph. Techn.*, vol. 218. Berlin, Germany: Springer-Verlag, 1985, pp. 417–426.

OSAMA A. KHASHAN was born in Jordan, in 1983. He received the B.S. degree in computer science from Irbid National University, Jordan, in 2005, the M.S. degree in information technology from Utara University Malaysia, in 2008, and the Ph.D. degree in computer science from the National University of Malaysia, Malaysia, in 2014. He is currently working as an Assistant Professor with the College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia. His research works focus on information and network security, cryptology, watermarking, digital image processing, and performance analysis.