IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 20, 2017, accepted September 23, 2017, date of publication September 28, 2017,
date of current version November 7, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2757844

A Security Model for Preserving the Privacy of Medical Big Data in a Healthcare Cloud Using a Fog Computing Facility With Pairing-Based Cryptography

HADEAL ABDULAZIZ AL HAMID[1], SK MD MIZANUR RAHMAN[1], (Member, IEEE), M. SHAMIM HOSSAIN[2], (Senior Member, IEEE), AHMAD ALMOGREN[3], (Member, IEEE), AND ATIF ALAMRI[2], (Member, IEEE)

[1]Information Systems Department, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia
[2]Department of Software Engineering, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia
[3]Computer Science Department, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia

Corresponding authors: M. Shamim Hossain (mshossain@ksu.edu.sa) and Sk Md Mizanur Rahman (mizan@ksu.edu.sa)

This work was supported by the Deanship of Scientific Research at King Saud University, Riyadh, Saudi Arabia, through the Profile Research Group Project under Grant PRG-1436-17.

ABSTRACT Nowadays, telemedicine is an emerging healthcare service where the healthcare professionals can diagnose, evaluate, and treat a patient using telecommunication technology. To diagnose and evaluate a patient, the healthcare professionals need to access the electronic medical record (EMR) of the patient, which might contain huge multimedia big data including X-rays, ultrasounds, CT scans, and MRI reports. For efficient access and supporting mobility for both the healthcare professionals as well as the patients, the EMR needs to be kept in big data storage in the healthcare cloud. In spite of the popularity of the healthcare cloud, it faces different security issues; for instance, data theft attacks are considered to be one of the most serious security breaches of healthcare data in the cloud. In this paper, the main focus has been given to secure healthcare private data in the cloud using a fog computing facility. To this end, a tri-party one-round authenticated key agreement protocol has been proposed based on the bilinear pairing cryptography that can generate a session key among the participants and communicate among them securely. Finally, the private healthcare data are accessed and stored securely by implementing a decoy technique.

INDEX TERMS Key management, security and privacy, medical big data, fog computing, pairing-based cryptography, decoy technique.

I. INTRODUCTION

Big data in healthcare refers to sets of electronic medical health data that are large and complex. Due to their huge volume and complexity, it is difficult (or infeasible) to manage those data sets using traditional software and/or hardware [1], [2]. The diversity and volume of multimedia medical big data (MBD) and efficient accessibility of these datasets make it irresistible [2]–[7]. MBD in the healthcare industry includes patient data in electronic patient records (EPRs); clinical data from computerized physician order entries (CPOEs); machine generated/sensor data, such as from monitoring vital signs; clinical decision support systems (medical imaging, physician's written notes and prescriptions, insurance, laboratory, pharmacy, and other administrative data); social media posts, including Twitter feeds (so-called tweets) [8], [9], blogs [10], [11], status updates on Facebook and other platforms, and web pages; and non-patient-specific information, including emergency care data, news feeds, and articles in medical journals.

Telemedicine is one of the emerging fields for e-health research. In the telemedicine service, EMRs including MBD, images, and multimedia medical data are transmitted on the fly over insecure internet connections as they are required by the remote doctors. The healthcare cloud infrastructure would make it much easier to pull all different healthcare information together for a patient while the patient moves from one hospital to another; as a result, the patients' information can be managed and tracked easily. The healthcare cloud

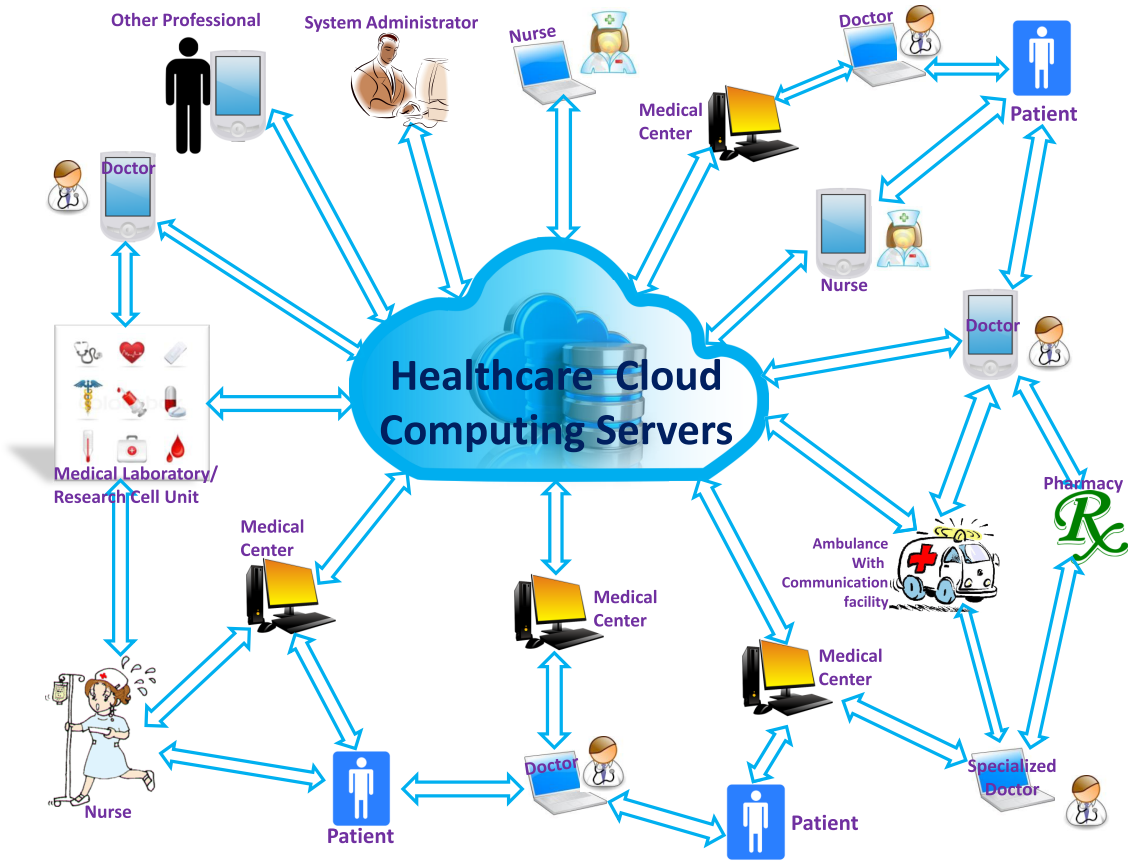

FIGURE 1. High level illustration of healthcare cloud.

is a cloud computing infrastructure where all the healthcare service providers and stakeholders can communicate with each other through the cloud servers, as illustrated in Figure 1.

Healthcare cloud computing offers the benefit of both software and hardware through the provision of services over the Internet. Cloud computing is defined by the NIST (2009) as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [12], [13].

Similar to cloud computing, healthcare cloud computing has different issues related to its security, the most important of which are: legal and policy issues, data protection, privacy protection, lack of transparency, cybersecurity issues, absence of security standards, and software licensing [14].

Each of these issues has different challenges that can be briefly discussed as follows. The challenges related to cloud computing's legal and policy issues are: liability, applicable law, compliance, copyright, data portability, and data protection. Speaking about protection, privacy protection means to protect the personally identifiable information (PII), by making it clear to the consumer how it is used and where it is stored. Usually, privacy issues are all about three things, which are trust, uncertainty, and compliance. Also, another issue related to the consumer is lack of transparency, which may appear through the consumer not knowing where his/her data are physically stored or what happens to it. On the other hand, another cloud security issue is cyber security. Cyber security challenges are related to four factors which are: (1) information input, (2) information and commands output, (3) shared tenancy, and (4) physical infrastructure. Each one of these challenges contains different sub-challenges, for example information input challenges are categorized into three areas: (1) challenges related to the way of collecting and delivering the information to cloud computing applications, (2) challenges related to the mechanism used to transport the information from utility to cloud computing facility, and (3) challenges related to information storage facility. So, each cloud computing issue or challenge has different staff we need to know more about. At the end, to define the relations between consumers, utilities, and third parties in the cloud, a proper policy is needed in order to make sure that the cloud computing is secure [15].

In this paper, a methodology is presented to secure patients' MBD in the healthcare cloud using the decoy technique with a fog computing facility. It serves as a second gallery to

contain decoy MBD (DMBD) that appear to the attacker as if it is the original MBD (OMBD). Unlike other methods, where the decoy files are called when an attacker is detected as accessing the system, in our proposed methodology the decoy files are retrieved from the beginning to ensure better security. Additionally, it uses a double security technique by encrypting the original file when an attacker recognizes that he/she is dealing with a decoy gallery; he/she would need to figure out how to decode the original gallery. As a result, our methodology ensures that the users' MBD are 100% secure and shortens the process. There is no need to worry if the user is an attacker, since by default it offers the decoy big data gallery directly to any user and keeps the original one hidden, which is only made available to a legitimate user after successful verification.

Organization of The Paper: Section II and its subsections describe the background and preliminaries of the proposed technique. Section III describes the existing literature on similar topics. The proposed system and methodology are described in Section IV; the justification for selecting cryptographic techniques have been described in Section V; computational analysis for cryptographic Implementation are described in Section VI, and Section VII concludes the paper.

II. BACKGROUND AND PRELIMINARIES

In this section we provide a few technical backgrounds that will help to ensure better understanding of our proposed technique.

A. CLOUD COMPUTING

Cloud computing has different service models, which are divided into three categories: (1) IaaS, which allows users to take advantage of the infrastructure without mentioning the hardware running behind it; (2) PaaS, which builds on IaaS and provides clients with access to basic operating software and optional services to develop and use software applications without software installation; and (3) SaaS, which enables clients to use software applications without having to install them on their personal computer, by offering these as a service through the Internet [16]. We can categorize cloud computing consistent with the deployment model into: (1) a public cloud, in which the resources are sold or rented to the public by the service provider, who at the same time is the owner; (2) a private cloud owned or rented by an organization; (3) community clouds, in which some closed communities share the same cloud resources; and (4) a hybrid cloud, which has the characteristics of two or more deployment models [17].

Several features are available in cloud computing, for example: on-demand broad network access, self-service, measured service, resource pooling, and rapid elasticity. Self-service means that the customers can manage and request their own resources. On the Internet or in private networks, the services offered are known as broad network access. In pooled resources, the customer draws from a pool of computing resources, usually in a remote data center. The services can be scaled larger or smaller, and customers are billed according to the measured use of a service [18].

B. FOG COMPUTING

Fog computing, as shown in Figure 2, is an emerging paradigm that provides storage, processing, and communication services closer to the end user. It reduces latency, provides location awareness, and supports high-density wireless networks. Providing data and putting them on the edge of a network to be nearer to the user are considered among the main tasks of fog computing. The end user is connected to different nodes, which are referred to as the ''edge,'' thus the term ''edge computing.'' Fog computing does not replace cloud computing. Rather, it extends the cloud to the edge of the network [19], [20]. Creating decoy information and locating it beside the real information in the cloud to hide the true data of the user is also called fog computing. This architecture offers a number of services that are related to the use of decoys. Hence, fog computing can be considered as an alternative name for the Decoy Document Distributor (D3), which is a tool for generating and monitoring decoys. This strategy is used to protect the real, sensitive data by providing a ''fog'' of misinformation. Decoy information, such as decoy documents, honey files, and honeypots, among others, can be generated when unauthorized access is detected. This confuses the attackers and makes them believe that they have the real, useful data when they actually do not. Decoys can be created manually by the user him/herself; for example, when the user creates a new document, he/she can create a fake document that will appear as a mirror document but contains bogus information. Such manual creation of decoys is obviously very tiring for the user, especially if we are talking about a large organization with multiple users and files. For this reason, fog computing is used to create decoys with minimal user intervention [21]–[23].

C. DECOYFILE

The basic idea behind this technique is to limit the damage caused by stolen data by decreasing the value of the stolen information. To achieve this, the decoy should have certain features. First, it should be believable. In the absence of any additional information, a perfectly believable decoy should make it impossible for an attacker to figure out that the data are not real. Thus, the decoy should seem authentic and trustworthy. Second, the decoy should be enticing enough to attract the attention of the attacker and make him/her open the file. Third, the decoy should be conspicuous, which is closely related to being enticing. Whereas enticing is related to how curious an attacker is about a decoy, conspicuousness has to do with how easy a decoy is to access. Therefore, the decoy should be easily located by search queries. Fourth, the decoy should be differentiable so that the real user can distinguish between the real and the decoy file. Balancing differentiability for authentic users with believability for attackers is one of the critical aspects of any decoy deployment system. Fifth, the decoy should be non-interfering so that the real user will

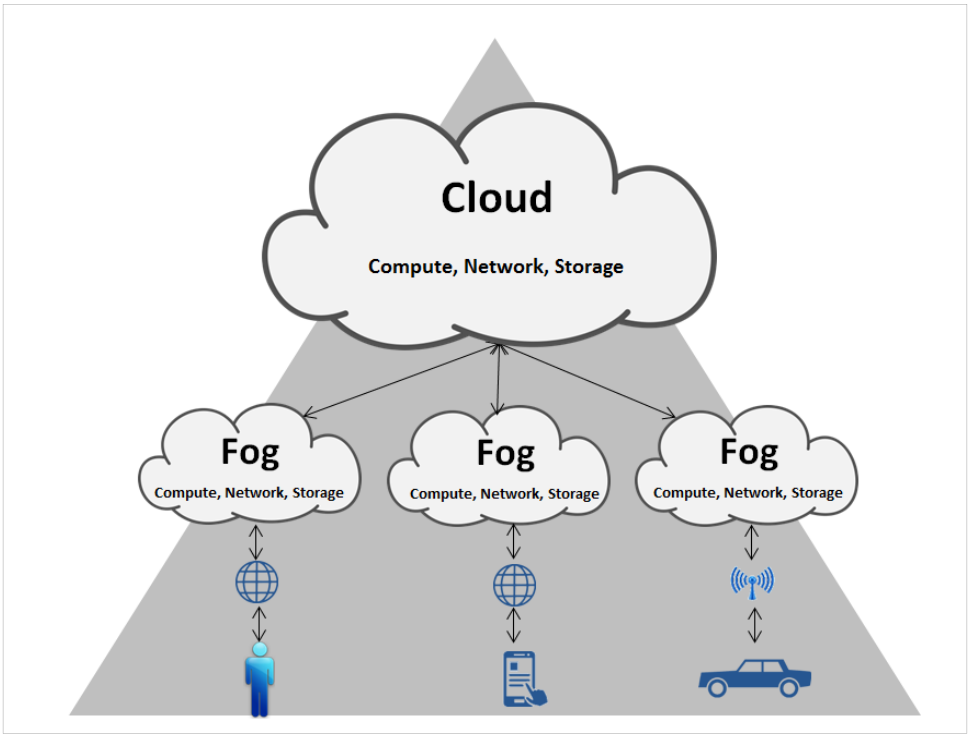

FIGURE 2. Fog computing architecture.

not accidentally misuse the bogus information contained in the decoy. Finally, the decoy should be detectable; this feature refers to the ability of decoys to alert their owners once they have been accessed [24]–[26].

D. BILINEAR PAIRING FUNCTION

A bilinear pairing is a map between cyclic groups. Let G_1, G_2, and G_T be cyclic groups with a prime order q. We consider G_1 and G_2 as additive groups and G_T to be a multiplicative group. A bilinear pairing is defined as: $e : G_1 \times G_2 \rightarrow G_T$, Which satisfies the following properties [27]–[30]:

- Bilinear: It holds the important property $e(aP, bQ) = e(P, Q)^{ab}$, where $\forall P, Q \in G_1$ and $\forall a, b \in Z$
- Non-degenerate: $e(P, P) \neq 1$
- Computability: A mapping is said to be computable if an algorithm exists which can efficiently compute $e(P, Q)$ for any $P, Q \in G_1$.

Also, the following are considered as bilinear pairing properties. $\forall P, Q \in G_1$:

- $e(P, \infty) = 1$ and $e(\infty, P) = 1$
- $e(P, -Q) = e(-P, Q) = e(P, Q)^{-1}$
- $e(P, Q) = e(Q, P)$

E. BILINEAR DIFFIE-HELLMAN PROBLEMS

The security of many pairing-based protocols is dependent on the intractability of the following problems [27]–[30]:

- Computational bilinear Diffie-Hellman problem (CBDH): which is the problem of computing $e(P, P)^{abc}$ when $P, aP, bP, cP \in G_1$ are given.
- Decision bilinear Diffie-Hellman problem (DBDH): which is the problem of deciding if $e(P, P)^{abc} = r$ when $P, aP, bP, cP \in G_1$ and $r \in G_T$ are given.

F. ELLIPTIC CURVE DIFFIE-HELLMAN

Elliptic curve Diffie-Hellman (ECDH) is a key-agreement protocol, more than an encryption algorithm. This means that ECDH defines how keys should be generated and exchanged between parties, and how actual data using such keys are encrypted is up to us. Based on our proposed system, we will encrypt the photos using the Blowfish algorithm while the key authentication and exchange will be done using ECDH [27]–[29]. Thus, the key is generated by the Elliptic Curve Cryptography (ECC) Private Key Generator (PKG) key generator and then the key agreement can be done by Diffie-Hellman (DH); the combination of these two concepts is ECDH. Now, the obvious question is why would using a curve make DH better than using the integers? Well, many different properties are related to using elliptic curves, such as EC, which improves efficiency by performing faster operations. Also, by using EC, we can achieve higher security with a smaller key size, so we can use a much smaller prime to achieve the same complexity as working with integers.

G. MOTIVATION AND OUR CONTRIBUTION

MBD stored in the healthcare cloud typically reside in a shared environment collocated with data from other stakeholders [31]. Data theft occurs when information is illegally

copied or taken from a business or individual. Such theft usually involves user information, such as passwords, social security numbers, credit card information, other personal information, and confidential corporate information. A recent study done in the Saudi population sought to determine the most important contents in mobile phones [32]. The results showed that 88% of the female respondents answered ''personal photos,'' whereas 63% of the male participants cited ''personal information'' and ''personal photos.'' The statistics clearly indicate that personal photos are the most important contents in mobile devices. This may also mean that types of content which may be important to some people may not be as important for others.

Based on what was described previously about the popularity of using cloud computing and the importance of securing the data within it, our paper will clarify the different security issues that can affect the protection of MBD in the healthcare cloud environment. Also, we have investigated the existing decoy technique and its suitability for protecting MBD in the healthcare cloud. Finally, we propose our method to secure the MBD in the healthcare cloud by using DMBD, which depends on using a fog computing facility and pairing-based cryptography (PBC). A session key is generated for secure communication among the participants by using PBC to access and store MBD in the cloud. An initial version of this research has been presented in a conference [33].

III. RELATED WORK

This section presents a comprehensive study on the use of a decoy technique to secure cloud data. According to Vikas *et al.* [34], there are different security issues in mobile cloud computing. These can be divided into five categories: (1) physical threats, which include mobile possession and lost or stolen devices; (2) application-based threats, such as those involving malware, spyware, privacy, and vulnerable applications; (3) network-based mobile security threats, including Wi-Fi sniffing, denial of service, and address impersonation; (4) web-based threats, such as phishing scams, drive-by downloads, browser exploits, and jail broken devices; and (5) other active attacks, including Internet protocol vulnerabilities, information recovery vulnerability, and unauthorized access to management interface.

A decoy defense network can be deployed to bolster the security in different situations. Voris *et al.* [21] discussed several scenarios in which a decoy can be used. One usage scenario involves using a decoy within a local computer, which means placing the decoy document within the same environment in which it was created. In another scenario, the decoy can be located on a network level. In both scenarios, the decoy is used to protect documents on different levels. However, a decoy can also be used to protect software, as by being made to look like a legitimate source code, decoy software can protect real software from unauthorized usage. Another decoy usage scenario applies a voicemail decoy to detect malicious activity; here the decoy is a legitimate voice message but contains false information. Lastly, a cloud-based decoy can be used to protect documents in the cloud against insider attacks.

A few studies have focused on securing cloud data by using decoy documents. For instance, Stolfo Salvator *et al.* [22] first carried out user behavior profiling to determine unauthorized access. When an attacker accesses the cloud, a decoy document is returned such that the real user's data are kept secure. Each decoy document header contains a hidden Hash-based Message Authentication Code (HMAC). Verification of whether or not the document is a decoy is done by calculating the HMAC based on the content of the document; if the two HMACs match, then the document is a decoy and an alert is issued. In this case, decoy documents are used for two purposes: first, to validate whether or not the data access is authorized when abnormal information access is detected, and second, to confuse the attacker by providing false documents. It should be noted that only decoy documents are used in this study, and these are selected manually and added into the file system by the user. In a similar technique carried out by Aruna *et al.* [35], malicious insider attacks were prevented by using decoy information technology. When abnormal information access is detected, the decoy helps to validate whether or not the access is authorized. Hence, when unauthorized data access is detected and verified, a malicious inside flood with bogus information is returned to dilute the real user data. Also, Patil *et al.* [36] presented an approach to secure cloud computing by using decoy documents. Abnormal data access patterns are detected by monitoring the data access. When unauthorized access is detected and verified, a large amount of decoy information is returned to the attacker to protect the real data from any misuse. Such technology could offer exceptional levels of user data security in cloud computing and social networks as well. Further, Khairnar and Borkar [37] explained the method that is used to secure cloud data in the following scenario: when a user logs in to the system, a ''Successful login'' SMS is sent to his/her mobile device. The user can execute all tasks after answering the security question. Thereafter, one of two different situations is presented: first, if the user answers the question correctly, the original file is downloaded. Second, if the user answers the question incorrectly, which means that he/she is an attacker, then the decoy document is downloaded. Then, the real user receives an SMS containing information on the attacker, such as the IP address, server name, and access date and time, which can be used to track the attacker. Sriram *et al.* [38], proposed a hybrid protocol that leverages on the advantages of encryption and fog computing to secure the cloud from insider attacks. The protocol used an approach called selective encryption: because of performance issues, not all data can be encrypted, and to address this concern, only selected information that needs more security is encrypted. This is done by giving the user an option to completely encrypt, selectively encrypt, or not encrypt his/her data at all. To protect the data from insider attacks, a data cleaning approach is used. When the data are decrypted by a legitimate user,

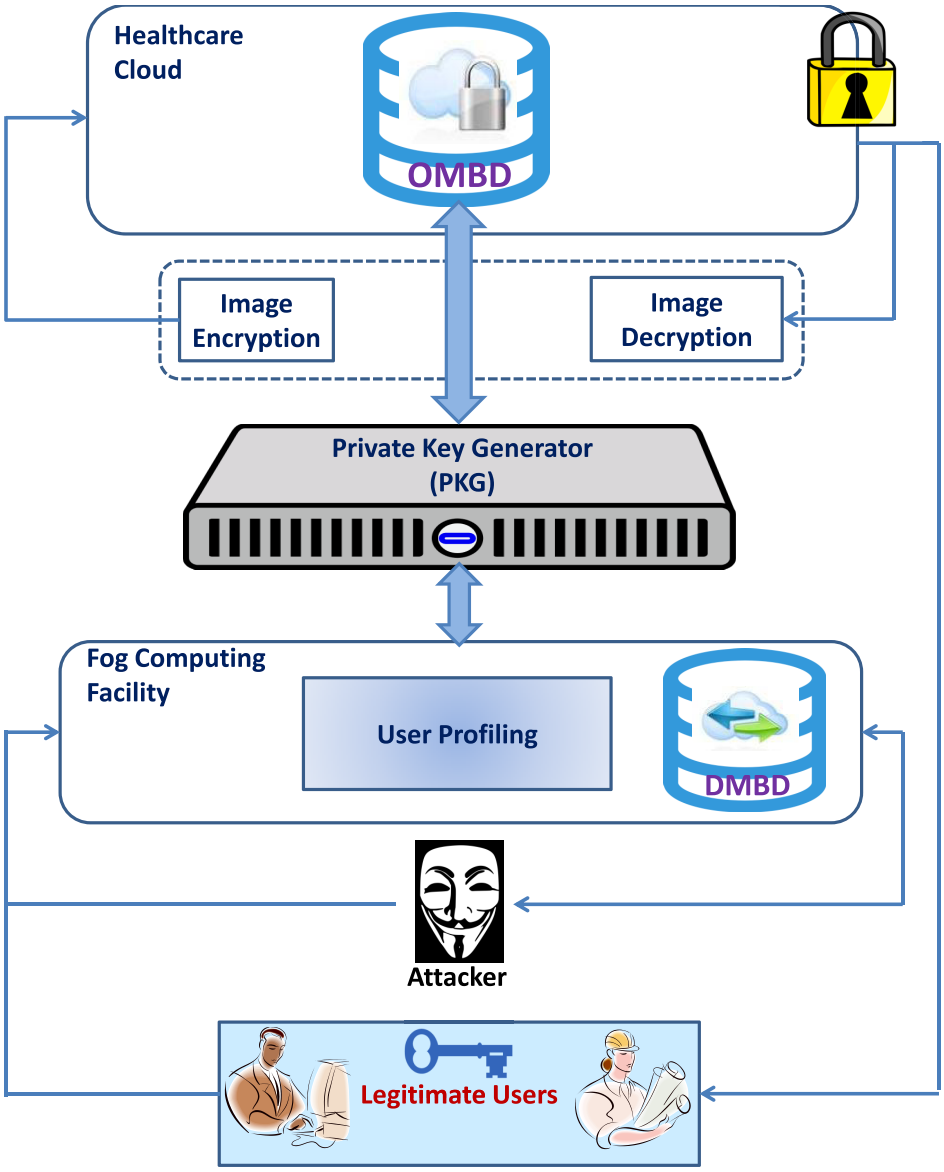

FIGURE 3. Proposed system architecture.

they are stored in the volatile memory of the physical machine for a temporary period, during which they could be misused by an insider attacker; a data cleaning technique is used to address this concern. As previously mentioned, besides selective encryption, fog computing is applied. To profile the user's search behavior, a neural network is used, with the following parameters: amount of downloaded data, nature of operation, division of tasks, IP address, and log file content. Whenever the system detects an attacker accessing it, decoy documents are deployed, and the legitimate user receives a warning e-mail once a decoy document is opened. Again, Vinod *et al.* [39], presented an integrated detection approach by combining user behavior and decoy technology. They cited that their proposed security system is appropriate only for a single cloud ownership system. Thus, they recommended enhancing their application to manage a cloud environment that has more than one cloud architecture. On the other hand, Liu [40] stated that a single security method cannot solve the cloud computing security problem and that many traditional and new technologies and strategies must be used together to protect the total cloud computing system because the data in the cloud are greatly dependent on the network and the server, which makes the data privacy issue more prominent than in the traditional network.

IV. PROPOSED SYSTEM

Now and in the subsequent sections, whenever we use the terms "Gallery/ Photo gallery" we mean "multimedia MBD

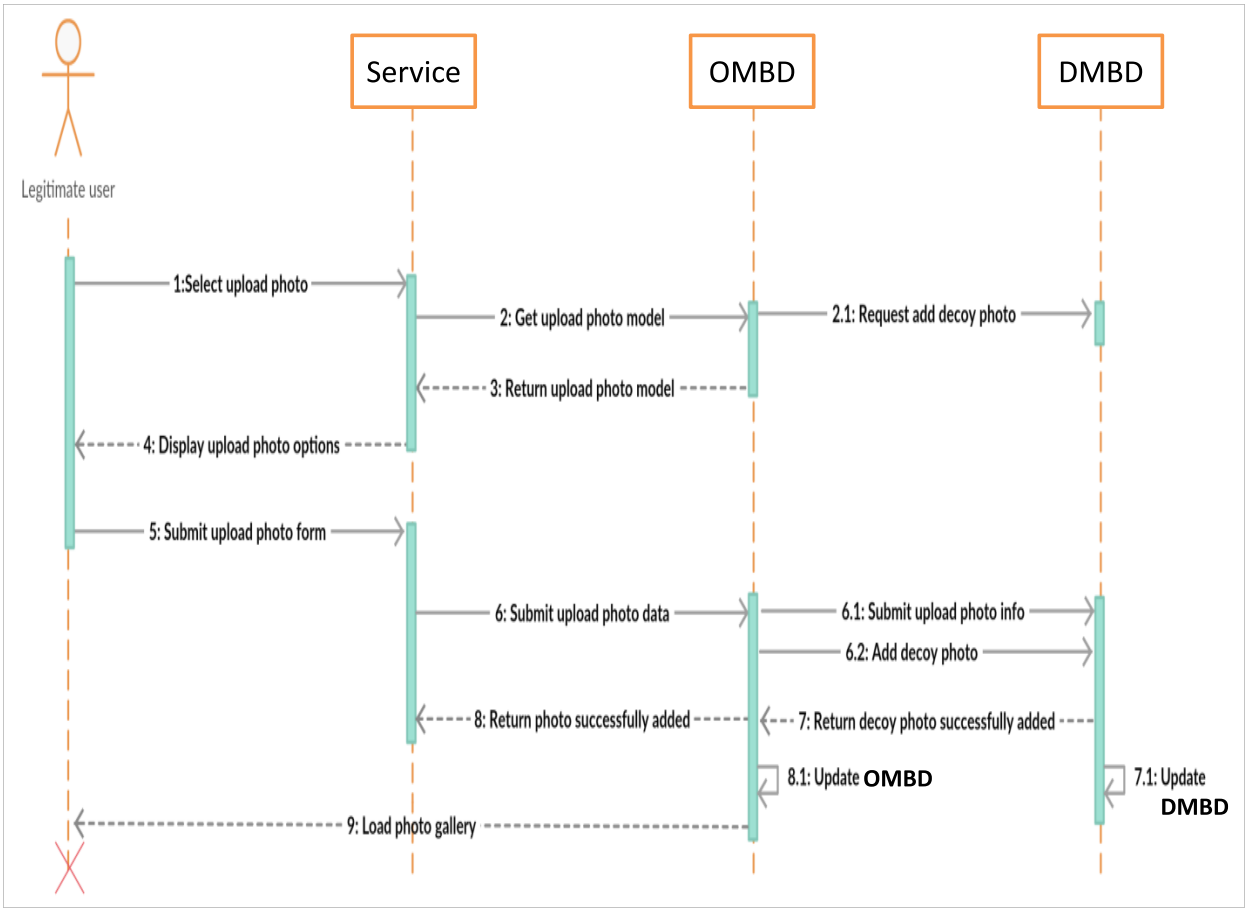

FIGURE 4. Upload photo/image process.

gallery," and the gallery contains MBD. Using fog computing facilities and the decoy technique, a DMBD is created. This technique can be considered as an illusion technique, as it makes the attacker believe that he/she has accessed the user's MBD while in fact it is just a decoy gallery. In our proposed system, as shown in Figure 3, once the user accesses his/her account, by default the DMBD is shown. Thus, both authorized and unauthorized users will be referred to the DMBD as the first step, while authorized legitimate users, as a second step, will be referred to the OMBD after being verified. We believe that by setting the default value of the DMBD as shown and the OMBD as hidden, we keep the original MBD more secure. Also, we believe that verifying that the user is legitimate is much easier than detecting the attacker, which is why we tried to deal with the attacker in the first place by offering the DMBD as the first step.

When the user accesses his/her account, whether he/she is a legitimate user or an attacker, his/her first step would be accessing the DMBD, which is located in the fog computing layer side by side with user profiling. User profiling is a familiar technique that can be applied to model in what way, at what time, and how considerable users access their information in the healthcare cloud. This method of behavior-based security is commonly used in fraud detection application. The DMBD contains fake MBD, which are supposed to make an attacker believe that he/she has accessed the user's photos/medical image while in fact it is just a decoy gallery. The legitimate user already knows that the gallery he/she accessed is not his/her original one, so would move on to the next step. Moving to the next step, the legitimate user can access his/her OMBD after being verified by passing the security challenge. The security challenge might be a challenging security question or even a verification code. Thus, if he/she passes the security challenge, that means he/she is the legitimate user, so will be able to access the OMBD which is located on the cloud computing layer. In the event of the user accessing only the DMBD, an SMS or email will be sent to the legitimate user to inform him/her that his/her account has been accessed. The message will contain the attacker's information (e.g., access time and date and the IP address).

Now, how do we ensure that the two galleries are similar to a large extent? Each time the legitimate user uploads a new photo/medical image on his/her account, a decoy photo from fog computing will be uploaded to his/her DMBD, as shown in Figure 4. When the user uploads the

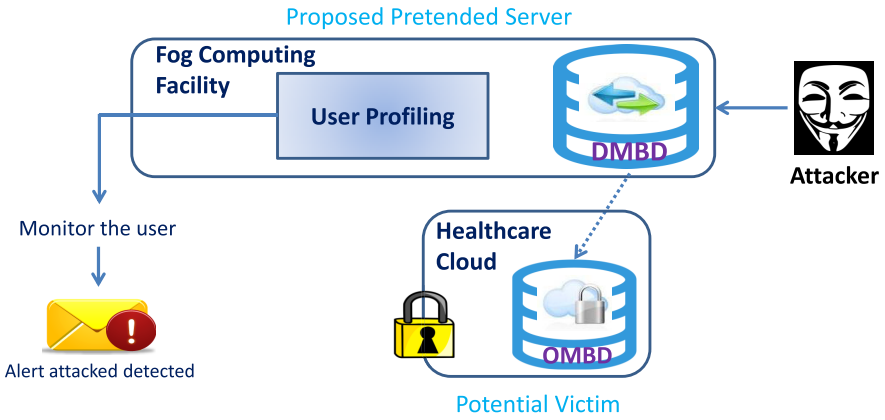

FIGURE 5. How the illusion technique works.

photo, he/she is supposed to recognize the photo category (ECG, X-ray, MRI, etc.), which will help fog computing to add the photo that belongs to the same category on the DMBD; this would make it closer to the original photo, so that the attacker would not differentiate between the real user's photo and the fake one. Thus, in our methodology, the user is not responsible for adding the decoy photo in his/her DMBD, since it will be added automatically while he/she is uploading the original photo to the OMBD.

A. DMBD ALGORITHM

DMBD is used as a trap gallery that makes it not of direct relevance to the legitimate user but it is used to secure his/her OMBD by distracting the attacker. As shown in Figure 5, the DMBD is placed in the fog computing as a honeypot to secure the original one, which is located in the cloud. As noted, a number of anomaly-detection systems are provided by fog computing such as user profiling and a decoy file system. Therefore, for each newly uploaded MBD in the OMBD, a decoy one will be placed on the DMBD.

B. USER PROFILING ALGORITHM

User profiling can help to determine whether a user is legitimate or not based on certain parameters, such as the user-search behavior, amount of downloaded data, nature of operations, division of tasks, and IP address. Knowing how a legitimate user deals with his/her cloud data based on these parameters will help determine whether or not the user is malicious [38]. There are three different types of user profiling, each with different advantages and disadvantages based on the techniques used. The type that we will use in our system is the hybrid user profile, which is a combination of explicit and implicit user profiles. The explicit user profile usually contains high-quality information because it is gathered from the user him/herself, but it requires a lot of effort from the user to update his/her profile information. On the other hand, the implicit user profile is automatically updated with minimal user effort; however, a large amount of interaction between the user and the content is required before an accurate user profile can be created. Thus, combining the two types into a hybrid user profile should reduce the weak points and enhance the strong points of each technique used to monitor the cloud data access and detect any unusual data access pattern [41].

C. KEY EXCHANGE ALGORITHM

In our proposed system, the OMBD agent and the DMBD need to communicate in different situations, for example, when the user uploads a new photo/image, the OMBD is supposed to communicate with the DMBD to inform it to add a new decoy photo. These communications between three parties (the user, the OMBD, and the DMBD) need to be secure. The following consecutive sub-sections describe the secure communication procedure among the parties. For better understanding the proposed protocol, Table 1 represents the symbols and corresponding meaning that are used to describe the protocol.

During the bootstrap of the communications, the Private Key Generator (PKG) Server generates following parameters:

- Determines two groups G_1 and G_2, of the same prime order q, where G_1 as an additive group and G_2 as a multiplicative group.
- Determines a generator P of G_1 and a bilinear map $e : G_1 \times G_1 \rightarrow G_2$
- Determines two collision resistant cryptographic hash functions H_1 and H_2, where$H_1 : \{0,1\}^* \rightarrow G_1$, (mapping from arbitrary-length strings to points in G_1) and $H_2 : \{0,1\}^* \rightarrow \{0,1\}^{\mu}$ (a mapping from arbitrary-length strings to μ-bit fixed length output).

1) KEY GENERATION

PKG picks $k+1$ random numbers $\rho_0, \rho_1, \rho_2, \cdots, \rho_k \in Z_q^*$ and generates a polynomial $g(x)$ of degree k, where $g(y) = \rho_0 + \rho_1 y + \rho_2 y^2 + \cdots + \rho_k y^k \in Z_q[y]$. PKG then computes $U_0 = \rho_0 P, U_1 = \rho_1 P, \ldots, U_k = \rho_k P$. The public parameters of the system published by PKG are $\{G_1, G_2, e, H_1, H_2, P, U_0, U_1, U_2, \ldots, U_k\}$ and the PKG keeps it's secret keys that are $\{\rho_0, \rho_1, \rho_2, \cdots, \rho_k\}$.

TABLE 1. Symbols are used to describe the protocol.

Symbols	Corresponding meaning
G_1	Is an additive group of prime order q
G_2	Is a multiplicative group of prime order q
H_1	A cryptographic hash function which maps from an arbitrary-length strings to a point in G_1
H_2	A cryptographic hash function which maps from an arbitrary-length strings to μ- bit fixed length string
$\rho_0, \quad \rho_1, \ldots, \rho_k$	Random master secrets of the system generated from Z_q^*
g(y)	Is a k degree polynomial using the random master secrets as the co-efficient
$U_0, U_1, U_2, \ldots, U_k$	The public parameters generated with the random secrets
σ_i	Secret key for the entity i
$?_i$	Public shared component for the entity i
K_i	Shared secret key for the entity i

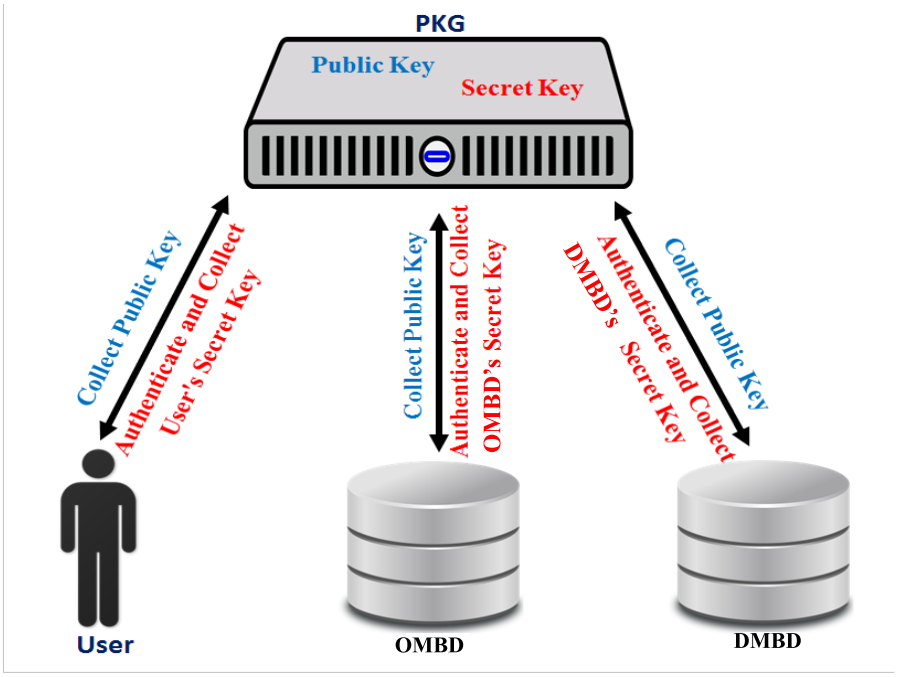

FIGURE 6. Collection of secret keys from the PKG.

On top of the above computations, PKG also computes $\sigma_i = g(ID_i) = \rho_0 + \rho_1 ID_i + \rho_2 (ID_i)^2 + \cdots + \rho_k (ID_i)^k \bmod q$ for the entity i whose identity is $ID_i \in Z_q^*$. In our proposed system there are three parties, i.e., the user, OMBD, and DMBD; and the identities can be represented as ID_R, ID_O and ID_D, respectively; $ID_R = H_1(ID_R^\ell)$, $ID_O = H_1(ID_O^\ell)$, $ID_D = H_1(ID_D^\ell)$ where $ID_R, ID_O, ID_D \in Z_q^*$. The parties send their public identity and collect their corresponding secret key from the PKG through private channel as shown in Figure 6. The public and private keys for the parties are computed by PKG as follows:

- For the User: Public Key: ID_R; Secret Key :$\sigma_R = g(ID_R)$.
- For the OMBD: Public Key: ID_O; Secret Key :$\sigma_O = g(ID_O)$.
- For the DMBD: Public Key: ID_D; Secret Key: $\sigma_O = g(ID_D)$.

2) KEY AGREEMENT

To establish a shared secret key among the parties (as shown in Figure 7), each party computes the following public parameters and sends to other two parties. The user entity computes $\psi_R = \sum_{i=0}^{k} (ID_R)^i U_i = \sigma_R P$ and sends to OMBD and DMBD. The OMBD computes $\psi_O = \sum_{i=0}^{k} (ID_O)^i U_i = \sigma_O P$ and sends to the user and DMBD. The DMBD computes $\psi_D = \sum_{i=0}^{k} (ID_D)^i U_i = \sigma_D P$ and sends to the user and the OMBD. Finally, each party can compute their shared secret key as follows:

- The user computes the shared secret key: $K_R = e(\psi_O, \psi_D)^{\sigma_R} = e(\sigma_O P, \sigma_D P)^{\sigma_R} = e(P, P)^{\sigma_R \sigma_O \sigma_D}$
- The OMBD computes the shared secret key: $K_O = e(\psi_R, \psi_D)^{\sigma_O} = e(\sigma_R P, \sigma_D P)^{\sigma_O} = e(P, P)^{\sigma_O \sigma_R \sigma_D}$
- The DMBD computes the shared secret key: $K_D = e(\psi_R, \psi_O)^{\sigma_D} = e(\sigma_R P, \sigma_O P)^{\sigma_D} = e(P, P)^{\sigma_D \sigma_R \sigma_O}$

Thus $K_R = K_O = K_D$.

D. MUTUAL AUTHENTICATION PROTOCOL

Now, each party needs to authenticate the other party in order to exchange messages between each other secretly.

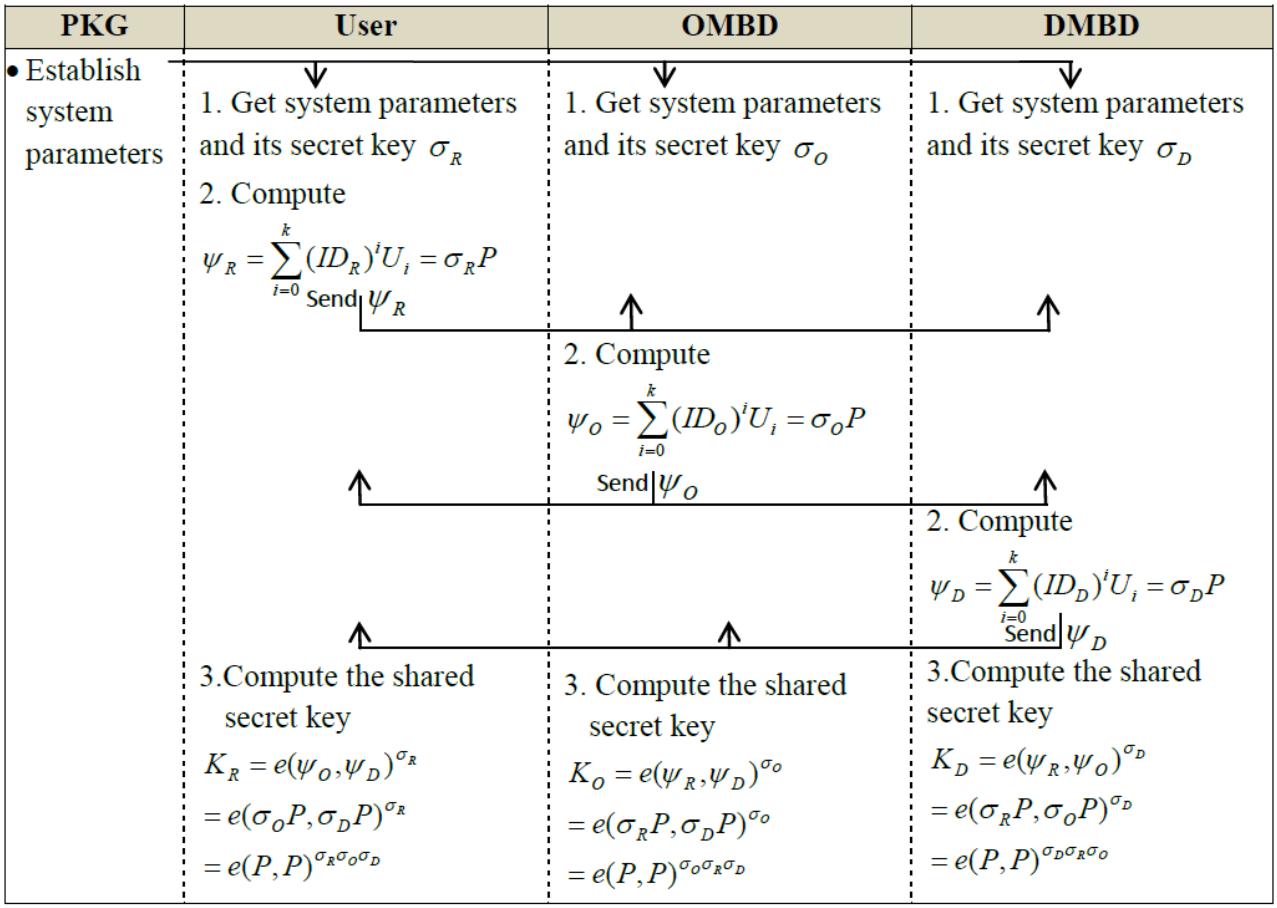

FIGURE 7. Three parties' secure communication.

As shown in Figure 8, firstly all the parties should compute their shared secret key, following the steps that described in the previous section. So, for the user the shared secret key is $K_R = e(\psi_O, \psi_D)^{\sigma_R} = e(\sigma_O P, \sigma_D P)^{\sigma_R} = e(P,P)^{\sigma_R \sigma_O \sigma_D}$. Then, the shared secret key for the OMBD is $K_O = e(\psi_R, \psi_D)^{\sigma_O} = e(\sigma_R P, \sigma_D P)^{\sigma_O} = e(P,P)^{\sigma_O \sigma_R \sigma_D}$. Also, for the DMBD the shared secret key is $K_D = e(\psi_R, \psi_O)^{\sigma_D} = e(\sigma_R P, \sigma_O P)^{\sigma_D} = e(P,P)^{\sigma_D \sigma_R \sigma_O}$. Now, let us assume that the user wants to authenticate the OPG and the DPG. The user will generate an authentication code $Aut_{R1} = H_2(K_R \| ID_R \| ID_O \| ID_D \| R_{R1})$ then send it with the random number R_{R1} to both OMBD and DMBD. After that, OMBD generates a verification code $Ver_{O1} = H_2(K_O \| ID_O \| ID_R \| ID_D \| R_{R1})$ and then compare it with Aut_{R1}, if they are equal then it generates another authentication code $Aut_{O2} = H_2(K_O \| ID_O \| ID_R \| R_{O1} \| R_{R1})$ and send it with the random number R_{O1} to the user. Finally, the user computes $Ver_{R2} = H_2(K_R \| ID_R \| ID_O \| R_{R1} \| R_{O1})$ and compare it to Aut_{O2}, if it matches then the authentication is successful otherwise it fails. On the other hand, DMBD generates a verification code $Ver_{D1} = H_2(K_D \| ID_D \| ID_R \| ID_O \| R_{R1})$ and then compare it with Aut_{R1}; if they are equal then it will generate another authentication code $Aut_{D2} = H_2(K_D \| ID_D \| ID_R \| R_{D1} \| R_{R1})$ and send it with the random number R_{D1} to the user. Finally, the user computes $Ver_{R3} = H_2(K_R \| ID_R \| ID_D \| R_{R1} \| R_{D1})$ and compare it to Aut_{D2}, if it matches then the authentication is successful otherwise it fails. Similarly OMBD and DMBD can also authenticate with the other party.

E. PHOTO ENCRYPTION ALGORITHM

Photo encryption is a technique used to secure a photo by changing it to an understandable one. Different photo encryption algorithms with different properties and different levels of security are available. In our proposed system, we are using the Blowfish algorithm. Blowfish is a symmetric key cryptography where the key does not change, such as an automatic file encryption. The reasons behind choosing this algorithm are the following: (1) Blowfish has a longer key length, making it the most secure algorithm; and (2) it can encrypt any photo file format of any size, black and white or even a color photo [42]. Note that in our system, the encryption/decryption key is established by sharing public

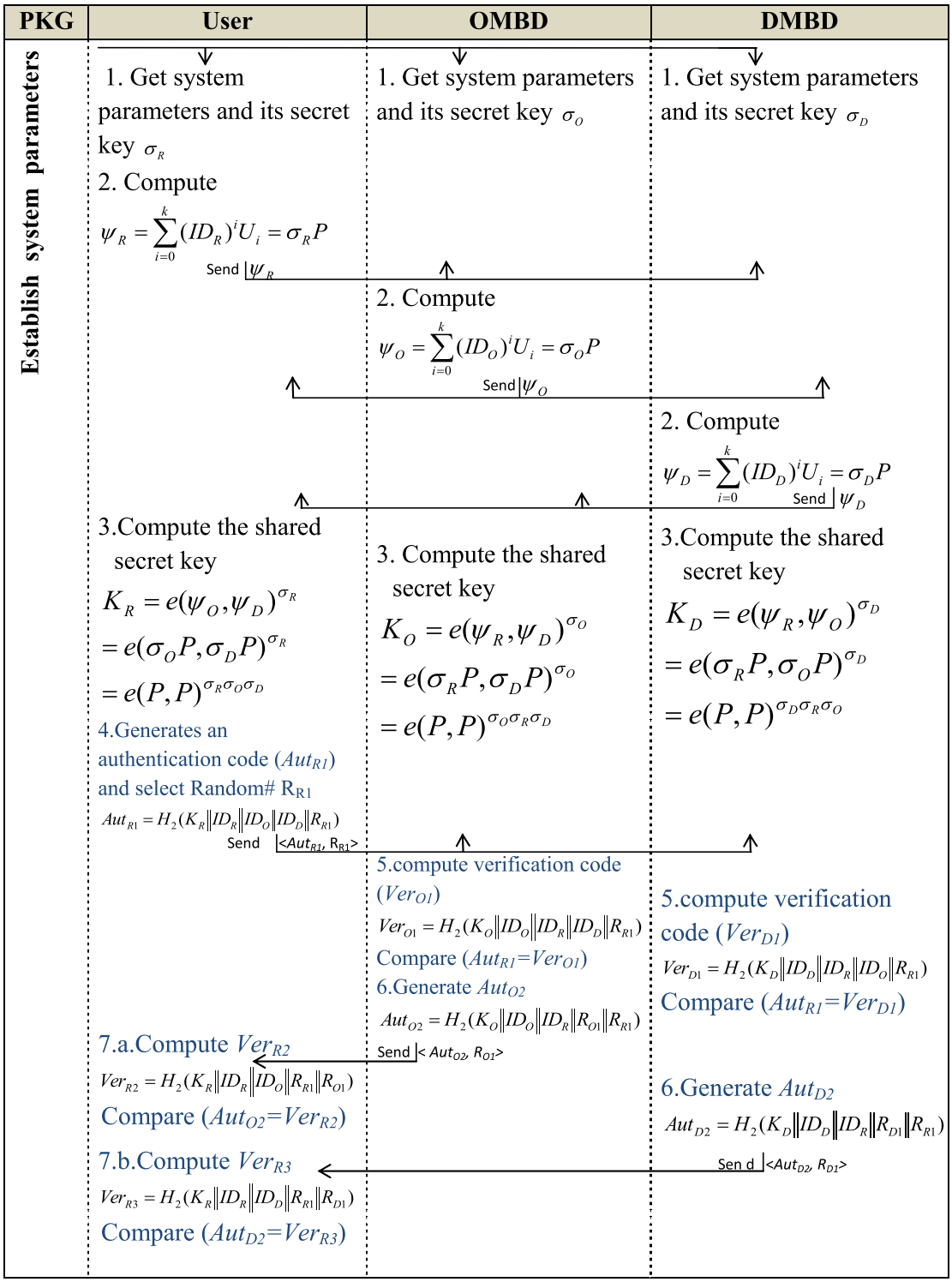

FIGURE 8. Three parties' authentication.

parameters and the parties can generate on the fly as shown in Section IV-C. In the encryption process as shown in Figure 9, the inputs would be the original photo and the encryption key. After that, based on the Blowfish algorithm length, the photo will be divided. The beginning of the array will be directly after the photo header since the header would not be encrypted. The array elements will be stored in rows, left to right ordered, with every photo scan line represented by one row, and the photo rows will be encrypted from top to bottom [43].

TABLE 2. RSA and ECC key size comparison.

Security Level (bits)	RSA Key Size (bits)	ECC Key Size (bits)
80	1024	160
112	2048	224
128	3072	256
160	4096	320
192	7680	384
256	15360	521

F. PHOTO DECRYPTION ALGORITHM

Photo decryption is the reverse of photo encryption. This process will restore the encrypted photo to what it was originally. As mentioned previously, in this process, the same photo encryption key will be used. The only difference between decryption and encryption is that supplying the sub-keys (P-array) with photo decryption is in reverse order.

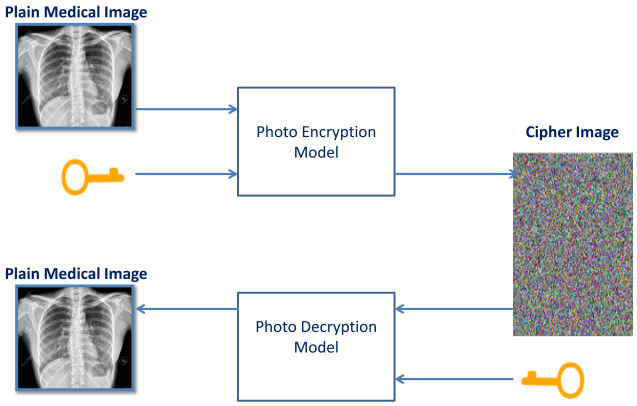

FIGURE 9. Encryption / Decryption process.

Figure 9 shows the process of photo encryption/decryption. To encrypt a photo, the two inputs are the plain photo and the key. After that, the photo will be converted into a cipher photo. For the reverse process, which is decryption, the inputs are the cipher photo and the key, and this process will convert the cipher photo to its original form, which is the plain photo.

G. ORIGINAL MBD ALGORITHM

The OMBD contains the real legitimate user's photos, for which the whole system was built to secure them. This gallery is located in the cloud. Each time the user needs to access it, he/she needs to pass the security challenge first. Moreover, each time the user uploads a new photo into the gallery, a decoy one will be added in the DMBD and the original photo will be encrypted. This was designed to make the system more secure since it has two levels of security: one is the honeypot DMBD and the other keeps the original photos encrypted while stored in the cloud.

V. JUSTIFICATION FOR SELECTING CRYPTOGRAPHIC TECHNIQUES

In this section, we will review the reasons for choosing those specific algorithms in our system. This will be done by comparing the algorithms used with other available algorithms. Based on the comparison, the vision and the reasons behind preferring them over the others will be clear.

A. WHY ELLIPTIC CURVE CRYPTOGRAPHY WAS SELECTED

Key exchange algorithm is used to ensure that the connection between the parties is secured and to let the party ensure that it is connected to the intended party. Key exchange can happen by either RSA or ECC algorithms. All of them have their strong and weak sides, as we will discuss here.

Different competitive studies have been done regarding key exchange algorithms. In our comparison, we will use the results of [44]–[47] studies. The comparison between the two algorithms will be from several angels. First of all, we will compare their key sizes. The key size used in cryptographic algorithms is measured in bits. The minimum key size that is considered strong security is 80 bits. Usually, the recommended key size that offers more security is 128 bits. Both in RSA and ECC, the private and public keys can be chosen from equal sizes. The key size has no impact on performance, but size matters when it comes to the cost of secure storage of the keys. As we can see in Table 2, ECC has the best performance in terms of using a smaller key size for the same level of security compared with RSA. Secondly, we will compare them based on the execution time for their private keys. Table 2 shows that at the 80-bit security level, RSA is reported to be 10-times slower than ECC for private key operations. For the performance at the 112-bit security level, RSA is approximately 40 times slower. That means the RSA private key operation is slower than the ECC. On the other hand, the public key operation is generally faster for RSA compared with ECC. The last comparison is about the memory used for each algorithm operation. As shown in Table 3, ECC achieves the same security level as RSA but with lower memory used.

Besides that, there are three different characteristics that were compared between these two cryptographic algorithms, namely performance, security, and space requirements. A 1024 bit RSA key has roughly the same strength as a 160-bit ECC key, and a 2048 bit RSA has about the same strength as a 210-bit ECC key. Based on this, a comparison was made between the speed and memory space of 2048-bit RSA operations and a 210 bit ECC, between 1024-bit RSA operations and a 160-bit ECC, between 768-bit RSA operations and a 132-bit ECC, and between 512-bit RSA operations and a 106-bit ECC. The results are shown in Table 4 [48].

TABLE 3. RSA and ECC key size comparison.

Security Level (bits)	Operation	RSA		ECC	
80	Public key operation	Time (Seconds)	Data Memory (Bytes)	Time (Seconds)	Data Memory (Bytes)
		0.43s	542	0.81s	282
	Private key operation	10.99s	930	0.81s	282
112	Public key operation	1.94s	1332	2.19s	422
	Private key operation	83.26s	1853	2.19s	422

TABLE 4. RSA and ECC performance, security, and space requirements comparison.

	Key Generation Time	Memory requirement	Encrypt /Decrypt Time
ECC (106 bits)	57 ms	108 bytes	11 ms
RSA (512 bits)	383 ms	157 bytes	77 ms
ECC (132 bits)	98 ms	117 bytes	17 ms
RSA (768 bits)	889 ms	236 bytes	160 ms
ECC (160 bits)	108 ms	125 bytes	16 ms
RSA (1024 bits)	2609 ms	313 bytes	338 ms
ECC (210 bits)	121 ms	140 bytes	15 ms
RSA (2048 bits)	18399 ms	621 bytes	1867 ms

B. WHY THE BLOWFISH ALGORITHM WAS CHOSEN

Now, the question that one might ask is ''why did we choose the Blowfish algorithm in particular?'' Well, many different competitive studies have been done regarding encryption algorithms to help us pick the best one. Thus, we used [49], [50] results from studies that were about the competitive analysis on different symmetric encryption algorithms, which are DES, 3DES, AES, and Blowfish. The comparison between the algorithms was based on seven criteria listed below in detail:

1) **Block size:** The relation between security and block size is positive, so the larger the block size, the more secure it is. The block size used for all of the algorithms is 64 bits except for AES, which uses 128 bits. Therefore, it is clear that, based on block size, AES is more secure than others, but at the same time it costs more to implement.
2) **Number of rounds:** This is the same as block size, so the higher the number of rounds, the more secure they are. The number of rounds for AES might be 10, 12, or 14 depending on the key length used. For DES and Blowfish, they have 16 rounds. On the other hand, 3DES has the highest number of rounds, which is 48, and that is why it is named triple DES since its number of rounds is three times more than DES.
3) **Key length:** The longest key length means a decreased likelihood of successful attacks. Based on that, Blowfish is the most secure algorithm since its key length is in the range of 32 bits to 448 bits, while DES's key length is 56 bits, 3DES's key length is either 112 or 168 bits, and finally, the AES key length might be 128, 192, or 256 bits.
4) **Encryption/Decryption time:** This is the time needed to convert plaintext into cipher text, and the reverse regarding decryption time. The algorithm that consumes the longest encryption/decryption time is 3DES, while the one that consumes the shortest time is Blowfish.
5) **Power consumption:** Regarding this criterion, 3DES consumes more power than the others, while Blowfish consumes the least.
6) **Memory usage:** Also, 3DES uses a very high memory while the memory usage of Blowfish is very low.
7) **Confidentiality:** The confidentiality of 3DES and AES is high while DES is low, but Blowfish has the highest confidentiality among them

Based on the previous comparisons, Blowfish gives a better performance than the others based on its encryption/decryption time. It is also more secure than the rest based on the key size used. Additionally, it consumes less memory and power than the others. As a result, Blowfish is the best algorithm and is thus the best candidate to be used in our proposed system.

VI. COMPUTATIONAL ANALYSIS FOR CRYPTOGRAPHIC IMPLEMENTATION

In this section, the computational analysis for cryptographic implementation of the proposed methodology is discussed. The selection of the elliptic curve and the finite field is discussed first, and then communication overhead is calculated. The computational cost of the proposed scheme is evaluated. Finally, a comparison analysis is provided between the proposed scheme and other schemes from different studies based on the computational cost.

A. CHOOSING ELLIPTIC CURVES AND FINITE FIELDS

There are two accessible options regarding choosing an elliptic curve for pairing-based cryptography; the first option is super singular curves, whereas the second option is non-super singular curves. The most required property for the selected elliptic curve is that it should have a small embedding degree denoted as k. In order to achieve 80-*bit* security level and based on Table 4, the subgroup order of $E(F_q)[l]$ should be in the range of (160 bits – 223 bits) and the size of the extension field F_{q^k} should be 1024 bits. Hence, the value of the embedding degree k should be nearly (6.4).

TABLE 5. Security strength comparison.

Security Level bits)	Integer Factorization Cryptography (bits)	Size of extension field F_{q^k} (bits)	Size of $E(F_q)[l]$ (bits)
	RSA	D-H	ECC
80	1024	1024	160-223
112	2048	2048	224-155
128	3072	3072	256-383
192	7680	7680	384-511
256	15360	15360	512+

A pairing-friendly elliptic curve over a finite field consists of the finite set of points on a curve, which can be defined by one of the below equations.

1) $E(F_{p^m}) : y^2 = x^3 + ax + b$
2) $E(F_{2^m}) : y^2 + y = x^3 + x + b$
3) $E(F_{3^m}) : y^2 = x^3 - x + b$

The three previous equations are examples of pairing-friendly elliptic curves with 80-bit security strength. In the first equation, the curve can be super singular with an embedding degree of $k = 2$ and finite field F_p. For the second equation, the finite field is F_{2^m} and the maximum embedding degree is $k = 4$ where $b \in 0, 1$. For the last equation, the finite field is F_{3^m} and the maximum embedding degree is $k = 6$ where $b \in -1, 1$. The required field size for achieving $80 - bit$ security level for all of the above the three cases are 512, 239, and 194 respectively [51], [52].

As a result, the candidate finite field for the implementation of our proposed system can be $F_{3^{97}}$ on the super singular elliptic curve $y^2 = x^3 - x + 1$ or $y^2 = x^3 - x - 1$. In the next analysis, we will use the parameter values given above, resulting in the elements in G_1 and G_2 to be roughly 160-*bit* and 1024-*bit*, respectively. Additionally, we assume that the revenue of keyed hash message authentication code is 160-bit output by using SHA-1.

B. COMMUNICATION OVERHEAD

Evaluation of our proposed system communication overhead can be done based on the sizes of the transmitted packets by the three parties (User, OMBD, and DMBD) over the communication during key generation and authentication, as describe previously in Sections IV-C and IV-D. Firstly, the communication overhead for the user as follows: (1) First packet is ψ_U with the size = ($|G_1|$ element) two times since it will be sent to the OMBD and DMBD. The second packet is ($<Aut_{R1}, R_{R1}>$) with the size ($160 - bit$ HMAC output $+160 - bit$ random number) also multiply by two since it will be sent to both OMBD and DMBD. Secondly, the communication overhead for the OMBD will be as follows: (1) First packet is ψ_O with the size = ($|G_1|$ element) two times since it will be sent to the user and DMBD. (2) The second packet is ($<Aut_{O2}, R_{O1}>$) with the size ($160 - bit$ HMAC output $+160$-*bit* random number). (3) And the third packet is ($< Aut_{O3}, R_{O2} >$) with the size ($160 - bit$ HMAC output $+ 160 - bit$ random number). Finally, the communication overhead for the DMBD will be as follows: (1) First packet is$\psi_D$with the size = ($|G_1|$ element) two times since it will be sent to the user and OMBD. (2) The second packet is ($<Aut_{D2}, R_{D1}>$) with the size (160-*bit* HMAC output + 160-*bit* random number). (3) And the third packet is ($<Aut_{D3}, R_{D2}>$) with the size (160-*bit* HMAC output + 160-*bit* random number).

TABLE 6. Comparison based on the computational cost.

Schemes	MUL	DIV	ADD	ECPM	ECPA	HASH
Hwang	1	0	1	5	1	2
Zheng	3	1	1	3	1	4
ID-based Signcryption	3	0	1	5	0	2
P2PDMS	1	0	1	1	0	5
Proposed Scheme	0	0	0	1	1	4

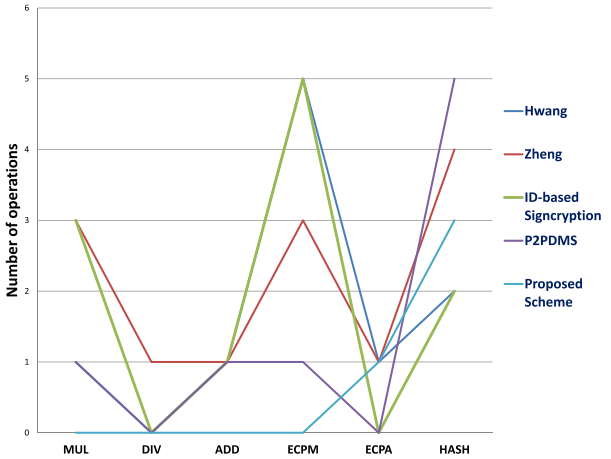

FIGURE 10. Comparison graph between the existing scheme and the proposed scheme.

C. COMPUTATIONAL COST

Our proposed scheme involves one elliptic curve point multiplication operation, one elliptic curve point addition operation, one hash evaluation on H_1, three hash evaluations on H_2, and one random numbers generation. All these calculations belong to only one party, and in our example we calculate the user operations. So, the total computational cost for all three parties together is three elliptic curve point multiplication operations, three elliptic curve point addition operations, three hash evaluations on H_1, nine hash evaluations on H_2, and three random numbers generations. We did not calculate the server part operations, which is why we have nothing regarding modular division operation, modular multiplication operation, and modular addition operation.

D. COMPARISON ANALYSIS

We compare the cost of our proposed work with some elliptic curve cryptography schemes that were listed in the study results of [51] and [53]. We have used some notations that are given below to define the number of operations and are used in Table 6 and Figure 10.

- MUL = modular multiplication operation.
- DIV = modular division operation.
- ADD = modular addition operation.
- ECPM = Elliptic Curve point multiplication operation.
- ECPA = Elliptic Curve point addition operation.

VII. CONCLUSION

As a part of securing the cloud data mission, this paper focuses on securing user's multimedia data within the cloud by using fog computing. To this end, two photo galleries are generated. The OMBD is kept secretly in the cloud and the DMBD is used as a honeypot and is kept in the fog. Therefore, instead of retrieving the DMBD only when any unauthorized access is discovered, the user, by default, accesses the DMBD. The OMBD is only accessible by a user after verifying the authenticity of the user. Thus, the original multimedia data become more secure by setting the default value of the DMBD, while the OMBD is kept in a hidden gallery. To facilitate the above process, an efficient tri-party authenticated key agreement protocol has been proposed among the user, the DPG, and the OPG based on paring cryptography.

REFERENCES

[1] M. Chen, J. Yang, Y. Hao, S. Mao, and K. Hwang, "A 5G cognitive system for healthcare," *Big Data Cognit. Comput.*, vol. 1, no. 1, p. 2, 2017, doi: 10.3390/.bdcc1010002.

[2] *Frost & Sullivan: Drowning in Big Data? Reducing Information Technology Complexities and Costs for Healthcare Organizations*. [Online]. Available: http://www.emc.com/collateral/analyst-reports/frost-sullivan-reducing-information-technology-complexities-ar.pdf

[3] M. Chen, S. Mao, and Y. Liu, "Big data: A survey," *Mobile Netw. Appl.*, vol. 19, no. 2, pp. 171–209, Apr. 2014.

[4] M. S. Hossain and G. Muhammad, "Healthcare big data voice pathology assessment framework," *IEEE Access*, vol. 4, no. 1, pp. 7806–7815, Dec. 2016.

[5] M. Chen, Y. Hao, K. Hwang, L. Wang, and L. Wang, "Disease prediction by machine learning over big data from healthcare communities," *IEEE Access*, vol. 5, no. 1, pp. 8869–8879, 2017.

[6] M. Chen, P. Zhou, and G. Fortino, "Emotion communication system," *IEEE Access*, vol. 5, pp. 326–337, 2017.

[7] M. Chen, Y. Ma, Y. Li, D. Wu, Y. Zhang, and C.-H. Youn, "Wearable 2.0: Enabling human-cloud integration in next generation healthcare systems," *IEEE Commun.*, vol. 55, no. 1, pp. 54–61, Jan. 2017.

[8] J. Bian, U. Topaloglu, and F. Yu, "Towards large-scale twitter mining for drug-related adverse events," in *Proc. SHB*, Maui, HI, USA, 2012, pp. 25–32.

[9] M. S. Hossain and G. Muhammad, "Cloud-assisted industrial Internet of Things (IIoT)—Enabled framework for health monitoring," *Comput. Netw.*, vol. 101 pp. 192–202, Jun. 2016.

[10] W. Raghupathi and V. Raghupathi, "An overview of health analytics," *J. Health Med. Informat.*, vol. 4, no. 3, pp. 1–11, 2013.

[11] M. S. Hossain, G. Muhammad, S. M. M. Rahman, W. Abdul, A. Alelaiwi, and A. Alamri, "Toward end-to-end biomet rics-based security for IoT infrastructure," *IEEE Wireless Commun. Mag.*, vol. 23, no. 5, pp. 45–51, Oct. 2016.

[12] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Proc. Grid Comput. Environ. Workshop*, Austin, TX, USA, Nov. 2008, pp. 1–10.

[13] P. T. Grance. (Oct. 2009). *The NIST Definition of Cloud Computing*. [Online]. Available: http://csrc.nist.gov/groups/SNS/cloud-computing

[14] D. Sangita, C. Ankita, and P. Reshamlal, "A review on issues and challenges of cloud computing," *Int. J. Innov. Adv. Comput. Sci.*, vol. 4, no. 1, pp. 81–88, 2015.

[15] B. A. Akyol, "Cyber security challenges in using cloud computing in the electric utility industry," Pacific Nortwest Nat. Lab., Washington, DC, USA, Tech. Rep. PNNL-21724, Sep. 2012.

[16] F. Lombardi and R. Di Pietro, "Secure virtualization for cloud computing," *J. Netw. Comput. Appl.*, vol. 34, no. 4, pp. 1113–1122, 2010.

[17] S. Zawoad and R. Hasan. (Feb. 2013). "Cloud forensics: A meta-study of challenges, approaches, and open problems." [Online]. Available: http://arxiv.org/abs/1302.6312

[18] D. Kumar and K. Morarjee, "Survey on insider data theft misuse attacks in the cloud," *Int. J. Comput. Sci. Mobile Appl.*, vol. 2, no. 2, pp. 26–29, 2014.

[19] J Shropshire, "Extending the cloud with fog: Security challenges and opportunities," in *Proc. 20th Amer. Conf. Inf. Syst.*, Savannah, Georgia, 2014, pp. 1–10.

[20] M. Kaur and M. Bharti, "Fog computing providing data security: A review," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 4, no. 1, pp. 832–834, 2014.

[21] J. Voris, J. Jill, A. Keromytis, and S. Stolfo, "Bait and snitch: Defending computer systems with decoys," Dept. Comput. Sci., Columbia Univ. Acad. Commons, New York, NY, USA, Tech. Rep., 2013. [Online]. Available: https://doi.org/10.7916/D8RN3H9S

[22] S. J. Stolfo, M. B. Salem, and A. D. Keromytis, "Fog computing: Mitigating insider data theft attacks in the cloud," in *Proc. IEEE CS Secur. Privacy Workshops*, May 2012, pp. 125–128.

[23] B. R. Singh, S. Sunanda, and Y. Moligi Sangeetha Lakshmi Kanth, "A secure framework for mollifying attacks in cloud," *Int. J. Comput. Trends Technol.*, vol. 16, pp. 204–207, 2014.

[24] S. P. Karekar and S. M. Vaidya, "Perspective of decoy technique using mobile fog computing with effect to wireless environment," *Int. J. Sci. Eng. Technol. Res.*, vol. 4, no. 14, pp. 2620–2626, 2015.

[25] M. Hajibaba and S. Gorgin, "A review on modern distributed computing paradigm: Cloud computing, jungle computing and fog computing," *J. Comput. Inf. Technol.*, vol. 22, no. 2, pp. 69–84, 2014.

[26] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st Ed. MCC Workshop Mobile Cloud Comput.*, 2012, pp. 13–16.

[27] S. M. M. Rahman and K. El-Khatib, "Private key agreement and secure communication for heterogeneous sensor networks," *J. Parallel Distrib. Comput.*, vol. 70, no. 8, pp. 858–870, 2010.

[28] D. Meffert, "Bilinear pairing in cryptography," M.S. thesis, Dept. Comput. Sci., Radboud Univ., Nijmegen, The Netherland, 2009.

[29] A. Menezes, "An introduction to pairing-based cryptography," in *Recent Trends in Cryptography*. Providence, RI, USA: AMS, 2009, pp. 47–65.

[30] D. Moody, R. Peralta, R. Perlner, A. Regenscheid, A. Roginsky, and L. Chen, "Report on pairing-based cryptography," *J. Res. Nat. Inst. Standards Technol.*, vol. 120, pp. 11–27, Feb. 2015.

[31] W. A. Jansen, "Cloud hooks: Security and privacy issues in cloud computing," in *Proc. HAWAII Int. Conf. Syst. Sci.*, Koloa, HI, USA, Jan. 2011, pp. 1–10.

[32] A. A. Aldaraiseh, D. Alomari, H. Alhamid, N. Hamad, and R. Althemali, "Effectiveness of iPhone's touch ID: KSA case study," *Int. J. Adv. Comput. Sci.*, vol. 6, no. 1, pp. 154–161, 2015.

[33] H. A. Al-Hamid and S. M. M. Rahman, "Securing photos in the cloud using decoy photo gallery," in *Proc. Int. Conf. Wireless Commun., Signal Process. Netw. (WiSPNET)*, Chennai, India, Mar. 2017, pp. 65-1–65-7.

[34] S. S. Vikas, K. Gurudatt, K. Pawan, and G. Shyam, "Mobile cloud computing: Security threats," in *Proc. Int. Conf. Electron. Commun. Syst.*, Coimbatore, India, Feb. 2014, pp. 1–4.

[35] E. Aruna, C. Prasad, and M. A. Reddy, "Securing the cloud using decoy information technology to preventing them from distinguishing the real sensitive data from fake worthless data," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, pp. 292–299, Sep. 2013.

[36] D. Patil, S. Patil, D. Pote, and N. Koli, "Secured cloud computing with decoy documents," *Int. J. Adv. Comput. Sci. Cloud Comput.*, vol. 2, no. 2, pp. 43–45, 2014.
[37] S. Khairnar and D. Borkar, "Fog computing: A new concept to minimize the attacks and to provide security in cloud computing environment," *Int. J. Res. Eng. Technol.*, vol. 3, no. 6, pp. 124–127, 2014.
[38] M. Sriram, V. Patel, D. Harishma, and N. Lakshmanan, "A hybrid protocol to secure the cloud from insider threats," in *Proc. IEEE Int. Conf. Cloud Comput. Emerg. Markets (CCEM)*, Bangalore, India, Oct. 2014, pp. 1–5.
[39] A. R. Vinod, B. S. Sunidatta, K. U. Rani, and P. P. Sasidharan, "Hindering data theft attacks through fog computing," *Int. J. Res. Eng. Technol.*, vol. 3, no. 9, pp. 427–429, 2014.
[40] W. Liu, "Research on cloud computing security problem and strategy," in *Proc. IEEE Conf.*, Yichang, China, Apr. 2012, pp. 1216–1219.
[41] A. Cufoglu, "User profiling—A short review," *Int. J. Comput. Appl.*, vol. 108, no. 3, pp. 1–9, 2014.
[42] P. Gaur and N. Manglani, "A survey on image encryption and decryption using blowfish & watermarking," *Int. J. Recent Innov. Trend Comput. Commun.*, vol. 3, no. 5, pp. 3285–3288, 2015.
[43] P. Singh, "Image encryption and decryption using blowfish algorithm in MATLAB," *Int. J. Sci. Eng. Res.*, vol. 4, no. 7, pp. 150–154, 2013.
[44] H. B. Nguyen, "An overview of the NTRU cryptographic system," M.S. thesis, Dept. Math. Stat., Univ. San Diego, San Diego, CA, USA, 2014.
[45] H. Gohel, "Design and development of combined algorithm computing technique to enhance Web security," *Int. J. Innov. Emerg. Res. Eng.*, vol. 2, no. 1, pp. 76–79, 2015.
[46] N. Gura, A. Patel, A. Wander, H. Eberle, and S. Shantz, "Comparing elliptic curve cryptography and RSA on 8-bit CPUs," in *Cryptographic Hardware and Embedded Systems* (Lecture Notes in Computer Science), vol. 3156. Berlin, Germany: Springer, 2004.
[47] A. Singh, A. K. Awasthi, and K. Singh, "A key agreement algorithm based on ECDSA for wireless sensor network," in *Proc. 3rd Int. Conf. Adv. Comput. Netw. Inform.*, 2015, pp. 143–149.
[48] N. Jansma and B. Arrendondo, "Performance comparison of elliptic curve and rsa digital signatures," Univ. Michigan College Eng., MI, USA, Tech. Rep., Apr. 2004.
[49] N. Tyagi and A. Ganpati, "Comparative analysis of symmetric key encryption algorithms," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 4, no. 8, pp. 348–354, Aug. 2014.
[50] N. Kumar and J. Thakur, "DES, AES and blowfish: Symmetric key cryptography algorithms simulation based performance analysis," *Int. J. Emerg. Technol. Adv. Eng.*, vol. 1, no. 1, pp. 6–12, Dec. 2011.
[51] S. M. M. Rahman, M. Masud, A. N. M. Noman, A. Alamri, and M. M. Hassan, "Towards secure data exchange in peer-to-peer data management systems," *Appl. Math. Inf. Sci.*, vol. 8, no. 6, pp. 2775–2787, 2014.
[52] M. Scott, N. Costigan, and W. Abdulwahab, "Implementing cryptographic pairing on smartcards," *Cryptographic Hardware and Embedded Systems* (Lecture Note in Computer Science). Berlin, Germany: Springer, 2006.
[53] B. Nayak, "Signcryption schemes based on elliptic curve cryptography," M.S. thesis, Dept. Comput. Sci. Eng., Univ. Nat. Inst. Technol., Rourkela, India, 2014.

HADEAL ABDULAZIZ AL HAMID received the B.Sc. degree in Information Technology form King Saud University, Riyadh, Saudi Arabia, in 2014, where she is currently pursuing the master's degree in information science. Her research interests include big data privacy, cloud security, information security, and machine learning.

SK MD MIZANUR RAHMAN received the Ph.D. degree in risk engineering, with a major in cyber security engineering, from the Laboratory of Cryptography and Information Security, Department of Risk Engineering, University of Tsukuba, Japan, in 2007. He was involved several years in cryptography and security engineering in the high-tech industry in Ottawa, Canada. He was a Post-Doctoral Researcher for several years with the University of Ottawa, University of Ontario Institute of Technology, and University of Guelph, Canada. He is currently an Assistant Professor with the Information System Department, College of Computer and Information Sciences, King Saud University, Saudi Arabia. He has authored around hundred scientific articles in peer reviewed renowned journals and conferences, including the IEEE TPDS, the IEEE Access, the IEEE WC, the *IEEE Communications Magazine*, the Elsevier *FGCS*, the Elsevier *JPDC*, the Wiley *CCPE*, the Wiley *SCN*, the Wiley *WCMC*, the Springer *Multimedia Systems*, and the Springer *Peer-to-Peer Networking and Applications*. He has a U.S. patent on white-box implementation for a NIST Standard Key Exchange Protocol. His primary research interests include cybersecurity, cryptography, software and network security, privacy enhancing technology, sensor networks security, application security, cloud, and the Internet of Things security. He received the IPSJ Digital Courier Funai Young Researcher Encouragement Award for his excellent contribution in IT security research from the Information Processing Society Japan. He also received the Gold Medal for the distinction marks in his undergraduate and graduate program.

M. SHAMIM HOSSAIN (SM'09) received the Ph.D. degree in electrical and computer engineering from the University of Ottawa, Canada. He is currently an Associate Professor with King Saud University, Riyadh,Saudi Arabia. He has authored or co-authored around 120 publications, including refereed IEEE/ACM/Springer/Elsevier journals, conference papers, books, and book chapters. His research interests include serious games, social media, IoT, cloud and multimedia for healthcare, smart health, and resource provisioning for big data processing on media clouds. He has served as a member of the organizing and technical committees of several international conferences and workshops. He is a member of the ACM and ACM SIGMM. He has served as the co-chair, general chair, workshop chair, publication chair, and TPC for over 12 IEEE and ACM conferences and workshops. He is a Co-Chair of the 7th IEEE ICME workshop on Multimedia Services and Tools for E-health MUST-EH 2017. He was a recipient of a number of awards including, the Best Conference Paper Award, the 2016 ACM Transactions on Multimedia Computing, Communications and Applications Nicolas D. Georganas Best Paper Award, and the Research in Excellence Award from King Saud University. He served as a Guest Editor of the IEEE Transactions on Information Technology in Biomedicine (currently JBHI), the *International Journal of Multimedia Tools and Applications (Springer)*, the *Cluster Computing (Springer)*, *Future Generation Computer Systems (Elsevier)*, *Computers and Electrical Engineering (Elsevier)*, *Sensors (MDPI)*, and the *International Journal of Distributed Sensor Networks*. He currently serves as a Lead Guest Editor of the *IEEE Communication Magazine*, the IEEE Transactions on Cloud Computing, the *Future Generation Computer Systems* (Elsevier), and IEEE Access. He is on the Editorial Board of the IEEE Access, the *Computers and Electrical Engineering (Elsevier)*, the *Games for Health Journal*, and the *International Journal of Multimedia Tools and Applications* (Springer).

AHMAD ALMOGREN received the Ph.D. degree in computer sciences from Southern Methodist University, Dallas, TX, USA, in 2002. He was an Assistant Professor of computer science and the Head of the Scientific Council, Riyadh College of Technology. He also served as the Dean of the Computer College and the Head of the Council of Academic Accreditation, Al Yamamah University. He is currently an Associate Professor and also the Vice Dean for development and quality with King Saud University. His research areas of interest include networking, security, mobile computing, and data consistency.

ATIF ALAMRI is currently an Associate Professor with the Software Engineering Department, College of Computer and Information Sciences, King Saud University. Riyadh, Saudi Arabia. His research interests include multimedia assisted health systems, ambient intelligence, and service-oriented architecture. He serves as a program committee member for many conferences in multimedia, virtual environments, and medical applications. He was a Guest Associate Editor of the IEEE Transactions on Instrumentation and Measurement, a Co-Chair of the first IEEE International Workshop on Multimedia Services and Technologies for E-health, a Technical Program Co-Chair for the 10th IEEE International Symposium on Haptic Audio Visual Environments and Games.