Secure Data Exchange in P2P Data Sharing Systems in eHealth Perspective

Mehedi Masud\textsuperscript{1} and Sk. Md. Mizanur Rahman\textsuperscript{2}

\textsuperscript{1} Computer Science Department, Taif University
Taif, 21974, KSA

\textsuperscript{2} School of Information Technology and Engineering, University of Ottawa
Ottawa, ON, Canada

Abstract
In P2P data sharing systems (P2PDSS) peers share data in a pair-wise fashion. Data are shared on-the-fly by establishing temporary data exchange session for user queries. Generally, the communication link between peers is unsecured while exchanging data. In P2P eHealth data sharing scenarios, peers may need to exchange highly confidential data among them. Hence, there are some security threats that need to be considered (e.g. data might be trapped and disclosed by the intruders). In a P2PDSS, we cannot assume any third party security infrastructure (e.g. PKI) to protect confidential data. Considering the need of secure data exchange in P2PDSS, in this paper we propose a secure data exchange model. The model is based on pairing-based cryptography and the data sharing policy between peers. Applying the model, peers compute secret session keys dynamically by computing pairing on elliptic curve, based on the data sharing policies while exchanging data. The proposed protocol is robust against the man-in-the middle attack, the masquerade attack and the replay attack. 

Keywords: eHealth, P2P, data security, PKI

1. Introduction

A peer in a P2P Data Sharing System (P2PDSS) works as a client/server according to the policy of data exchange between the peers, and it is a highly scalable system. The local databases on peers are called peer databases. In P2PDSS, there is no global mediated schema like in the traditional data integration systems, where a global mediated schema is required for data exchange. There is an increasing interest in the creation of peer-to-peer database systems, which includes establishing and maintaining mappings between peers, processing queries using appropriate propagation techniques, and exchanging data between peers [11,12,13,14]. While there is a rich body of research concerning frameworks and mapping issues among peers, the aspect of sharing data between trusted or acquainted peers in an anonymous and secured way is given less attention. Due to the security holes, P2PDSS is not being adopted in a practical scenario such as eHealth data sharing systems.

In a peer-to-peer system, we cannot assume a fixed secure channel for data exchange between each pair of peers since peers are dynamic and may leave the network anytime, or acquaintances between peers are temporary. Moreover, it would be highly expensive and not feasible to maintain a secure link for each pair of peers. When data are exchanged through an unsecured link between acquainted peers, data are no longer secured despite the assumption that each source protects its own data from malicious tampering and accessing by external intruders. The following example illustrates the need to use a security policy for exchanging confidential data between peers in eHealth P2PDSS. This scenario relates to an 'eHealth network', where different parties (e.g. family physicians; walk-in clinics; hospitals; medical laboratories; pharmacists, and other stakeholders) are willing to share data about patients' treatments, medications, and test results over an insecure network.

Example 1 Consider a scenario of an eHealth P2PDSS. In the system, family doctors (FDB), hospitals (HDB), medical laboratories (LABDB), pharmacists (PHDB), and other stakeholders (e.g. medical research cells (RDB)) are willing to exchange or coordinate information about patients' treatments, medications, test results, and diseases. In the system, a peer may need to exchange data with other related peers according to established policies between them. For example, family doctors may want to keep track of patients' medications for some specific diseases. Therefore, family doctors should have an acquaintance with the pharmacist database (PHDB), and any patient in PHDB diagnosed with a disease that is of interest to family doctors may need to be exchanged with FDB. Moreover, family doctors may be interested in collecting test results of their patients from laboratories and the medications that their patients take while staying...
at hospitals. The acquaintances between peers are formally a set of mappings or mapping constraints.

The acquaintances between peers are established with predefined policies and trust relationships without having a centralized security policy. But, centralized-trusted control system is needed for the public key infrastructure (PKI). Therefore, the existing conventional PKI is not suitable to apply in eHealth P2PDSS. Recent progress of Elliptic Curve Cryptography (ECC) [1], Identity-Based Cryptography (IBC) [2], and Pairing-based cryptography (PBC) [3] show that it is feasible to implement PBC on ECC. Furthermore, studies have shown that ECC consumes considerably less resources than conventional public key cryptography (PKC) for a given security level [4].

In order to achieve secure data exchange in an eHealth P2PDSS dynamic network, this paper presents a protocol based on Identity Based Encryption (IBE) and PBC. Using bilinear properties, each peer in the network generates a dynamic secret session key based on the attributes mentioned in the query and the predefined data exchange policy. In this protocol, peers authenticate each other in a pair-wise fashion without a centralized authentication policy. The protocol is mainly a query-based secure session key generation for secure data exchange between peers. There is not enough available research work directly related to the secure data exchange in P2PDSS. The only work that is close to the proposal is the work of [5], where the authors claim secure data propagation among multiple nodes by using pre-existing friendship relationships among the nodes in the network.

In brief, our protocol has the following properties: (1) flexible message-oriented secure data exchange between peers (2) exchange of data between peers without any third party certificates (3) communication between peers could be as simple as a single TCP connection (4) both parties (i.e. source and target) authenticate each other during data exchange. The initial version of the paper has appeared in [6].

Organization of The Paper: The next section introduces the primitives of cryptography that are necessary to describe our proposed protocol. Section 3 describes how the data exchange policy/mapping is established between two peers. In Section 4, the paper presents our cryptographic solution and describes the proposed protocol for exchanging data between peers. In section 5, we discuss issues of cryptographic implementation and prevention of different attacks. Finally, Section 6 concludes and points out avenues for further research.

2. Cryptographic Primitives

In this section, we describe some basic cryptographic primitives which are useful to implement and understand our proposed protocol.

Let $G_1$ be an additive group and $G_2$ be a multiplicative group of the same prime order $q$. Let $P$ be an arbitrary generator of $G_1$. Note that $aP$ denotes $P$ added to itself $a$ times. Assume that the discrete logarithm (DL) problem is hard in both $G_1$ and $G_2$. We can think of $G_1$ as a group of points on an elliptic curve over $F_q$, and $G_2$ as a subgroup of the multiplicative group of a finite field $F_q^*$ for some $k \in Z_q^*$, where $Z_q^* = \{\xi | 1 \leq \xi \leq q-1 \}$. A mapping $e: G_1 \times G_1 \rightarrow G_2$, satisfying the following properties, is called a cryptographic bilinear map.

- **Bilinearity:** $e(aP, bQ) = e(P, Q)^{ab} = e(bP, aQ) \in G_2$ for all $P, Q \in G_1$ and $a, b \in Z_q^*$. This can be restated in the following way. For all $P, Q, R \in G_1$; then $e(P+Q, R) = e(P, R) e(Q, R) = e(Q, R) e(P, R)$ $\in G_2$ and $e(P, Q+R) = e(P, Q) e(P, R) = e(P, R) e(P, Q) \in G_2$.

- **Non-degeneracy:** If $P$ is a generator of $G_1$, then $e(P, P)$ is a generator of $G_2$. In other words, $e(P, P) \neq 1$.

- **Computable:** A mapping is efficiently computable if $e(P, Q)$ can be computed in polynomial-time for all $P, Q \in G_1$.

Modified Weil Pairing [3] is an example of cryptographic bilinear map.

Let the group $G_1$ represents the group of points on the elliptic curve $E: Y^2 = X^3 + aX + b \mod \tau$, where $\tau$ is a prime number, then using the group $G_1$, we can define the following hard cryptographic problems applicable to our proposed protocol.

- **Computational Diffie-Hellman (CDH) Problem:** Given a triple $(P, aP, bP) \in G_1$ for $a, b \in Z_q^*$, find if there exists any element $abP \in E$.

- **Decisional Diffie-Hellman (DDH) problem:** Given a quadruple $(P, aP, bP, cP) \in G_1$ for $a, b, c \in Z_q^*$, decide whether $c=ab \mod q$ or not.

- **Gap Diffie-Hellman (GDH) Problem:** A class of problems where the CDH problem is hard but the DDH problem is easy.

**Bilinear Diffie-Hellman (BDH) Problem:** Given a quadruple $(P, aP, bP, cP) \in G_1$ for some $a, b, c \in Z_q^*$, compute $e(P, P)^{abc}$. 


3. Secure Data Exchange Setup

In this section, we introduce the concept of data sharing settings between peers in a P2PDSS and then discuss different security threats that can happen during the exchange of data between peers through an unsecured channel.

Attributes are symbols taken from a given finite set \( U = \{A_1, \ldots, A_k\} \) called the universe. We use the letters A, B, C, … to denote single attributes and X, Y, … to denote sets of attributes. Each attribute \( A_j \) is associated with a finite set of values called the domain of \( A_j \) and is denoted by \( \text{dom}(A_j) \). Suppose \( X = \{A_1, A_2, \ldots, A_k\} \subseteq U \), with the elements \( A_i (1 \leq i \leq k) \) taken in the order shown, then \( \text{dom}(X) \subseteq \text{dom}(A_1) \times \text{dom}(A_2) \times \ldots \times \text{dom}(A_k) \). A non-empty subset of \( U \) is called a relation schema \( R \). A database schema is a finite collection \( \mathcal{R} = \{R_1, \ldots, R_m\} \) of relation schemas.

Let \( S \) be a schema at a peer \( P_i \) and \( T \) be a schema at another peer \( P_j \). If a data exchange policy is specified from \( S \) to \( T \), then we call \( S \) a source schema and \( T \) a target schema. Each peer has instances corresponding to its schema. Next we discuss the data exchange settings.

Generally, in data exchange settings [7], source-to-target data exchange policies are constituted by a set of assertions of the forms

\[
\Sigma_{st} = q_S \rightarrow q_T
\]

where, \( q_S \) and \( q_T \) are two queries, respectively over the source schema \( S \), and over the target schema \( T \). Intuitively, an assertion \( q_S \rightarrow q_T \) specifies that the concept represented by the query \( q_S \) over the sources corresponds to the concept in the target schema represented by the query \( q_T \). The assertions are basically tuple-generating dependencies [8]. Assertions can be specified as logical expressions of the form:

\[
\forall x[\exists y \phi(x,y) \rightarrow \exists z \psi(x,z)]
\]

where, the left-hand side (LHS) of the implication, \( \phi \), is a conjunction of relation atoms over the schema of \( S \) and the right-hand side (RHS) of the implication \( \psi \) is a conjunction of relation atoms over the schema \( T \). The policy expresses a constraint about the appearance of a tuple in the instance satisfying the constraint of the RHS, given a particular combination of tuples satisfying the constraint of the LHS. Basically, the policies provide a structural relationship of data between source and target as well as allowing data to be exchanged between the two. Through the policies, a source also exports part of its schema accessible to the target. The following is a simple example of a data exchange setting.

**Example 2** Consider a family physician database (FDDB) in Example 1 with the schema \( S \) consisting of two relations \( R_1(\text{OHIP}, \text{DOB}, \text{Name}, \text{Address}) \) and \( R_2(\text{OHIP}, \text{TestName}, \text{Result}, \text{Date}) \). Also consider a database in a medical research cell (RDB) with the schema \( T \) consisting of a relation \( R_3(\text{OHIP}, \text{Name}, \text{Illness}, \text{DOB}, \text{TestName}, \text{Result}) \). Assume the following policy is assigned between \( S \) and \( T \):

\[
\forall_{\text{ohip, name, illness, dob, testname, result}} \exists_{\text{name, address, illness, dob}} R_3(\text{ohip, name, address, illness, dob}, R_2(\text{ohip, testname, result, date}) \rightarrow R_3(\text{ohip, illness, dob, testname, result})
\]

The policy expresses that patients’ data (ohip, name, illness, dob, testname, result) are exchanged from FDDB to RDB. It also shows that the attributes \( \{\text{OHIP}, \text{Illness}, \text{DOB}, \text{Name}, \text{TestName}, \text{Result}\} \) are shared between FDDB and RDB. Although the attributes are shared for RDB, they also contain some confidential attributes e.g. \( \{\text{OHIP}, \text{DOB}\} \) that should not be exposed to others by any means during the exchange. We can say that these attributes are more confidential compared to the attributes \( \{\text{TestName}, \text{Result}\} \), since the values of those attributes do not have any meaning unless one knows corresponding OHIP and date of birth. Note that only the source knows which attributes are confidential attributes among the shared attributes. The administrator of the source is responsible to distinguish shared and confidential attributes. Note that in this paper we only consider the schema-level mappings between a source and a target. We assume that when the mappings are created only the source and the corresponding target know the structural relationship between their schemas (i.e., correspondences between the attributes and relations). The structural relationship is not known to other peers. Therefore, during the exchange of data in an unsecured channel, we need a protocol that secures confidential information of shared attributes.

Now we formally define the shared attributes, confidential attributes, and non-confidential attributes as follows:

**Definition 1** [Shared attributes] Consider two peers \( P_i \) and \( P_j \) in a P2PDBS. Let \( S \) be a schema with a set of attributes \( U_s \) in \( P_i \) and \( T \) be a schema with a set of attributes \( U_t \) in \( P_j \). Assume a policy \( \Sigma_{st} = q_S \rightarrow q_T \) between \( P_i \) and \( P_j \). Let \( \text{att}(\Sigma_{st}) \) denote the set of attributes exposed by \( P_i \) using the policy \( \Sigma_{st} \). Therefore, the shared attributes, denoted by \( SA \), are \( SA \subseteq U_i = \text{att}(\Sigma_{st}) \).

**Definition 2** [Confidential attributes] Consider a data sharing policy between two peers \( P_i \) and \( P_j \) is \( \Sigma_{st} = q_S \rightarrow q_T \).
Let $SA$ be the set of shared attributes. Therefore, the confidential attributes, denoted by $CA$, are $CA \subseteq SA$.

**Definition 3** [Non-confidential attributes] Consider a data sharing policy between two peers $P_i$ and $P_j$, $\Sigma_{i}=q_s \rightarrow q_t$. Let $SA$ be the set of shared attributes and $CA$ be the set of confidential attributes. Hence, the non-confidential attributes, denoted by $NCA$, are $SA - CA$.

**Definition 4** [Private attributes] Consider the data sharing policy $\Sigma_{i}=q_s \rightarrow q_t$ between two peers $P_i$ and $P_j$ and let $SA$ be the set of shared attributes, the private attributes, denoted by $PA$, is $U_s - SA$.

**Example 3** Consider example 2. Based on the data sharing policy, we see that the shared attributes are $\{Ohip, Illness, DOB, TestName, Result\}$, the confidential attributes are $\{Ohip, DOB\}$, and the non-confidential attributes are $\{Illness, TestName, Result\}$. Note that administrators of the peers implicitly define the attributes that are confidential during the creation of policies.

### 4. Description of the proposed protocol

In a P2P DSS, a peer may act as a source and/or a target, therefore source and target peers are responsible to generate secret session key for a specific data exchange session. For exchanging data from a source peer $P_i$ to a target peer $P_j$ source-to-target, data exchange policies are constituted. Thus if the target $P_j$ requests data from the source $P_i$ by a query, then the source provides data depending on the query request and according to the data exchange policies. As there is no established security mechanism between the source and the target, hence there could be an attack (i.e., man-in-the-middle, masquerade Attack) on the communication. To prevent the attacks, an "on-the-fly" security setup is needed between the source $P_i$ and the target $P_j$, based on the query.

Assume a source peer $P_i$ with schema $S$ and a target peer $P_j$ with schema $T$. Also assume that based on the data exchange policy between $P_i$ and $P_j$ the shared attributes are classified as follows:

Confidential attributes ($CA$) = $\{CA_1, CA_2, ..., CA_m\}$

Non-confidential attributes ($NCA$) = $\{NCA_1, NCA_2, ..., NCA_p\}$

The purpose of the security protocol is to ensure secure data exchange when $P_j$ requests data from $P_i$ through a query $Q_i$ that contains confidential attributes as well as non-confidential attributes. Assume a query $Q_i$ at any time instance $t$ is requested from $P_j$ to $P_i$. Before forwarding the query $Q_i$, $P_j$ generates system as well as session parameters.

**System parameters:** System parameters (e.g., group, bilinear map, hash function) are used for generating secret session keys for data exchange between peers. Depending on the mutual agreement between peers, system parameters may be fixed for each data exchange session or they may be changed for each session.

**Session parameters:** Session parameters (e.g. dynamically generated id of peers, random number in $Z_q^*$, random numbers) are used for a specific data exchange session in order to generate the secret session key. These parameters are dynamic for each session of data exchange.

In order to request data from $P_j$, peer $P_i$ generates the following system and session parameters.

**System parameters:**

- $G_1$, an additive group of prime order $q$.
- $H_1: \{0,1\}^* \rightarrow G_1$, a collision resistant cryptographic hash function which maps from arbitrary-length strings to points in $G_1$.

**Session parameters:**

- $ID_{P_j} = H_1(P_j^\gamma) \in G_1$, a dynamically generated id of peer $P_j$, where $\gamma$ is a random number.

After creating the parameters $< G_1, H_1, ID_{P_j} >$, peer $P_j$ sends the parameters with the query $Q_i$ to $P_i$. When $P_i$ receives the parameters and the query, it identifies the confidential and non-confidential attributes. Assume $P_i$ identifies the following confidential and non-confidential attributes from the query $Q_i$:

Confidential attributes in $Q_i$, denoted by $CA_{Qi} = \{QCA_1, QCA_2, ..., QCA_m\} \subseteq CA$

Non-confidential attributes in $Q_i$, denoted by $NCA_{Qi} = \{QNCA_1, QNCA_2, ..., QNCA_p\} \subseteq NCA$

When $P_i$ receives the parameters from $P_j$, it also generates system and session parameters for computing a secret session key for the authentication of $P_j$ and for encryption of the query result, $Q_i^S$. The generated parameters are given below.

**System parameters:**

- $G_2$, a multiplicative group of the same prime order $q$ as the order of the additive group $G_1$.
- A bilinear map $\langle e \rangle: G_1 \times G_1 \rightarrow G_2$. 


Depending on the query attributes, session key \( K_{\sigma} = H_{1}(P_{i}^* \times \Pi_{q} \times \zeta) \), where \( Z_{q}^{*} = \{ \mu | 1 \leq \mu \leq q^{\prime} \} \), \( H_{1} : \{0,1\}^{*} \rightarrow \{0,1\}^{*} \); a mapping from arbitrary-length strings to \( \lambda \)-bit fixed length string.

**Session parameters:**

- An ID \( ID_{P_{i}} = H_{1}(P_{i}^*) \in G_{1} \), where \( \zeta \) is a random number.
- A random number \( R_{i-SESSION} \) which is used for generating the authentication code \( Aut_{0} \).

Depending on the confidential and non-confidential attributes, \( P_{i} \) now generates the secret session key \( K_{Si} \) and authentication code \( Aut_{0} \) using its own parameters and the parameters of \( P_{j} \). The generation and purpose of \( K_{Si} \) and \( Aut_{0} \) are discussed as follows:

### 4.1 Generation of Secret Session Key and Authentication Code

In identity-based crypto there is generally a private key generator (PKG) which entities use in order to obtain their private keys. This is a trusted authority (like a CA in a PKI). In our proposed protocol there is no PKG but still our protocol works properly. In this proposed security protocol, the responsibilities of a PKG are mutually performed by the source and the target.

The source \( P_{i} \) computes a shared secret element in \( Z_{q^{*}} \), called a shared secret parameter and denoted as \( \sigma \) based on the query attribute sets \( CA_{Qi} \) and \( NCA_{Qi} \) as follows:

\[
\sigma = H_{2}(NCA_{Qi} \times CA_{Qi}) \in Z_{q^{*}}
\]

\( P_{i} \) also computes another shared secret identity in \( G_{1} \), called shared secret identity, denoted by \( ID_{SP} \) based on the query attribute set \( CA_{Qi} \) as follows:

\[
ID_{SP} = H_{1}(CA_{Qi}) \in G_{1}
\]

Depending on the query attributes, session key \( K_{Si} \) for each session is generated by the source \( P_{i} \) as follows:

\[
K_{Si} = \sigma \times e(\Pi_{q}, ID_{SP})
\]

Source \( P_{i} \) also generates authentication code \( Aut_{0} \) as follows:

\[
Aut_{0} = H_{3}(K_{Si} || ID_{P_{i}} || ID_{P_{j}} || R_{i-SESSION} || 0)
\]

where \( R_{i-SESSION} \) is a random number generated by the source \( P_{i} \) to distinguish every session from each other so that a replay attack cannot take place on the communication.

Finally, source \( P_{i} \) sends the system parameters < \( G_{3}, \sigma, H_{2}, H_{3} \) > including the session parameters < \( ID_{P_{i}}, K_{i-SESSION}, Aut_{0} \) > to the target \( P_{j} \). After receiving the system parameters as well as session parameters from the source \( P_{i} \), target \( P_{j} \) generates \( \sigma \) and \( ID_{SP} \). Finally target \( P_{j} \) computes a session key \( K_{Sj} \) as follows:

\[
K_{Sj} = e(\Pi_{q}, ID_{SP})
\]

\[
= e(\sigma, ID_{SP}) = e(\Pi_{q}, e(\Pi_{q}, ID_{SP}))
\]

\[
= e(\Pi_{q}, ID_{SP}) = e(\Pi_{q}, ID_{SP})
\]

Target also computes the verification code \( Ver_{0} \) as follows:

\[
Ver_{0} = H_{3}(K_{Si} || ID_{P_{i}} || ID_{P_{j}} || R_{i-SESSION} || 0)
\]

The verification code \( Ver_{0} \) is computed to verify the authentication code \( Aut_{0} \) of \( P_{i} \). Target \( P_{j} \) compares \( Ver_{0} \) with \( Aut_{0} \); if \( Ver_{0} = Aut_{0} \) then target \( P_{j} \) generates another authentication code \( Aut_{1} \) as follows:

\[
Aut_{1} = H_{3}(K_{Si} || ID_{P_{i}} || ID_{P_{j}} || R_{j-SESSION} || R_{i-SESSION} || 1)
\]

\( R_{j-SESSION} \) is a random number generated by the target and different from each session so that replay attack (request to source) cannot take place in the communication. Finally, \( P_{j} \) sends \( < Aut_{1}, R_{i-SESSION} > \) to source \( P_{i} \).

Upon receiving \( < Aut_{1}, R_{i-SESSION} > \) from the target \( P_{j} \), source \( P_{i} \) generates another verification code \( Ver_{1} \) as follows, and compares it with \( Aut_{1} \).

\[
Ver_{1} = H_{3}(K_{Si} || ID_{P_{i}} || ID_{P_{j}} || R_{j-SESSION} || R_{i-SESSION} || 1)
\]

If \( Ver_{1} \) matches \( Aut_{1} \), i.e \( (Ver_{1} = Aut_{1}) \) then source peer sends the data of the query result \( Q_{R} \) by encrypting it with the private session key \( K_{Si} \).

For distinguishing the computation of authentication codes by the source and the target and the communication of the authentication codes between the source and the target, "0" and "1" are used.

### 4.2 Secure Authenticated Data Exchange

After authentication between the source and the target, source \( P_{i} \) generates a message authentication code, denoted by \( MAC_{MESSAGE} \) on query result \( Q_{R} \), which is computed as \( MAC_{MESSAGE} = H_{3}(Q_{R}) \). The source also encrypts \( Q_{R} \) with its secret session key \( K_{Si} \), denoted by
CIPHER\textsubscript{Q\_i}, which is computed as CIPHER\textsubscript{Q\_i}=E\_KS(Q\_i), where E\_KS means encryption using the session key K\_Si. Finally, P\textsubscript{i} sends the following packet to P\textsubscript{j}:

\[
< \text{ID}_{P_i}, \text{CIPHER}_{Q\_i}, \text{MAC}_{MESSAGE}, \text{ID}_{P_j} >
\]

After receiving the packet, P\textsubscript{j} decrypts CIPHER\textsubscript{Q\_i} with the session key K\_Si (denoted as D\_KS(CIPHER\textsubscript{Q\_i})) and generates the verification message authentication code, denoted by VER\_MESSAGE, which is computed as follows:

\[
\text{VER\_MESSAGE} = H_3(D\_KS(CIPHER\textsubscript{Q\_i}))
\]

Finally, P\textsubscript{i} compares VER\_MESSAGE with MAC\_MESSAGE. If VER\_MESSAGE = MAC\_MESSAGE, then the data is accepted.

The whole process is illustrated in Figure 1 and described in the following steps:

**STP 1:** A query Q\textsubscript{i} is generated at the target P\textsubscript{j}.

**STP 2:** Target P\textsubscript{i} determines group G\textsubscript{1}, hash function H\textsubscript{1} and performs the following steps:
1. Generates an ID ID\textsubscript{P_{ij}}; 2. Sends \(< G\textsubscript{1}, H\textsubscript{1}, Q\textsubscript{i}, ID_{P_{ij}} >) to the source P\textsubscript{j}.

**STP 3:** Source P\textsubscript{j} executes the query Q\textsubscript{i} on its local database and performs the following steps:
1. Determines group G\textsubscript{2}, bilinear mapping function \(-e, \) and cryptographic hash functions H\textsubscript{2} and H\textsubscript{3}.
2. Generates an ID ID\textsubscript{P_{ij}}, a random number R\textsubscript{i\_SESSION}.
3. Generates secret session key K\_Si, authentication code Aut\textsubscript{0}.
4. Sends \(< G\textsubscript{2}, -e, H\textsubscript{2}, H\textsubscript{3}, ID_{P_{ij}}, R_{i\_SESSION}, \text{Aut}_0 >) to P\textsubscript{j}.

**STP 4:** Target P\textsubscript{j} generates session key K\_Si, verification code Ver\textsubscript{0}.
1. Generates R\textsubscript{i\_SESSION}; and compares Ver\textsubscript{0} with Aut\textsubscript{0}; if Ver\textsubscript{0} = Aut\textsubscript{0} then generates Aut\textsubscript{1}.
2. Sends < R\textsubscript{i\_SESSION}, Aut\textsubscript{1} > to the source P\textsubscript{j}.

**STP 5:** Source P\textsubscript{j} generates verification code Ver\textsubscript{1}.
1. Compares Ver\textsubscript{1} with Aut\textsubscript{1}; if Ver\textsubscript{1} = Aut\textsubscript{1} then generates message authentication code MAC\_MESSAGE.
2. Encrypts query result Q\textsubscript{R}, with the session key K\_Si, denoted as CIPHER\textsubscript{Q\_R}, MAC\_MESSAGE, ID\textsubscript{P_{ij}} > to the target P\textsubscript{j}.

**STP 6:** Target decrypts CIPHER\textsubscript{Q\_R} with session key K\_Si; generates verification message authentication code VER\_MESSAGE; compares VER\_MESSAGE with MAC\_MESSAGE; if VER\_MESSAGE = MAC\_MESSAGE; then data is exchanged successfully.

5. Cryptographic Implementation and Attack Analysis

In this section we discuss the cryptographic implementation overhead and the prevention of different attacks.

5.1 Communication Overhead

Communication overhead for our proposed protocol can be evaluated in terms of packet sizes that are transmitted by the source and the target peer over the communication link during the key setup and authentication phase, described in section 4.1 and 4.2.

Communication overhead for the target peer P\textsubscript{j} is two packets: (1) First packet = (Descriptor Packet for G\textsubscript{1} + Descriptor Packet for H\textsubscript{1} + 160 bit + Descriptor Packet for Q\textsubscript{i}), where |G\textsubscript{1}|=160 bit; (2) Second packet = (< Aut\textsubscript{0}, R\textsubscript{i\_SESSION} >) = (160 bit HMAC output + 160 bit random number ). Communication overhead for the source peer P\textsubscript{j} is two packets: (1) First packet = (< G\textsubscript{2}, -e, H\textsubscript{2}, H\textsubscript{3} > ) = (Descriptor Packet for G\textsubscript{2} + Descriptor Packet for -e + Descriptor Packet for H\textsubscript{2} + Descriptor Packet for H\textsubscript{3} ); (2) Second Packet = (< ID\textsubscript{P_{ij}}, R_{i\_SESSION}, Aut\textsubscript{0} > ) = (|G\textsubscript{1}| element + 160 bit random number + 160 bit HMAC output )

5.2 Computation Cost

The total computation cost for both the source and target peers together is: 2 pairing computations, 2 point additions, 2 point multiplications (for deriving the
symmetric key), 4 hash evaluations on $H_1$, 2 hash evaluations on $H_2$, 4 hash evaluations on $H_3$, and 2 random number generations. Among all the computation tasks, pairing computations are undoubtedly the most time-consuming task [9], but recent progress of Tate pairing computation on elliptic curves of characteristic 2 and 3 has been significantly improved [10], which is more realistic in security applications for pairing-based cryptosystems. Hence, we can conclude that the real-time computation intensity in our protocol is quite acceptable.

5.3 Man-in-the-middle Attack (MITM)

In MITM attack, an intruder can establish independent connections with the source and the target by forging the authentication policy of a valid source/target. In our proposed protocol the secret keys $K_S$ and $K_T$ are generated based on the confidential and the non-confidential attributes that are only shared between the source and the target peers. Therefore, an intruder node cannot generate a session key in the middle of a data exchange session between two peers. Thus, man-in-the-middle attack is not effective on the proposed protocol.

5.4 Masquerade Attack

In this attack, an attacker peer may pretend to be a valid target of a source by disguising its own identity and publishing the identity of a real target. Thus, a malicious peer may gain access to the data of the source. In this proposed protocol, peers authenticate each other before exchanging data. Furthermore, in every session of data exchange between peers, parameters (session/system) are generated dynamically. The session parameters $<R_{i\text{-Session}}, Au_{i0}, Au_t, R_{j\text{-Session}}>$ are completely different in each session. Hence, by storing these session parameters and using these parameters in challenge/response session during authentication phase, an intruder node cannot pass the authentication process. Therefore, the intruder cannot pretend to be a valid peer in the data exchange. Thus, a masquerade attack is prevented.

6. Conclusion

We have presented a novel secure data exchange protocol for an eHealth P2PDSS using pairing-based cryptography and data exchange policy between peers. Using the protocol, any two peers that need to exchange data over an insecure medium can generate on-the-fly a secret session key by exchanging some system and session parameters. An important feature of the proposed protocol is that peers always generate a new session key for every new data exchange session; therefore, every session is completely independent with respect to the session key generation. Furthermore, the proposed protocol is robust against man-in-the-middle attack, masquerade attack and the replay.

References