IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 8 December 2022, accepted 2 January 2023, date of publication 5 January 2023, date of current version 12 January 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3234932

RESEARCH ARTICLE

Secure, ID Privacy and Inference Threat Prevention Mechanisms for Distributed Systems

TAHANI HAMAD ALJOHANI[1] AND NING ZHANG[2]
[1]College of Computer and Information Sciences, Princess Nourah Bint Abdul Rahman University, Riyadh 11564, Saudi Arabia
[2]School of Computer Science, The University of Manchester, M13 9PL Manchester, U.K.

Corresponding author: Tahani Hamad Aljohani (thalgihani@pnu.edu.sa)

ABSTRACT This paper investigates facilitating remote collection of a patient's data in distributed system while protecting the security of the data, preserving the privacy of the patient's ID, and preventing inference attack. The paper presents a novel framework called SPID stand for a Secure, ID Privacy, and Inference Threat Prevention Mechanisms for Distributed Systems. In designing this framework, we make the following novel contributions. The SPID presents a novel architecture that supports the use of a distributed set of servers owned by different service providers. The SPID allows the patient to access these servers using certificates generated by the patient. The SPID allows the patient to select one server to be the home server, and select a number of servers to be the foreign servers. The patient uses the foreign servers to upload data. The home server is responsible for collecting the patient's data from the foreign servers and sending them to the healthcare provider. The SPID proposes a method for efficient verification of each request from the patient without searching in the server's database for the verification key. This is done by using some of the Elliptic Curves Cryptography (ECC) properties. The SPID has been analyzed using a bench-marking tool and evaluated using queuing theory. The evaluation results indicate an efficient performance when the number of servers increases. We uses Shannon entropy method to measure the likelihood of the inference attack.

INDEX TERMS Security, ID privacy, distributed authentication, elliptic curves, inference attack.

I. INTRODUCTION

The Internet of Things (IoT) can be defined as the paradigm of connecting smart things (e.g. sensors, devices) together by means of information and communication technologies to build intelligent systems and services to obtain required information. The smart things can sense the surrounding environment and communicate with each other to exchange information. These features (i.e. sensing and communicating) facilitate many emerging attractive applications. One of these applications is Patient Health Monitoring (PHM) systems. A PHM system involves the use of mobile computing and wireless communication technologies to regularly collect data from a patient for the purpose of analyzing the patient's health and making health-related decisions [1], [2].

A typical PHM system, as shown in Figure 1, consists of body sensors and a mobile or fixed device at the patient's end (e.g. home) and remote servers at the healthcare provider's end. The body sensors worn by the patient are connected wirelessly to the device that is, in turn, connected to the servers via wireless and/or wired networks. The health-related data (e.g. heart rate, blood pressure, etc) collected by the body sensors are sent to the device, which are then delivered to the healthcare provider for further analysis and decision making. The operations performed by a PHM system are typically of three stages: data collection, data analysis (i.e., the analysis of the collected data), and decision

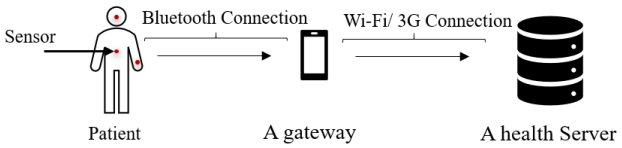

FIGURE 1. A typical PHM system.

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

making (based on the outcome of the data analysis). The data collection stage is crucial to the correct running of a PHM system, as the correctness of the analysis and decision making are dependent on the correctness of the data collected [3], [4].

It is expected that future PHM systems will be built on infrastructure owned by a third-party service provider which has more resourceful storage and data processing capabilities, such as Microsoft [5], [6]. This is because on-premises infrastructures, on which most healthcare providers rely today, might not be able to handle the volume of data generated from wearable devices. It is estimated that by 2021 more than 222 million wearable devices may have been poured onto the market and three in five patients may use remote monitoring services. These devices can generate data at high frequencies (e.g. every 5 minutes), generating massive volumes of data. When the patient's data is collected by the third-party service provider, a number of security and privacy threats will be brought up. These threats are authentication, data confidentiality and authenticity, ID privacy, and inference threats. The last threat comes from using one service provider.

The inference attack [7] occurs when an unauthorized entity (e.g. a third-party service provider) can use some attributes such as the pattern of communication (e.g. how many times per day a patient uploads data to the service provider) to link multiple transactions to the same user. For example, a patient uploads some data (such as blood pressure) at 1 pm every day and other data (such as heart rhythm) every 10 minutes. Then, even if that patient uses a different pseudonym in each transaction, over time and by observing the upload pattern (i.e., 1 pm and every 10 minutes), the service provider may infer that all the data uploaded at this time belongs to the same patient. This is considered as a threat to the patient's privacy. Many research [8], [9], [10], [11], [12], [13], [14], [15] devoted to investigate in mechanisms that prevent and detect authentication, data confidentiality and authenticity threats.

To prevent the inference attack some research has suggested using group signature schema and a broadcasting strategy to protect contextual privacy. Boussada et al [12] present a privacy-preserving aware data transmission protocol that preserves the privacy of a patient's data and the contextual data. To preserve the patient's data privacy, the patient's health data is encrypted and to preserve the contextual privacy (to prevent the inference attack) the protocol used pseudonym IDs combined with a broadcasting strategy. Liang [14] propose a privacy-preserving emergency call scheme, called PEC, enabling patients in life-threatening emergencies to transmit emergency data to the nearby helpers via mobile healthcare social networks (MHSNs). In PEC the ID of the patients is preserved via using group signature. Lin et al [15] proposed a strong privacy-preserving scheme against global eavesdropping, named SAGE, for eHealth systems. SAGE uses a broadcasting strategy to prevent an adversary from linking patients to their respective physicians. Marin et al [16] proposed a secure and ID privacy-preserving data collection protocol. The protocol considers a number of security and ID privacy-preserving requirements. To protect the ID privacy of the patient they used a combination of group signature and pseudonym IDs. However, the reliance on a fixed data concentrator for a group of patients would make it easy for that entity to analyze the meta-data associated with the messages and over time cloud build sensitive information about each patient.

In this paper we are going to present an innovative framework called Secure, ID Privacy and Inference threat prevention mechanisms for Distributed Systems (SPID). The SPID uses distinguished mechanisms that consists of a system architecture and software to solve the problem of the inference threat and prevent other threats. The idea is to allow the patient to use different service providers to collect his/her data rather than relying on one provider. Regarding the system architecture, our system architecture consists of a healthcare provider (HCP) and a number of service providers (SPs) (e.g. Amazon, Google, Yahoo). These service providers should first communicate with the HCP and request to participate in the health data collection service to be authorized. The patient can upload the health data to any authorized SP. Each SP is responsible for storing the data temporally and then send them to the HCP for further analysis and decision. The SP is not allowed to know any thing about the patient's health data and the patient's real ID. In more details, the patient can select a number of authorized SPs every day. One of the SPs is going to be a home provider and the other ones are the foreign providers. The patient uses foreign providers to upload the encrypted data. The home provider is responsible for collecting the patient's data from the foreign providers, aggregating and sending them to the HCP. The set of SPs is not static; it is dynamic. This means that the patient has to change the home and foreign servers every day. The HCP need not to know anything about the selected SPs (i.e, do not know the patient's home and foreign servers). The patient uses a swarming algorithm to upload on the foreign servers (the detailed designed of this algorithm will not be covered in this paper). Moreover, each SP can ensure the authenticity of the patient and his/her data. The HCP can also ensure the authenticity of the patient's data. Regarding the software components, SPID allows the patient to generate pseudonyms and certificates to access foreign servers. The certificates are generated by the patient and signed blindly by the patient's home server. To protect against the data authenticity and confidentiality threats, the patient encrypts and signs the data before uploading. The SPID framework has been analyzed using a bench-marking tool and evaluated using queuing theory. The evaluation results indicate an efficient performance when the number of service providers increases. We uses Shannon entropy method to measure the likelihood of the inference attack.

II. RELATED WORKS

The related works is divided in to two board categorise one to study the health data collection systems and the other one examine the authentication in distributed systems. Regarding

the health data collection systems, in [8] the authors proposed a federated cloud and Internet of Things (IoT) health monitoring system. The system fails to satisfy a number of security and privacy requirements, such as ensuring patients' data confidentiality and authenticity, and patients' ID privacy. Similar to [8], authors in [9] proposed a secure fog-based smart health service. The patient data is collected by heterogeneous fog layers. This system supports data encryption and confidentiality but the patients' ID privacy was not considered. In [11], the authors proposed a privacy-preserving priority classification scheme, called PPC. The proposed system does not consider the ID privacy of the patient. The authors in [12] presented a privacy-preserving aware data transmission for IoT-based health applications. The proposed system preserves the privacy of a patient's data and the contextual data. To preserve contextual privacy the proposed system uses a building path algorithm and broadcasting strategy. The authors in [15], proposed SAGE which stands for Strong privacy-preserving scheme Against Global Eavesdropping for e-health system. SAGE protects the confidentiality of the health data and cuts off the relationship between the patient and his/her physician. SAGE preserves the content and contextual privacy. This is done by first encrypting the health data and then sending the encrypted data to a health centre. The centre is responsible for broadcasting the data to all physicians. Then, only the potential physicians will be aware of the data of their patients. In [17], the authors proposed a health remote monitoring system based on the blockchain technology called Healthchain. The architecture of the system supports large-scale health data and has good scalability. In Healthchain, the authors neglect preserving the patient's ID privacy. In [18], the authors proposed a decentralized privacy-preserving healthcare blockchain for IoT. In this work, the patient uploads confidential and authenticated data on a cloud server. The cloud server verifies the data, generates a data block, generates a hash for the data block, and sends the hash of the data block to the blockchain. For anonymous transactions, the authors proposed a ring signature scheme. Each block has information about the sender (the patient) and the receiver (doctor). Thus, the blockchain network can link the patients to their doctors. In this work, each time the patient is uploading to the same server.

The architecture of the proposed framework SPID resembles the architecture of Federated Identity Management (FIM) systems [19]. The FIM system has several models: the centralized identity model, the user-centric identity model, and the decentralized identity model. The centralized identity model [19] relies on a central entity to conduct the authentication between two entities. Shibboleth [20] is an example of federated identity management systems. One of the early federated identity management systems that applied this approach was PseudoID [21]. In this work, the identity provider issues the user a number of credentials that are signed blindly using the identity provider's private key. To prevent the identity provider from tracking the user, the author [22] presented a solution called a Crypt-book. The Crypt-book is a layer between a number of identity providers and service providers, which hides the real identity of a user from the service providers. In user-centric identity model, the user is involved in generating credentials to access different service providers. In [23], the user performs authentication with different service providers based on a set of certified attributes issued by the user's identity provider.The Blank Digital Signature (BDS) allows the user to generate a signature on a subset of attributes which can be verified by the service provider. A framework proposed by [24] is called the SPICE framework. In this framework, the authors presented a novel authentication mechanism where the main service provider issues only one credential to each user no matter how many service providers the user wants to use. For authentication, the user generates (based on the credential) many certificates to prove the possession of (different sets of) attributes required by different service providers, without asking the registrar to issue a new certificate each time. In the decentralized identity model, the power is given to a collection of nodes that use distributed ledger technology (DLT) to store identity information about users where no node can change the content of DLT. The trust between nodes is reached by a consensus mechanism. The distributed DLT is built using blockchain technology. In Trustroam [25], the authors proposed an authentication method to provide cross-domain roaming authentication. This means that a user from institute A can use his/her identity credentials to access a network of institute B. The authentication of this proposed solution is based on distributed consensus algorithms of the blockchain. However, this method allows institutes to link different accesses by the same user as the user is using the same identity across domains. To solve the linkability problem, the authors proposed blockchain lightweight anonymous authentication (BLA) [26]. The proposed mechanism allows a vehicle to access services across distributed domains. This mechanism allows the vehicle to use different pseudonyms each time to prevent linking multiple requests sent by the same vehicle.

From the critical analysis of the related works, we can find that none of the related works provide a system that prevents all the threats (i.e. security, ID privacy, and inference threats). Therefore, this paper presents an innovative framework SPID that prevents all the threats with an acceptable performance.

III. SPID ARCHITECTURE

In this section we present SPID architecture. The SPID architecture is different from the one shown in Figure 1. The generic and detailed SPID architectures are shown in Figures 2 and 3 respectively. The SPID architecture consists of the HCP and different service providers.

A. SPID COMPONENTS AND INTERACTIONS

- Service Providers: Each service provider plays two roles: home and foreign providers. This means that the same service provider may act as a home provider for one patient, but a foreign provider to another patient.

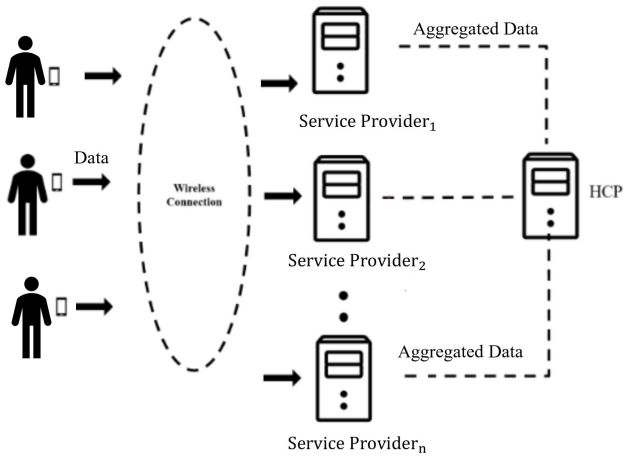

FIGURE 2. A generic SPID architecture.

The foreign provider is used by the patient to upload the health data. The foreign provider forwards the health data to the patient's home provider. The home provider is responsible for forwarding the data after aggregating them to the HCP. It should be noted that the home and foreign providers for the patient are not static but dynamic. Figure 3 illustrates the SPID architecture from the patient's perspective i.e., the idea of the home and foreign providers. In the rest of this paper we may refer to the home provider or foreign provider as a home server and a foreign server respectively or data collection servers.

- The Healthcare Provider Server (HCP): This runs the business logic of the system. It is the ultimate destination for all patients' data. It is where this data is stored, processed, and analyzed. The HCP receives the patients' data from their respective home providers. In addition, the HCP issues a certificate for the service provider that want to participate in the data collection service. Then, the HCP adds the service provider to its open access database.
- The Patient: The patient wears small devices integrated with low-power computation, communication and storage modules to measure and monitor health data (e.g. ECG, blood pressure) from the patient. The health data is sent to a mobile device (i.e. via Bluetooth) for processing. The mobile device classifies the data into normal or abnormal data, and structures the data into a predefined format, called Patient-Generated Health Data (PGHD). The PGHD is uploaded to foreign providers. In details, each patient registers with one service provider which is called the patient's home provider. The other providers are called the patient's foreign providers. The patient changes the home and foreign providers daily. In the rest of this paper we may use a patient or mobile device interchangeably.

IV. THREATS ANALYSIS

In this section we analyze the potential threats on SPID architecture.

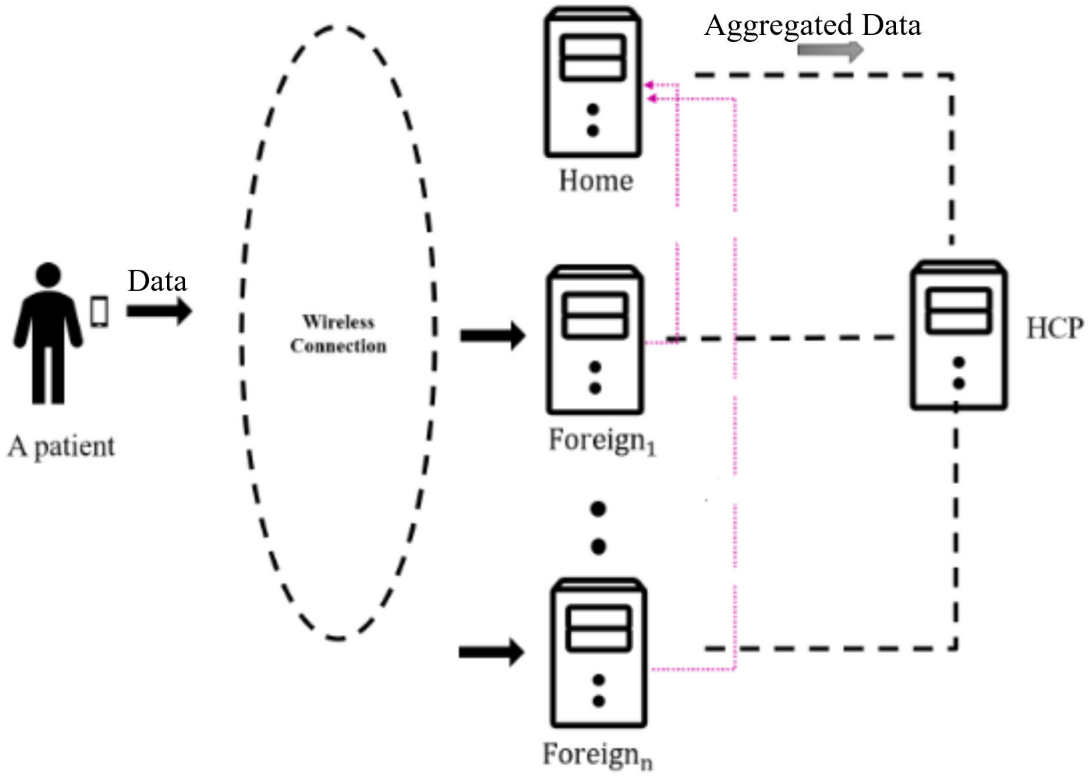

FIGURE 3. A detailed SPID architecture Each patient has one home and many foreign servers.

- Authentication threats: A malicious service provider or an external adversary may try to impersonate a legitimated service provider or patient respectively to gain unauthorised access to the patient's data.
- Data authenticity threats: The patient's data may be delayed, replayed, or even modified. A malicious service provider or an external adversary may try to forge data to make it seems as it is from a legitimate patient.
- Data confidentiality threats: If the patient's data is not protected during transit or in store, a malicious service provider or an external adversary may gain access to the patient data, e.g. by eavesdropping the channel.
- Patient's ID privacy threats: Even if the patient is using different artificial IDs when registering with different service providers, the patient's sessions with the same service provider may be linked if the patient does not change the artificial ID for each session.

A. DESIGN REQUIREMENTS

Here we specify a set of design requirements which facilitate the design of a secure and ID privacy-preserving protocols for SPID framework. The SPID protocols should provide strong protection against predefined threats.

- The system should preserve patient's ID privacy (P): To satisfy this requirement, only the HCP can learn the real ID of the patient. We should prevent linkability across different service providers, each service provider should identify the patient under a pseudonym ID. We should prevent linkability across sessions with the same service provider, the patient should use different pseudonym IDs for each session.
- The system should support mutual entity authentication(S1): Entity authentication ensures that a communicating entity is indeed who it claims to be. This requirement should be satisfied without compromising the patient's ID privacy.
- The system should support end-to-end data authenticity (S2): Data authenticity assures that data are indeed from the claimed source and that it is the same as has been sent by the original sender. Each service provider should ensure that the uploaded data is authenticated without compromising the ID privacy of the patient.
- The system should support end-to-end data confidentiality(S3): This requirement is to protect against unauthorized access to data in transit or on store.

B. HIGH LEVEL IDEAS

In this section we list a number of novel ideas to satisfy the design requirements.

- Design multiple index pseudonyms and multiple request pseudonyms to satisfy ID privacy preservation requirements.
  - First, with each provider, the patient should have a pseudonym ID called the 'index pseudonym'. The index pseudonym serves as an account name. This is to prevent multiple providers from colluding together to identify the patient.
  - Second, for each request with the same provider, the patient should use a fresh pseudonym called a request pseudonym. The request pseudonym should be generated based on the index pseudonym. This is to make the linking process feasible. The pseudonym generation and linkage algorithms are presented in Section VI.
- Design double signing and encryption method to satisfy the security requirements (S2, S3): This method is explained in Section VI.
- Design an anonymous authentication method to satisfy the security requirements (S1): To ensure authorized use of the data collection service, the patient should be identified and authenticated before s/he is allowed to access the collection service and this should be achieved without compromising the patient's ID privacy. To support such authentication in a seamless and scalable manner, we designed two anonymous authentication credentials as follows.
  - A Home Pseudonym Certificate (HCert) for authenticating the patient with the home provider and Foreign Pseudonym Certificates (FCerts) for authenticating the patient with foreign providers. Figure 4 and Figure 5 shows the HCert and FCert respectively.
  - The HCert is generated by the HCP to allow the patient to register with any provider as a home provider.
  - The FCert is generated by the patient and blindly signed by patient's home provider. The FCert allows the patient to use any foreign provider.

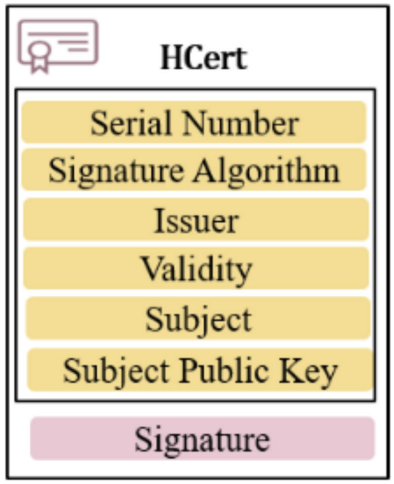

FIGURE 4. The home pseudonym certificate.

C. NOTATION

The notation used throughout the paper is summarized below.

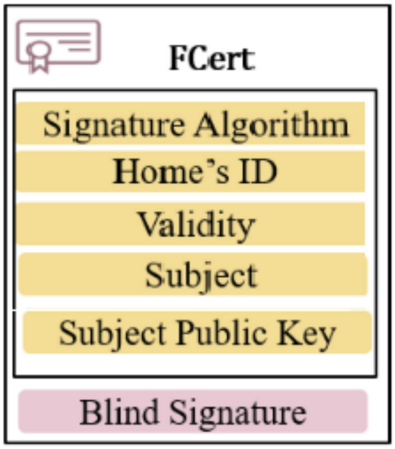

FIGURE 5. The foreign pseudonym certificate.

HCP:	The healthcare provider.
PID_i:	Patient Real ID for patient *i*.
HIP_i :	Home Index Pseudonym for patient *i*.
FIP_i^f :	Foreign Index Pseudonym for patient *i* with a foreign server *f*.
$HRP_{(i,r)}$:	Home Request Pseudonym for a patient *i* and a session *r*.
$FRP_{(i,r)}^f$:	Foreign Request Pseudonym for a patient *i* with a foreign server *f* and session *r*.
RPK_e:	The RSA Public Key for entity e.
RPR_e :	The RSA Private Key for entity e.
EPK_e:	The Elliptical Curve Public Key for entity e.
EPR_e:	The Elliptical Curve Private Key for entity e.
SK_i:	The Shared Key between patient *i* and HCP.
HSK_i:	The Home Shared Key for patient *i* with a home server.
FSK_i^f:	The Foreign Shared Key for patient *i* with a foreign server *f*.
tPK_i^f,tPR_i^f:	Temporal Public and Private Keys generated by patient *i* with a foreign server f.
Enc, Dec:	Asymmetrical Encryption and Decryption.
E, D:	Symmetrical Encryption and Decryption.
T :	Time stamp.
Inc:	Index cryptographic nonce.
Rnd :	Random value.
$2EMPGHD_i$:	Double Encrypted and MAC Patient Generated Health Data.

V. CRYPTOGRAPHY BUILDING BLOCKS

In this section we introduce the cryptographic building blocks which are used in designing our methods and protocols. We use Rivest, Shamir, Adleman (RSA), Advanced Encryption Standard (AES), Elliptic Curve Cryptography (ECC), blind signature based on ECC, and the digital certificate [27]. We here present only the ECC and the blind signature [7], [28].

A. ELLIPTIC CURVE CRYPTOGRAPHY (ECC)

The following defines the elliptical curve field and equation. The elliptic curve over a finite field : $\mathbb{Z}_p$, p > 3, is the set of all pairs (x,y) in $\mathbb{Z}_p$ which satisfies the equation $E : y^2 \equiv x^3 + ax + b \pmod n$ where $a, b \in \mathbb{Z}_p$, and the following condition should be satisfied $4a^3 + 27b^2 \neq 0 \pmod n$.

The domain parameters which define the elliptic curve are $(t,a,b,\mathbb{P},n)$, where n is the module prime, a and b are coefficients of the elliptic curve equation, $\mathbb{P}$ is the generator point, and t is the number of points in the field.

The ECC comprises four algorithms: a key generation algorithm (EKeyGen), a key exchange and agreement algorithm (EKeyExg), a signature generation algorithm (ESigGen) and a signature verification algorithm (ESigVer). We assume two entities A and B execute the following algorithms:

1) EKeyGen

- A chooses a random private key (EPK), where $0 < EPR < t$
- Computes a public key, EPK as EPK= EPR*$\mathbb{P}$, where the operation (*) means multiplication.

2) EKeyExg

- Both entities A and B exchange their public keys, EPK_A, EPK_B
- A and B compute their shared secret key (S) as follows, $S = EPR_A * (EPK_B) = EPR_B * (EPK_A)$.

3) ESigGen

- A selects an integer as a random private key (EPR_r), where $0 < EPR_r < t$
- A computes $R = EPR_r * \mathbb{P} = (x_R, y_R)$, Let $r = x_R$.
- A computes a signature on a message (x) as $s \equiv (h(x) + EPR * r) * EPR_r^{-1} \pmod n$. The h(x) is a hash function.

4) ESigVer

- A verifier computes an auxiliary value w, $w \equiv s^{-1} \pmod n$.
- The verifier computes an auxiliary value $u1$, $u1 \equiv w * h(x) \pmod n$.
- The verifier computes auxiliary value $u2$, $u2 \equiv w * r \pmod n$.
- The verifier computes $Q = u1 * \mathbb{P} + u2 * EPK_B$. This results in a point (x_Q, y_Q).
- The verifier checks if $r = x_Q \pmod n$ signature is valid.

B. ECC BLIND SIGNATURE

1) BLIND MESSAGE (BlndMsgGen) ALGORITHM

Here a sender wants to use the proxy blind signature service provided by the proxy signer. The sender first generates a

blinding value (R^*) using Equation (1), where a, b, and c are random blinding factors. It then computes a hash value of the message (m) using Equation (2). It then blinds the hash value e^* using Equation (3). After that it sends a request for a blind signature on the hash value to the proxy signer.

$$R^* = a * R_p + c * \mathbb{P} - b * EPPK_p \quad (1)$$

$$e^* = h(R^*||m) \quad (2)$$

$$e = a^{-1}(e^* - b) \mod n \quad (3)$$

2) BLIND SIGNATURE GENERATION (BlndSigGen) ALGORITHM

Once the request is received by the proxy signer, it generates the blind signature (S^*) on (e) using Equation (4). Then it sends the blind signature (S^*) to the sender.

$$S^* = e * EPPR_p + k_p \mod n \quad (4)$$

3) BLIND SIGNATURE DRIVEN (BlndSigDrv) ALGORITHM

After receiving the blind signature (S^*) from the proxy signer, the sender derives the unblind version of signature (S) using Equation (5). This signature (S) can be proven by any verifier using Equation (6).

$$S = S^* a + c \mod n \quad (5)$$

4) THE BLIND SIGNATURE VERIFICATION (BlndSigVer) ALGORITHM

This algorithm is executed by a verifier. After receiving the blind signature, the verifier verifies the proxy blind signature using Equation (6).

$$e^* = h((S * \mathbb{P} - e^* EPPK_p)||m) \quad (6)$$

VI. SPID METHODS

In this section we present a number of methods that used in designing the SPID protocols.

A. DOUBLE SIGNING AND ENCRYPTION

To satisfy the data confidentiality and authenticity requirements we designed the double signing and encryption method. The method (as shown in Figure 6) can be explained as follows. The patient's data, which is uploaded to several foreign servers, has two forms. The first is Encrypted Message Authenticated Code and Patient Generated Health Data (EMPGHD), and the second form is Double Encrypted Message Authenticated Codes and Patient Generated Health Data (2EMPGHD). In EMPGHD, the patient generates a MAC on PGHD using the shared key (SK_i). This shared key is known to the patient and the HCP. Then, the patient encrypts both the PGHD and MAC with the same key. In 2EMPGHD, the patient generates the MAC on the EMPGHD using the home shared key (HSK_i) which is a key known to the patient and his/her home server. Then, the patient encrypts both the MAC and the EMPGHD using the home shared key. The result is the 2EMPGHD that will be uploaded onto different foreign servers. By using this method, the patient's data is kept confidential and authenticated through encryption and MAC generation, respectively. Only the HCP can learn the unencrypted form of the patient's data and verify the authenticity of the data. Only home server can verify the authenticity of the patient's data (i.e., EMPGHD) that is forwarded by foreign servers. By using this method, the HCP and the patient ensure that no entity can generate data on behalf of the patient and no entity can learn anything about the patient's data.

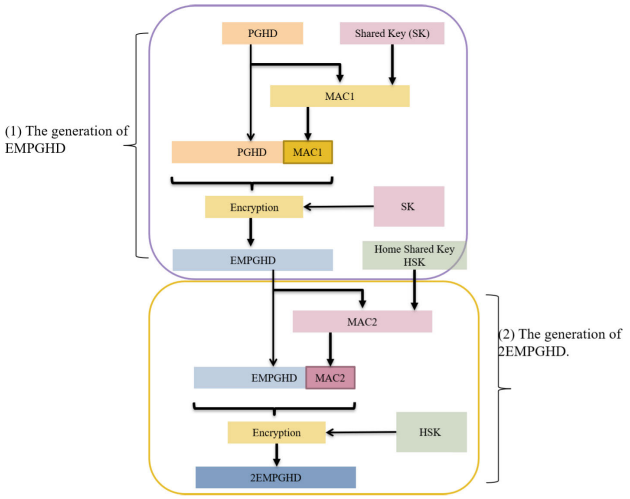

FIGURE 6. The sign then encrypt method.

B. PSEUDONYM GENERATION AND LINKAGE

To satisfy the ID privacy and mutual authentication requirements, we designed pseudonym generation and linkage algorithms. We first explain the type of pseudonyms used in SPID. Then, we explain how to generate and link them. There are four types of pseudonym and they are as follows. The Home Index Pseudonym (HIP), the Foreign Index Pseudonyms (FIPs), the Home Request Pseudonyms (HRPs), and the Foreign Request Pseudonyms (FRPs). The following algorithms show who, how to generate, and link the pseudonyms.

1) HIP ALGORITHMS

The HIP generation (HIP-Gen) algorithm is executed by the patient. In this algorithm, the patient generates an elliptical curve public key (EPK) using the ECC key generation algorithm. This EPK will be the patient's HIP known to both the home server and HCP. The HIP linkage (HIP-Lnk) algorithm is executed by the HCP. In this algorithm, the HCP links the HIP to the patient's real ID.

2) FIPs ALGORITHMS

The FIPs generation (FIPs-Gen) algorithm is executed by the patient using an RSA encryption and home server's RSA public key (RPK_h). The inputs to FIPs-Gen are the public key of the home server (RPK_h), the HIP, a random number (Rnd), and the current time (T). The output from the FIPs-Gen is a foreign index pseudonym (FIP_i^f), i.e., FIP_i^f = Enc (RPK_h,

HIP_i || T || Rnd). Where Enc means encryption using RSA and (||) is the concatenation symbol.

The FIPs Linkage (FIPs-Lnk) algorithm is the reverse process of FIPs-Gen executed by a patient's home server, i.e., HIP_i || T || Rnd = Dec (RPR_h,FIP_i^f). Where Dec means decryption using RSA.

3) THE HRPs ALGORITHMS

The HRPs generation (HRPs-Gen) algorithm is executed by the patient. The HRPs-Gen is used to generate a fresh HRP for each request carried out by the patient with his/her home server. The inputs to HRPs-Gen are the HIP, the RSA public key of home server (RPK_h), an index nonce (Inc), a random number (Rnd), and a priority tag (PR), i.e., $HRP_{i,r}$ = Enc (RPK_h, HIP_i || PR || Rnd || Inc). The priority tag helps to prioritize the data.

The HRPs linkage HRPs-Lnk alogrithm is the reverse process of HRPs-Gen executed by the patient's home server to link each HRP back to its home index pseudonym (HIP), i.e., HIP_i||PR || Rnd ||Inc = Dec (RPR_h,$HRP_{i,r}$).

4) FRPs ALGORITHMS

The FRPs generation (FRPs-Gen) algorithm executed by the patient to generate a fresh FRP for each uploading request carried out by the patient with the foreign server. The inputs to FRPs-Gen are a string format of the temporal public key ($stPK_i^f$), an RSA public key of foreign server (RPK_f), index nonce (Inc), a random number (Rnd), and the priority tag (PR), i.e., $FRP_{i,r}^f$ = Enc (RPK_f, tPK_i^f ||PR ||Rnd|| Inc). The FRPs-Lnk is the reverse process of the FRPs-Gen executed by the patient's foreign server, $stPK_i^f$|| Pr || Rnd || Inc = Dec (RPR_f,$FRP_{i,r}^f$).

VII. SPID PROTOCOLS

In this section we will use the SPID methods and cryptography algorithms to design the SPID protocols. In this paper we only going to cover the following protocols. The FCert Generation (FCertG), Foreign Server Registration (FSR), and data uploading protocols.

A. ASSUMPTIONS

We have the following assumptions

- The HCP initializes the system by establishing the domain parameters which define the asymmetrical, symmetrical, and elliptic curve cryptosystems. All the entities of the system download these parameters from the HCP. Each patient, along with each service provider (SP), generates an ECC public/private key pair, EPK_i/EPR_i and EPK_{sp}/EPR_{sp}, respectively. The public keys are signed by the HCP and certified in the form of digital certificates. In addition, each SP generates RSA public/private key pairs, RPK_{sp}/RPR_{sp} of different sizes (i.e., 15360 and 2048). These public keys are signed by the HCP and certified in digital certificates. A list of valid SP certificates is stored in an open-access database. The patient's mobile device can access and download a number of certificates for the SPs with which it wants to establish communication. It should be noted that, even if the SP is an intruder, it can not learn anything about the patient's data as the data is uploaded encrypted.
- The patient has registered with the HCP using the home index pseudonym (HIP) and the HCP has issued to the patient the home pseudonym certificate (HCert).
- The mobile device has a software agent that randomly selects a number of service providers, one is assigned as the home provider and the other ones are the foreign providers. The home and foreign servers are dynamic for each patient (i.e., the home provider for the patient can be the foreign provider in the next day).
- The patient has registered with the home provider using the HCert and by using the EKeyExg algorithm, the patient and the home server has generated a home shared key (HSK)

B. FCert GENERATION (FCertG) PROTOCOL

In this protocol the patient is generating a number of foreign pseudonym certificates (FCerts) and needs the home server to sign them blindly. The patient uses the FCerts to access foreign providers. The protocol below explains how the patient generates one FCert.

(1) The patient's mobile device performs the following.

- It generates a temporal elliptic curve key pair public and private key respectively (tPK_i^f, tPR_i^f) using the EKeyGen algorithm.
- It then generates a home request pseudonym ($HRP_{i,r}$) using the HRP-Gen algorithm. It also generates the foreign index pseudonym (FIP) using FIPs-Gen algorithm.
- It generates the other data structure fields of the FCert which are the signature algorithm, the home ID, and validity. Then it uses the BlndMsgGen algorithm to generate blind factors, hash the FCert data structure fields and then blind the hash result. The FCert data structure fields are the signature algorithm, the home ID, the validity, and the subject which is a foreign index pseudonym (FIP).

(2) Then, the mobile device constructs the blind signature request message (BlndSigReq) which contains the home request pseudonym ($HRP_{i,r}$), the ID of the home server (ID_h), the blind hash (e). After that, the mobile device generates the MAC on the message using the home shared key (HSK_i). Then it sends the message to the home server, i.e., BlndSigReq = (ID_h || $HRP_{i,r}$|| e || MAC).

The home server receives the request and do the following.

(3) The home server uses the HRP-Lnk to learn the home index pseudonym. Then, to find the home shared key (HSK_i) which is used to verify the MAC, the home server multiplies the home index pseudonym (elliptical curve public key) with its elliptical private key (EPR_h).

The result of this multiplication is the home shared key. This key is used to verify the MAC.

(4) If the MAC is successfully verified, the home server validates the index nonce. This validation guarantees the freshness of the home request pseudonym (to protect against replay attacks). It then checks that the HCert of the patient ($HCert_i$) has not expired. If both verifications are correct, the home server moves to the next step.

(5) The home server signs the blind hash (e) using the BlndSigGen algorithm.

(6) The home server constructs the blind signature response message (BlndSigRes), which contains the patient request pseudonym ID ($HRP_{i,r}$), its Id (ID_h), and blind signature (S^*). Then it generates the MAC on the message using (HSK_i) and sends the response to the patient's mobile device. i.e. BlndSigRes=(ID_h || $HRP_{i,r}$|| S^* || MAC).

(7) The mobile device receives the BlndSigRes from the home server and do the following. It verifies the MAC associated with the message using his/her (HSK_i). Then, it extracts the blind signature (S^*).

(8) Subsequently, the patient uses the Blind Signature Driven (BlndSigDrv) algorithm to deduce the unblind signature.

(9) Finally, the patient attaches the unblind signature in the signature field of the FCert. The certificate is now ready to be used. The patient generates as many FCert as the number of the foreign providers.

C. FOREIGN SERVER REGISTRATION (FSR) PROTOCOL

This protocol registers the patient with a foreign server using the FCert.

(1) The mobile device generates a message (M) which contains the foreign index pseudonym of the patient (FIP_i^f) and the FCert certificate ($FCert_i^f$), i.e., M = (FIP_i^f||$FCert_i^f$).

(2) It then generates a digital signature (σ) on the message (M) using the ESigGen algorithm with temporal private key (tPR_i). This private key corresponds to the temporal public key (tPK_i) stated in the FCert certificate, i.e., σ = ESigGen(M, tPK_i).

(3) The message (M) is encrypted using the RSA public key of foreign server (RPK_f), i.e., Me = Enc(RPK_f, M).

(4) The mobile device constructs a registration request (fRegReq) message. This message contains the ID of the foreign server (ID_f) and the encrypted message M*e*, i.e. fRegReq = (ID_f||Me||σ).

(5) The mobile device sends the request message to the foreign server.

(6) The mobile device generates the foreign shared key (FSK_i^f) using the EKeyExg algorithm (i.e. FSK_i = $tPR_i^f * EPK_f$).

The foreign server receives the fRegReq message and do the following.

(7) It decrypts the message using its private key (RPR_f), i.e., M = Dec(Me, RPR_f).

(8) It extracts the patient's $FCert_i^f$ and verifies the following: a) the home server's signature on the FCert using its proxy public key (PPK_h) as an input to the blind signature verification (BlndSigVer) algorithm, b) the FCert is within its validity period (24 hours), c) the patient's digital signature using his/her tPK_i^f stored in the FCert.

(9) If verification is successful, the foreign server stores both the foreign index pseudonym (FIP_i^f) and the patient's certificate ($FCert_i^f$) in its database. Then, it initialises the index nonce for the patient.

(10) The foreign server generates a foreign shared key by using EKeyExg algorithm. (i.e., FSK_i^f = tPK_i^f * EPR_f).

(11) The foreign server generates a message (M) which contains the index nonce (Inc).

(12) The message is then encrypted with the foreign key (FSK_i^f), i.e., Me = E(FSK_i^f, M).

(13) The foreign server generates the MAC on the message (M*e*) using the foreign shared key (FSK_i^f), i.e., MAC = HMAC(Me, FSK_i^f).

(14) The foreign server constructs the registration response message (fRegRes). The response message contains the ID of the foreign server (ID_f), Me, the foreign index pseudonym of the patient (FIP_i^f), and MAC, i.e., fRegRes = (ID_f||FIP_i^f||Me||MAC). It then sends the response message to the mobile device.

The mobile device receives the fRegRes message and does the following.

(15) It verifies the MAC using the foreign shared key (FSK_i^f).

(16) If the MAC is successfully verified, the mobile device uses the same key to decrypt the encrypted part of the message, i.e., Md = D(Me, FSK_i^f). Then, it stores index nonce (Inc) to be used later when requesting an uploading service.

The patient repeats the registration process with the selected foreign servers. Then, the patient moves to the uploading process.

D. DATA UPLOADING PROTOCOL

This protocol is executed between the patient and a foreign provider. The purpose of this protocol is to allow the patient to upload his/her data to the foreign provider. It should be noted that, the patient has a number of foreign servers to upload to. The patient shuffle randomly among the servers to upload the data.

(1) The patient's mobile device generates the 2EMPGHD using a double signing and encryption method.

(2) Then, the patient's mobile device generates a foreign request pseudonym ($FRP_{i,r}^f$) using the FRPs-Gen algorithm.

(3) Subsequently, it constructs an uploading request (UpReq) message. This message contains the foreign server ID (ID_f), the patient's ID ($FRP^f_{i,r}$), and the patient's data ($2EMPGHD_i$).
(4) Then, the mobile device generates a MAC on the message by using the foreign shared key (FSK^f_i).
(5) The mobile device then sends an uploading request (UpReq) message to the foreign server.

$$\textbf{Patient}_i \rightarrow \textbf{ID}_\textbf{f}:(\textbf{ID}_\textbf{f}||\textbf{nFRP}^\textbf{f}_{\textbf{i,r}}||\textbf{2EMPGHD}||\textbf{MAC}).$$

The foreign server receives the request message and performs the following.

(6) It uses its RSA private key (RPR_f) as an input to the FRP-Lnk algorithm to decrypt the ($FRP^f_{i,r}$) and find the temporal elliptical curve public key (tPK^f_i) of the patient and other information related to the patient (i.e., nonce, priority of the data). The foreign shared key can be found by multiplying the (EPR_f) with the (tPK^f_i). The result is the foreign shared key which is used to verify the MAC.
(7) The foreign server next validates the index nonce. This validation is to guarantee the freshness of the foreign request pseudonym (i.e., protect against replay attack). It then checks that the foreign pseudonym certificate (FCert) of the patient has not expired (i.e., within 24 hours). If both verifications are correct, the foreign server sends an acknowledgement to the patient.
(8) Next, the foreign server checks the priority tag (PR). If the priority tag is set, it sends a notification to the patient's home server. Otherwise, it stores the patient's 2EMPGHD in its database.

The patient can continue uploading on the same server or select randomly another server to upload.

VIII. REQUIREMENT ANALYSIS

- The SPID framework supports mutual anonymous authentication. As we can see the patient is using anonymous credentials to access service providers. The patient requests a home pseudonym certificate (HCert) from the HCP to register with the selected home server. The patient is using a number of foreign pseudonym certificates (FCerts) to register with any foreign servers. These FCerts are generated by the patient and are blindly signed by the home server. The registration with any server is done by sending the certifiacte (i.e., HCert or FCert) along with the digital signature. For repeated authentication with the same server, the patient uses a fresh request pseudonym each time. The authentication of these pseudonyms is achieved by verifying the MAC associated with the message. To verify the MAC without looking for the key in the database, we use a brilliant method. When a server receives the request pseudonym, the server decrypts the request pseudonym to get the underlying index pseudonym. Recall that, the underlying index pseudonym is an ECC pubic key. So, the server multiplies index pseudonym with the server's ECC private key. This multiplication process results in the shared key. This key is used to verify the request.
- The SPID framework preserves patient's ID privacy (P). The patient's real ID is protected, as the patient uses an elliptical curve public key (EPK_i) as a home index pseudonym only the HCP knows the real ID of the patient. The linkability of the patient's foreign index pseudonym IDs that are used as identities for the patient across service providers are only linked by the patient's home provider to the patient's home index pseudonym. The patient uses a new request pseudonym for each upload. This is to protect the sessions from being linked by unauthorized entity. Only the same server can link these request pseudonyms to the patient's index pseudonym used to identify the patient with that server. The patient uses a number of foreign servers to upload data and not to stick to one server.
- The SPID framework supports end-to-end data authenticity (S2). In the data uploading protocol, the patient generates a MAC on the upload message before sending it to a foreign server. By using the MAC, the foreign server guarantees that the message has not been altered from the point at which it was dispatched. The key used to generate the MAC is generated from the private key of the patient. This guarantees that no other entity can generate the key and the message is from the claimed patient. In the forwarding protocol (i.e., from foreign to home), the patient's data is in the form of 2EMPGHD (i.e., double signed and encrypted). When the patient's data arrives at the home server, it can verify the authenticity of this data using a home shared key which is only known to the patient and the home server. In the second forwarding protocol (i.e., from home to HCP), the patient's data is still in the form of EMPGHD. When the patient's data arrives at the HCP, it can verify the authenticity of this data using a shared key which is only known to the patient and the HCP.
- The SPID framework supports end-to-end data confidentiality(S3). The patient's data that is uploaded on any foreign server is double encrypted using two keys. Therefore, no entity can decrypt and learn anything about the patient's data.

IX. SECURITY ANALYSIS

In this section, the SPID is analysed against threats. Here, Alice is an authorised patient. Eve is an adversary.

- SPID is protected against an impersonation attack. Suppose that Eve learns Alice's certificate (i.e., HCert or FCert), Eve is attempting to play Alice's role and deceive a data collection server. Eve sends the certificate to the data collection server, and the server accepts the certificate from Eve. The data collection server finds that the certificate has been sent before by Alice, and in this case asks Eve to prove knowledge of the private key which corresponds to the public key stated in the certificate.

Eve needs to prove to the data collection server that she is the owner of the certificate, so she needs to generate a digital signature using the private key which corresponds to the public key stated in the certificate. Eve fails to generate the digital signature as she does not have the private key. By design, Eve fails to impersonate Alice.

- The SPID is protected against the man in the middle attack. Suppose a malicious server intercepts the messages between the patient and home server to make the patient believe it is the home server. The patient sends the HCert to the malicious server to register. The malicious server verifies both the HCP's signature on HCert and Alice's signature. The malicious server generates the shared key by multiplying its elliptical curve private key with the patient's elliptical curve public key (stated in the HCert). By using the generated shared key, the malicious server generates a MAC on response message and sends it to the patient. The patient receives the message from the malicious server and then verifies the MAC associated with the message using the home shared key. The verification step fails because the malicious server generates the MAC using its key, not a key which the patient used to generate the home shared key.
- The SPID is protected against linkability attack. As explained in the protocol section each type of protocol involves using different pseudonyms. In the registration protocol, the patient uses a different index pseudonym and certificate for each server. In addition each request carried out by the patient with any server involves using fresh request pseudonyms, one for each request. Thus linking these request pseudonyms to the same patient is quite difficult, as we will explain in the degree of anonymity section.
- SPID is protected against a data forgery attack launched by an internal entity. Suppose a home server tries to generate data for Alice. The home server cannot generate data for Alice. This is because the home server needs to obtain Alice's shared key (SK) (i.e., the key between Alice and the HCP) to generate the EMPGHD. Further if the foreign server tries to generate data for Alice, the foreign server needs to obtain two keys. The first key is the SK used to generate the EMPGHD and the second is the home shared key (HSK) (i.e., the one between the patient and home server) to generate the 2EMPGHD, which is not feasible.
- SPID is protected against data forgery attack launched by an external entity. Suppose Eve has captured Alice's uploading message; as explained previously the message contains the following fields: Alice's ID (i.e., request pseudonym), Alice's data (i.e. EMPGHD), and the MAC generated on the message fields. Suppose Eve generates the MAC on the request pseudonym and her data. The foreign server decrypts the request pseudonym, extracts the index pseudonym, and multiplies the index pseudonym after decoding it (recall this is a public key) with the foreign server's elliptical curve private key. This process results in a shared key. The key will not verify the MAC associated with the message so the server discards the message.
- SPID is protected against replay attack. Suppose Eve tries to replay uploading messages to disturb the server. Eve forwards Alice's old uploading messages to a foreign server. After the server performs all the verifications, the server verifies the freshness of the index nonce, and discovers that the messages sent by Eve are old and so discards the messages.
- SPID is protected against repudiation attacks. Messages exchanged between Alice and any server contain a MAC generated by Alice's shared key. As explained previously, generating the shared key involves using both parties' private keys. The knowledge of the private key requires solving the elliptic curve discrete logarithm problem (ECDLP), which has been proven to not be computationally feasible. Thus, it is very hard for Eve to generate a message on behalf of Alice. So Alice can not claim that an entity has generated a message of behalf of her.

X. DEGREE OF ANONYMITY

The degree of anonymity provided by pseudonym generation and linkage method measured by using the Shannon entropy method [29]. What we need to show is that the patients are indistinguishable from the attacker. If the attacker could determine a particular patient to be the generator of a pseudonym by any means, we can say that the anonymity provided by the pseudonym generation and linkage method is not acceptable.

To calculate the degree of anonymity provided by the method, we assume that (X) is a discrete random variable which represents a pseudonym. This pseudonym can be correctly linked to a certain patient (i) from a set of (N) patients. In mathematics, this can be represented as ($p_i = Pr(X = i)$), where Pr is the probability. The current entropy (H(X)) of the corresponding pseudonym can be calculated as:

$$H(X) = -\sum_{i=0}^{n} p_i \log(p_i) \tag{7}$$

The maximum entropy of the system (H(M)) can be calculated as:

$$H(M) = \log N \tag{8}$$

The degree of anonymity (d) provided by the method can be calculated as:

$$d = H(X)/H(M), 0 \leq d \leq 1. \tag{9}$$

1) When $d = 0$, the attacker knows the generator of the pseudonym (p_i) with probability 1.
2) When $d = 1$, all patients appear as being an originator with the same probability.

Suppose that we have 100 patients (i.e., N = 100). Each patient generates several pseudonyms and the attacker cannot distinguish one particular patient as the owner of these

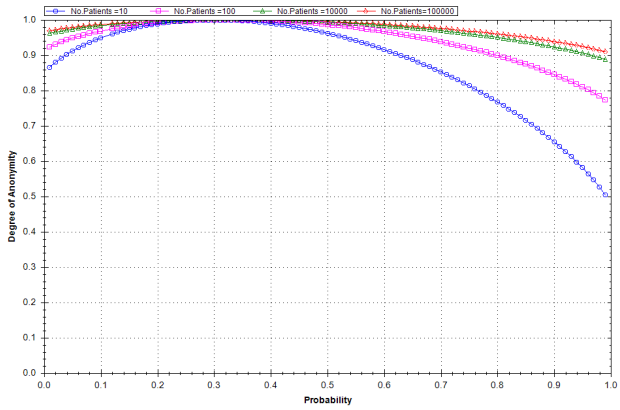

FIGURE 7. Degree of anonymity.

pseudonyms. However, the attacker can divide the patients into say two groups (G_1 and G_2), G1 has 40 patients and G2 has 60 patients. Then the attacker assigns each group a probability of generating a set pseudonyms as follows.

$$G_1 = Pr/40, 0 \leq i \leq 40 \quad G_2 = 1 - Pr/60 \quad 41 \leq i \leq 60 \tag{10}$$

In Equation 10, we have two groups of patients, one with 40 patients and the other with 60 patients. Patients belonging to the same group are seen by the attacker as having the same probability of generating the pseudonyms.

Figure 7 shows that the degree of anonymity (d) is proportional to the number of patients. When the number of patients is 100, the maximum degree of anonymity (d = 1) is achieved for the probability distribution (p = .41). The degree of anonymity is equal to 0.8 when one group is assigned the probability p = .93 and the number of patients is (N = 10,000). However, the anonymity does not drop to zero even in the case that we have only two patients in the system. This is because the attacker sees all patients as potential owners of the pseudonyms.

XI. SPID PERFORMANCE

In this section we are going to measure the performance of the SPID protocols theoretically using queuing theory.

- The software which is used to implement the SPID protocols is Java 2 Platform, Standard Edition (J2SE). Java provides the implementation of several cryptographic primitives and key management services required in our protocols.
- The performance metrics used for the SPID protocols evaluation are the average time a patient spends waiting in a queue, and the average response time of the system. We want these metrics to be within reason.
- To measure the execution time for each protocol, we use a Java benchmarking tool called Java Microbenchmark Harness (JMH) [30].
- To prototype the protocols, a desktop computer running Windows 10 with a 1.99 GHz Intel Core i7 and 8GB of RAM is used. The timing results from the protocols execution presented here are based on this computer specifications.
- The queuing theory [31] is used to predicate the performance of protocol.

A. QUEUING THEORY

Queuing theory [31] helps to predict the following: an average waiting time a user spends in the queue, the total response time of the system, the number of users in the system, and the average utilization of the system. There are two models in queuing theory, the single sever model and the multi-server model. The single server queuing model (M/M/1) assumes there is only one server in the system. The multi-server queuing model (M/M/C), where C is the number of servers. To predict the performance metrics of our protocols, we applied queuing theory. Queuing theory uses mathematical formulas to theoretically analyze the performance metrics of a system and they are as follows.

Mathematical Formulas for the Single Queue Model:

- λ = mean arrival rate of patients (average number of patients arriving per unit of time).
- μ = mean service rate (average number of patients that can be served per unit of time).
- $\rho = \lambda/\mu$ = the average utilization of the system.
- $L = \lambda/(\mu - \lambda)$ = the average number of patients in the service system.
- $L_Q = \rho * L$ = the average number of patients waiting in line.
- $W = 1/(\mu - \lambda)$ = the average time spent waiting in the system, including service time.
- $W_Q = \rho * W$ = the average time spent waiting in line.

Mathematical Formulas for Multi-Server Queuing Model:

- s = the number of servers in the system.
- $\rho = \lambda/(s * \mu)$ = the average utilization of the system.
- $P_0 = [\ \sum_{n=0}^{s-1}(\lambda/\mu)/n! + ((\lambda/\mu)^s/s! * (1/1-\rho))\ ]^{-1}$ = the probability that no patients are in the system.
- $L_Q = P_0(\lambda/\mu)^s\rho/s!(1-\rho)^2$ = the average number of patients waiting in line.
- $W_Q = L_Q/\lambda$ = the average time spent waiting in line.
- $W = W_Q + 1/\mu$ = the average time spent in the system, including service time.
- $L = \lambda * W$ = the average number of patients in the service system.

To theoretically analyze the performance of a protocol, we first need to determine the following inputs: the service time (m), the arrival rates of the requests (λ), and the service rate (μ). The service time (m) is the time the server needs to fulfil each request. For example, the service time for a registration request is the sum of the execution time of each method executed on the server side, listed in Table 3, which is equal to 0.75 seconds. The service rate μ can be calculated as (1/m).

FIGURE 8. The FCertG protocol performance using 3 servers.

TABLE 1. Queue values for FCertG protocol.

λ	ρ	L_Q	L	W_Q	W	P_0
Servers = 1						
95.5	28.7	0.12	0.402	0.001	0.004	71.3
100.5	30.2	0.130	0.432	0.001	0.004	69.8
150.5	45.1	0.372	0.823	0.002	0.005	54.8
200.5	60.2	0.91	1.51	0.005	0.008	39.84
250.5	75.2	2.27	3.025	0.009	0.012	24.84
300.5	90.2	8.260	9.2	0.027	50.03	9.8
325.5	97.5	38.182	39.16	0.125	0.128	2.5
Servers = 3						
100.5	10.2	0.00	0.302	0	0.003	73.9
300.5	30.1	0.030	0.93	0	0.003	40.28
500.5	50.1	0.238	1.74	0	0.003	21.01
700.5	70.1	1.154	3.3	0.002	0.005	9.54
800.5	80.1	2.6	5.004	0.003	0.006	5.6
900.5	90.1	7.4	10.11	0.008	0.011	2.5

1) FCertG PROTOCOL PERFORMANCE

Here we describe the performance of the FCertG protocol. The service time for the protocol is the summation of the execution time of the methods executed on the home server listed in Table 2. Table 1 shows the performance of our system using the mathematical formulas for the single and multiple queuing models. The overall average of the waiting time and response time of our system using one server are 0.88 seconds and 0.95 seconds respectively. In the case of 3 servers, the arrival rates should not exceed $\mu * s$ which is (333.3*3) = 999.9 requests per second. We selected the arrival rates to range from 100.5 to 900.5. The minimum waiting time and response time arise when the arrival rate is 100.5 requests per second and the maximum waiting time and response time arise when the arrival rate reaches 900.5 requests per second. The overall average of the waiting time and response time of our system using 3 servers are 0.17 seconds and 0.19 seconds respectively. Figure 8 shows the performance of the protocol.

2) FSR PROTOCOL PERFORMANCE

Here we describe the performance of the foreign server registration protocol. The service time for the protocol is the summation of the execution time of the methods executed on the server listed in Table 3. Table 4 shows that we have calculated the performance of our system using the mathematical formulas for the single and multiple queuing models. The overall average of the waiting time and response time of our system using one server are 9.9 seconds and 10.6 seconds respectively. In the case of 3 servers, the arrival rates should not exceed the $\mu * s$ which is (1.33*3) = 3.9 requests per second. We selected the arrival rates to range from 1.5 to 3.9. The minimum waiting time and response time arise when the arrival rate is 1.5 requests per second. The maximum waiting time and response time arise when the arrival rate reaches 3.9 requests per second. The overall average of the waiting time and response time of our system using 3 servers are 2.2 seconds and 2.9 seconds respectively. We enhance the system by 70% by using 3 servers.

TABLE 2. FCertG protocol execution time.

Operations	Key Size	Time (ms)
The patient executes the following algorithms		
Retrieve DataBase	-	0.293
EKeyGen	256	0.293
RsaEncryption	2048	3.1
BlndMsgGen	-	0.594
HmacGen256	256	0.003
The home server executes the following algorithms		
RsaDecryption	2048	3.1
EMul	256	0.1
HmacVer256	256	0.003
BlndSigGen	256	0
HmacGen256	256	0.003
The patient executes the following algorithms		
HmacVer256	256	0.003
BlndSigDrv	-	0
BlndSigVer	256	0.236
Store DataBase	-	3.4

TABLE 3. Registration protocol execution time.

Operations	Key Size	Time (ms)
The patient executes the following algorithms		
Retrieve DataBase	-	0.293
ESigGen	256	1.1
doEPHD	256	1.7
RsaEncryption	15360	3.1
The server executes the following algorithms		
RsaDecryption	15360	729.2
CertVer	-	1.6
ESigVer	256	1.6
doEPHD	256	1.7
Store DataBase	-	3.4
HmacGen256	256	0.003
AesEncryption	256	0
The patient executes the following algorithms		
AesDecryption	256	0
HmacVer256	256	0.003
Store DataBase	-	3.4

3) DATA UPLOADING PROTOCOL PERFORMANCE

Here we describe the performance of the data uploading protocol. The service time for the protocol is the summation of the execution time of the methods executed on the server

TABLE 4. Queue values for registration protocol.

λ	ρ	L_Q	L	W_Q	W	P_0
Servers = 1						
1	75.1	2.278	3.03	2.278	3.03	24.8
1.15	86.4	5.524	6.389	4.804	5.556	13.53
1.19	89.47	7.605	8.500	6.391	7.143	10.526
1.25	93.98	14.685	15.625	11.748	12.500	6.015
1.29	96.99	31.280	32.250	24.248	25.00	3.0
Servers = 3						
1.5	37.6	0.07	1.2	0.049	0.80	31.8
1.9	47.6	0.19	1.6	0.10	0.85	22.8
2.5	62.7	0.65	2.5	0.261	1.013	13.2
2.9	72.7	1.42	3.6	0.45	1.24	8.4
3.5	82.7	5.6	8.2	1.6	2.3	3.1
3.9	97.7	41.5	44.4	10.6	11.4	0.5

TABLE 5. Uploading protocol execution time.

Operations	Key Size	Time (ms)
The patient executes the following algorithms		
Retrieve DataBase	-	0.293
2 AesEncryption	256	0.004
RsaEncryption	2048	3.1
2 HmacGen256	256	0.016
HmacGen265	2048	0.003
The foreign server executes the following algorithms		
RsaDecryption	2048	3.1
EMul	256	0.1
HmacVer256	256	0.003
Store DataBase	256	3.4
HmacGen256	256	0.003
The patient executes the following algorithms		
HmacVer256	256	0.003

TABLE 6. Queue values for uploading protocol.

λ	ρ	L_Q	L	W_Q	W	P_0
Servers = 1						
28.6	20.0	0.1	0.3	0.002	0.01	80
56.6	39.6	0.3	0.7	0.005	0.01	60.4
84.6	59.2	0.9	1.5	0.01	0.02	40.8
112.6	78.8	2.9	3.7	0.03	0.03	21.2
140.6	98.4	59.6	60.6	0.42	0.4	1.6
Servers = 3						
85.7	20	0.01	0.61	0	0.01	54.8
171.4	39.9	0.1	1.3	0	0.01	29.4
256.4	59.8	0.52	2.3	0	0.01	14.7
341.4	79.6	2.51	4.9	0	0.01	5.7
426.4	99.5	183.5	186.5	0.430	0.44	0.12

listed in Table 5. Table 6 shows that we have calculated the performance of our system using the mathematical formulas for the single and multiple queuing models. The overall average of the waiting time and response time of our system using one server are 6.8 seconds and 0.24 seconds respectively. In the case of 3 servers, the arrival rates should not exceed $\mu * s$ which is (142.85*3) = 428.6 requests per second. We selected arrival rates ranging from 85.7 to 426.4. We can see that the waiting time and the response time are negligible.

XII. DISCUSSION

The SPID framework has the following features that are not found in other related works. The SPID allows the patient to select where to store the data and who is responsible for delivering the data to the final destination (HCP). The SPID allows the patient to generate certificates and pseudonym IDs to access any service providers. In SPID, the authentication per uploading can be done without searching in the foreign provider's database for the verification key. This is achieved by using some properties of the ECC. The SPID allows the patient to double sign and encrypt the data before upload so no service providers can know the content of the patient's data. The SPID allows the patient to not be recognized based on the communication pattern by using different service providers and pseudonyms and the patient is responsible for choosing a new set of service providers everyday. The SPID framework uses a combination of different cryptographic building blocks to enhance performance. For example, the request pseudonym is generated based on an elliptical curve public key (EPK) and some random text. Then, the EPK is encrypted using the server's RSA public key to form the request pseudonym. So when the pseudonym arrives at the server, the server decrypts the pseudonym using the server's RSA private key to discover the EPK. Then, the server multiples the EPK with its elliptical curve private key (EPR) to get the verification key and verify the message.

XIII. CONCLUSION

To protect the security and ID privacy in data collection distributed system we designed a secure anonymous data collection framework called SPID. The proposed framework has advantages of using multiple service providers to collect a patient's data to prevent a single provider from inferring the patient's identity based on the pattern of interactions, allow the patient to generate pseudonym identities and certificates to access these service providers, and allow each patient's home service provider for anonymously linking the patient's data which are scattered across different foreign service providers. Then, the home provider of the patient delivers the patient's data after aggregation to the healthcare provider. From the design requirements, it is understood that the SPID framework has advantages that are not supported in other related works. The security analysis shows the strength of the SPID in preventing a number of security attacks. From the performance result, we found the SPID performance is acceptable when the number of servers increased. The Shannon entropy method showed that it is hard to distinguish one patient as the generator of the a set of pseudonyms.

REFERENCES

[1] K. T. Kadhim, A. M. Alsahlany, S. M. Wadi, and H. T. Kadhum, "An overview of patient's health status monitoring system based on Internet of Things (IoT)," *Wireless Pers. Commun.*, vol. 114, pp. 2235–2262, May 2020.

[2] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K.-S. Kwak, "The Internet of Things for health care: A comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, 2015.

[3] G. Paliwal and A. W. Kiwelekar, "A comparison of mobile patient monitoring systems," in *Proc. Int. Conf. Health Inf. Sci.* Springer, 2013, pp. 198–209.

[4] P. Pawar, V. Jones, B.-J. F. van Beijnum, and H. Hermens, "A framework for the comparison of mobile patient monitoring systems," *J. Biomed. Informat.*, vol. 45, no. 3, pp. 544–556, 2012.

[5] Microsoft Reporter. (Jan. 2018). *The NHS is About to Take an 'Important' Step Into the Cloud, Says Microsoft*. [Online]. Available: https://news.microsoft.com/en-gb/2018/01/19/the-nhs-is-about-to-take-an-important-step-into-the-cloud-says-microsoft/
[6] P. Kakria, N. K. Tripathi, and P. Kitipawang, "A real-time health monitoring system for remote cardiac patients using smartphone and wearable sensors," *Int. J. Telemed. Appl.*, vol. 2015, pp. 1–11, Dec. 2015.
[7] J. Lopez and R. Dahab, "An overview of elliptic curve cryptography," Tech. Rep., 2000.
[8] W. Tang, K. Zhang, D. Zhang, J. Ren, Y. Zhang, and X. S. Shen, "Fog-enabled smart health: Toward cooperative and secure healthcare service provision," *IEEE Commun. Mag.*, vol. 57, no. 5, pp. 42–48, May 2019.
[9] J. H. Abawajy and M. M. Hassan, "Federated Internet of Things and cloud computing pervasive patient health monitoring system," *IEEE Commun. Mag.*, vol. 55, no. 1, pp. 48–53, Jan. 2017.
[10] Q. Shen, X. Liang, X. Shen, X. Lin, and H. Y. Luo, "Exploiting geo-distributed clouds for a E-health monitoring system with minimum service delay and privacy preservation," *IEEE J. Biomed. Health Informat.*, vol. 18, no. 2, pp. 430–439, Mar. 2014.
[11] G. Wang, R. Lu, and Y. L. Guan, "Achieve privacy-preserving priority classification on patient health data in remote eHealthcare system," *IEEE Access*, vol. 7, pp. 33565–33576, 2019.
[12] R. Boussada, B. Hamdane, M. E. Elhdhili, and L. A. Saidane, "Privacy-preserving aware data transmission for IoT-based e-health," *Comput. Netw.*, vol. 162, Oct. 2019, Art. no. 106866.
[13] R. Lu, X. Lin, and X. Shen, "SPOC: A secure and privacy-preserving opportunistic computing framework for mobile-healthcare emergency," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 3, pp. 614–624, Mar. 2013.
[14] X. Liang, R. Lu, L. Chen, X. Lin, and X. Shen, "PEC: A privacy-preserving emergency call scheme for mobile healthcare social networks," *J. Commun. Netw.*, vol. 13, no. 2, pp. 102–112, Apr. 2011.
[15] X. Lin, R. Lu, X. Shen, Y. Nemoto, and N. Kato, "Sage: A strong privacy-preserving scheme against global eavesdropping for eHealth systems," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 4, pp. 365–378, May 2009.
[16] E. Marin, M. A. Mustafa, D. Singelée, and B. Preneel, "A privacy-preserving remote healthcare system offering end-to-end security," in *Proc. Int. Conf. Ad-Hoc Netw. Wireless*. Springer, 2016, pp. 237–250.
[17] J. Xu, K. Xue, S. Li, H. Tian, J. Hong, P. Hong, and N. Yu, "Healthchain: A blockchain-based privacy preserving scheme for large-scale health data," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8770–8781, Oct. 2019.
[18] A. D. Dwivedi, G. Srivastava, S. Dhar, and R. Singh, "A decentralized privacy-preserving healthcare blockchain for IoT," *Sensors*, vol. 19, no. 2, p. 326, Jan. 2019.
[19] E. Birrell and F. B. Schneider, "Federated identity management systems: A privacy-based characterization," *IEEE Security Privacy*, vol. 11, no. 5, pp. 36–48, Sep./Oct. 2013.
[20] S. Cantor and T. Scavo, "Shibboleth architecture," *Protocols Profiles*, vol. 10, p. 16, Sep. 2005.
[21] A. Dey and S. Weis, "PseudoID: Enhancing privacy in federated login," Tech. Rep., 2010.
[22] J. Maheswaran, D. Jackowitz, E. Zhai, D. I. Wolinsky, and B. Ford, "Building privacy-preserving cryptographic credentials from federated online identities," in *Proc. 6th ACM Conf. Data Appl. Secur. Privacy*, Mar. 2016, pp. 3–13.
[23] T. Lenz and V. Krnjic, "Towards domain-specific and privacy-preserving qualified eID in a user-centric identity model," in *Proc. 17th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun./12th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Aug. 2018, pp. 1157–1163.
[24] S. S. Chow, Y.-J. He, L. C. Hui, and S. M. Yiu, "SPICE—Simple privacy-preserving identity-management for cloud environment," in *Proc. Int. Conf. Appl. Cryptogr. Netw. Secur.* Springer, 2012, pp. 526–543.
[25] C. Li, Q. Wu, H. Li, and J. Liu, "Trustroam: A novel blockchain-based cross-domain authentication scheme for Wi-Fi access," in *Proc. Int. Conf. Wireless Algorithms, Syst., Appl.* Springer, 2019, pp. 149–161.
[26] Y. Yao, X. Chang, J. Mišić, V. B. Mišić, and L. Li, "BLA: Blockchain-assisted lightweight anonymous authentication for distributed vehicular fog services," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3775–3784, Apr. 2019.
[27] J. Hoffstein, J. Pipher, J. H. Silverman, and J. H. Silverman, *An Introduction to Mathematical Cryptography*, vol. 1. Springer, 2008.
[28] D. Chaum, "Blind signature system," in *Advances in Cryptology*. Springer, 1984, p. 153.
[29] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 5, no. 1, pp. 3–55, 2001.
[30] (2019). *Code Tools JMH*. [Online]. Available: https://openjdk.java.net/projects/code-tools/jmh/
[31] C.-H. Ng and S. Boon-Hee, *Queueing Modelling Fundamentals: With Applications in Communication Networks*. Hoboken, NJ, USA: Wiley, 2008.

TAHANI HAMAD ALJOHANI received the B.Sc. and M.Sc. degrees in computer science from King Abdulaziz University, Jeddah, Saudi Arabia, and the Ph.D. degree in computer science from The University of Manchester, Manchester, U.K. Her current research interests include information security, security and ID privacy in distributed environments, and the Internet of Things (IoT).

NING ZHANG received the B.Sc. degree in electronics engineering from Dalian Maritime University, Dalian, China, and the Ph.D. degree in electronic engineering from the University of Kent, Canterbury, U.K. She is currently a Senior Lecturer with the School of Computer Science, The University of Manchester, Manchester, U.K. Her current research interests include security in networked and distributed systems, applied cryptography, data privacy, and trust and digital rights management. She has authored papers and acted as a referee and a reviewer in these topic areas.